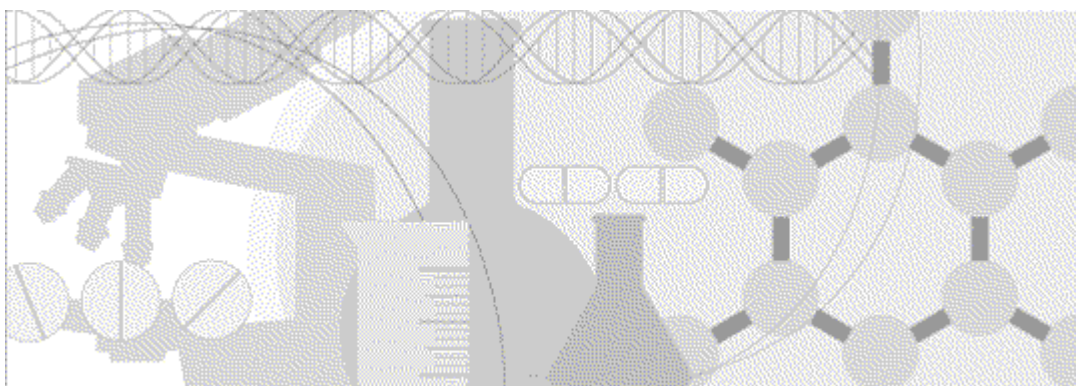


Setting Up a Trial with InForm Architect and MedML Guide

Oracle[®] Health Sciences InForm 4.6.5



ORACLE[®]

Copyright © 1998, 2015, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

This documentation may include references to materials, offerings, or products that were previously offered by Phase Forward Inc. Certain materials, offerings, services, or products may no longer be offered or provided. Oracle and its affiliates cannot be held responsible for any such references should they appear in the text provided.

Contents

About this guide xiii

Overview of this guide.....	xiv
Audience	xiv
Documentation	xv
Documentation accessibility.....	xvii
If you need assistance.....	xviii
Finding InForm information and patches on My Oracle Support.....	xviii
Finding Oracle documentation	xix
Finding prerequisite software for Oracle Health Sciences applications	xix

Chapter 1 Introduction to the InForm Architect application 1

InForm Architect application features.....	2
Setting up new trials.....	2
Enhancing existing trials	2
Debugging.....	2
The InForm Architect application and MedML	3
About trial components.....	4
Trial building blocks.....	4
Starting with the Base trial	6
Customizing Base trial data.....	6
Learning with sample trials	7
Planning a trial.....	8
Planning forms.....	8
Planning edit checks.....	10
Planning CRF help	12
Planning data mappings	13
Planning user setup	14
Planning reports.....	14

Chapter 2 Quick tour 15

Overview	16
Starting and stopping the InForm Architect application.....	17
InForm Architect application window.....	18
Trial Objects window.....	19
Properties window.....	21
Properties tab	21
Mappings tab.....	22
Output window	23
Design Workspace.....	24
Protocol Editor window.....	25
Form editor window	26
Element Editor window.....	30
Rule editor window	31
Execution Plan window	35
Data Mappings window	37
Toolbars.....	39
Main toolbar	39
Edit toolbar.....	40
Rules toolbar	41
Trial toolbar.....	42

Object Palette toolbar.....	42
Settings toolbar.....	43
Customizing the application workspace.....	45
Customizing toolbars.....	45
Customizing windows.....	47
Using editing and navigation features.....	49
Modifying trial components.....	49
Cutting, copying, and pasting.....	51
Searching for text strings.....	54
Working with bookmarks.....	55
Deleting trial components.....	56
Chapter 3 Setting up a trial	57
Overview.....	58
About trials.....	59
Creating a new trial.....	60
Opening an existing trial.....	64
Design and production mode.....	64
Closing a trial.....	66
Inserting MedML into a trial.....	67
Specifying additional paths.....	67
Specifying strict MedML processing.....	68
Installing the sample trial.....	68
Saving a trial.....	69
Saving a copy of the contents of an editor window.....	69
Saving all components.....	70
Saving active trial components.....	72
Saving by component.....	72
Installing trial component definitions in the database.....	73
Working with a source control system.....	73
Moving a trial into production.....	74
Chapter 4 Defining visits	75
About visits.....	76
Creating a new visit.....	78
Creating a visit definition.....	78
Defining visit properties.....	79
Renaming a visit.....	81
Reordering visits in a trial.....	82
Deleting a visit from a trial.....	83
Chapter 5 Defining forms	85
About forms.....	86
Types of forms.....	86
Form definition hierarchy.....	87
RefNames.....	89
UUIDs.....	91
Position and caption properties.....	91
Design notes.....	93
Creating a new form.....	94
Creating a form definition.....	94
Defining form properties.....	95
Adding form subcomponents.....	98
Managing forms and visits.....	99
Adding a form to a visit.....	99
Reordering a form in a visit.....	101

Removing a form from a visit	101
Defining a section	102
Creating a new section	102
Defining section properties	102
Adding section subcomponents	103
Including an existing section definition in a form	104
Defining an item	105
Creating a new item	105
Defining item properties	105
Adding controls to an item	107
Including an existing item definition in a section	107
Defining an itemset	108
Creating a new itemset	108
Defining itemset properties	108
Adding items to an itemset	110
Viewing the items in an itemset	111
Including an existing itemset definition in a section	112
Using key items	113
Enhancing navigation in repeating forms	113
Checking for data uniqueness	115
Defining controls	119
Nesting controls	120
Creating a new control—general instructions	121
Using an existing control definition	122
Defining a group, radio, or checkbox control	122
Defining a pulldown control	124
Defining a text box control	125
Defining a date/time control	128
Defining a calculated control	131
Defining a simple control	132
Defining element components	134
Creating an element definition	134
Navigating in the Element Editor window	135
Defining units	136
Creating a unit definition	136
Example of conversion rule for Inches unit	138
Example of conversion rule for Centimeter unit	138
Annotating forms	139
Annotated View	140
Annotated Data Mappings Display	141
Annotated Data Mappings Tables Display	141
Selecting metadata for display	142
Companion tables	145
Time and Events Schedule	149
Cover page	150
Viewing the annotated trial design	150
Printing the annotated trial design	151

Chapter 6 Defining data mappings 153

Overview	154
About data mappings	155
Two methods for creating data mappings	155
Mapping data to a Customer-Defined Database	156
About CDD mappings	156
Setting up a CDD	156
Types of CDD data transfer	157

Designing CDD mappings	159
CDD mapping and generic visits.....	167
Auto-generating CDD mappings	168
Controlling column length for radio group mappings.....	168
Mapping new forms to an auto-generated CDD.....	169
Creating manual CDD mapping definitions	170
Creating a CDD mapping definition object	170
Creating a CDD mapping definition table.....	172
Creating a CDD mapping definition table column	174
Defining CDD mapping table column properties.....	178
Specifying properties for a control path.....	180
Ordering columns in a table	180
Mapping a form control	181
Mapping associations to a CDD	186
Association table schema definition.....	186
Creating an association mapping manually.....	187
Defining association mapping properties.....	188
Mapping data to the Clintrial software	189
Definitions of terms in Clintrial mappings	189
Data mapping components	190
Autogeneration and manual definition.....	193
Clintrial mapping and generic visits	194
Setting up Clintrial data transfer processing.....	195
Mapping considerations—InForm application to Clintrial software	196
Autogenerating Clintrial mappings	199
Overview of autogenerating mappings.....	199
Creating an autogenerated mapping definition	199
Mapping date time controls—Principles.....	200
Mapping date time controls—Automatic mappings.....	202
Mapping new forms to an autogenerated mapping definition	203
Creating manual Clintrial mapping definitions.....	204
Creating a Clintrial mapping definition object	204
Mapping definition object properties	206
Deleting a data mapping definition component	207
Creating a panel mapping definition	207
Panel mapping properties	208
Creating a panel item mapping definition	210
Item mapping properties.....	215
Creating a control path mapping definition.....	218
Control path mapping properties	222
Customizing Clintrial mappings.....	223
Autogenerated codelists and mapping definitions.....	223
CONTEXT panel mappings	224
Mapping date time controls—Manual mappings	225
Creating mappings without a control path	226
Modifying Clintrial Item Order in a panel	226
Using page keys and block keys.....	227
Making panel property changes	229
Implementing Clintrial subsets	231
Creating a type 0 panel	231
Mapping autocoded data	233
Autocode mapping workflow.....	233
Differences between Autocode mapping and Central Coding mapping.....	233
Creating an autocode mapping	234
Specifying a source control.....	236
Mapping data for coding with Central Coding.....	239
Definitions of Central Coding mapping terms.....	239

Central Coding mapping workflow	240
Differences between Autocode mapping and Central Coding mapping.....	241
Creating coding mapping definitions	241
Viewing coding mappings	257
General instructions for using the Build RefName tool.....	259
Mapping data to Oracle Clinical.....	260
Setting up Oracle Clinical processing	260
About Oracle Clinical mappings.....	261
Creating an Oracle Clinical mapping definition	261
Defining a CPE Reference.....	263
Defining a control mapping	268
Displaying RefName paths for all controls.....	269
Defining an individual control mapping	271
Maintaining mappings.....	274
Viewing mapping definition MedML.....	274
Updating a data mapping definition.....	274
Deleting a data mapping definition component	275
Removing a mapping definition.....	275
Viewing Data Mappings for a form control	275
Viewing the form to which a control is mapped	276
Viewing data mappings in annotated trial design	277

Chapter 7 Defining rules 279

Overview	280
About rules.....	281
Rules and events	281
Types of rules.....	281
Browser and server rules	283
About rule contexts.....	284
About arguments	288
About rule dependencies.....	289
Creating rules.....	293
Creating a new rule definition	293
Specifying rule properties.....	294
Writing a rule script.....	295
Defining default arguments	298
Creating a context.....	300
Viewing summarized context and dependency counts.....	308
Specifying context-specific arguments and values	311
Defining additional rule dependencies	314
Using the Build RefName tool.....	316
Rule objects and methods	319
Form rule and calculation script examples.....	319
Testing rules.....	323
Creating test data	323
Running rule scripts against test data.....	324
Running rule scripts against actual patient data	326
Debugging rule scripts	327
Rule script debugging examples.....	327
Using the Patient.OutputDebug method.....	329

Chapter 8 Defining events and execution plans 331

About events.....	332
Creating events.....	333
Creating a new event.....	333
Specifying event properties.....	333

Associating an existing event with a rule..... 334
 Creating an execution plan..... 335
 Execution plan objects and methods 337
 Examples of scripts that use execution plan objects..... 337
 Debugging execution plans..... 338
 Using the Global.OutputDebug method 338

Chapter 9 Enabling specialized InForm application features 339

Overview 340
 Alternate versions of forms 341
 Implementing an alternate form 341
 Example of implementing an alternate form..... 342
 Associations 343
 Overview..... 343
 Defining an association 343
 Including repeating forms in an association 344
 Coding dictionaries..... 346
 Coding dictionary XML 346
 Dictionary XML restrictions 347
 Dictionary XML updates 347
 Common forms..... 348
 Defining a common form 348
 When entry is not required 348
 Links to CRF Help 349
 Overview - Creating CRF Help links 349
 Creating a CRF Help bookmark 349
 Defining a HelpLink component 350
 Including a HelpLink component in a CRF Help document..... 350
 Date display format 352
 Dynamic forms..... 353
 Overview - Implementing dynamic forms 353
 Creating a dynamic form in a visit..... 353
 Creating a dynamic form calculation..... 353
 Removing a dynamic form..... 354
 Dynamic visits 355
 Overview - Implementing dynamic visits 355
 Creating a dynamic visit definition 355
 Creating a rule for selecting a dynamic visit schedule..... 355
 Removing a dynamic visit 356
 Monitoring forms 357
 Creating a Visit Report 357
 Creating a Regulatory Document Checklist..... 358
 Patient navigation mode 359
 Enabling patient navigation modes 359
 Patient ID information 360
 Patient Initials data entry item..... 360
 Patient Number data entry item..... 360
 Editable patient ID..... 361
 Recommendation for note in patient ID section..... 362
 Patient identification and patient record transfer 362
 Patient record transfer affidavit..... 364
 Customizing the patient record transfer affidavit..... 364
 Patient transfer affidavit XML 364
 Randomization 366
 Randomization sequences..... 367
 Drug Kit section and randomization control..... 368
 Randomization rule..... 368

Randomization source manager configuration	369
Randomization sequence format configuration.....	370
Randomization source database setup.....	370
ODBC connection for the randomization database	374
Configuring the trial to use the randomization DSN	376
Repeating forms	377
Creating a repeating form	377
Enhancing repeating form navigation	377
Reports.....	378
Calculations.....	378
Date of Visit	378
Itemset column headers	379
Required visits and items.....	379
Study completion and dropout reason	379
Study version acceptance dates	380
Subject states	380
Visit start hour	380
Screening and enrollment forms	381
Screening UUIDs	381
Enrollment UUIDs	382
Screening or enrollment copy calculation rule	383
Using Patient Initials and Patient Number items	383
Forms requiring signature	384
Overview of implementing electronic signatures	384
Creating the text of a signing affidavit.....	385
Creating a signature group	386
Associating a form with a signature group	389
Signature invalidation.....	391
Displaying a required signature list on a CRF	392
Study Completion form	393
Required Study Completion form components.....	393
Form design examples	395
Unscheduled or repeating visits	401
UUIDs	402
Conflict visit	402
Date Of Visit item	403
Enrollment form.....	403
Patient ID form	404
Patient Initials item.....	404
Patient Number item	405
Randomization	405
Regulatory Documents (Reg Docs) form	405
Repeating visit and reporting DOV	406
Screening form	406
Screening section with Patient Initials	407
Sequences	407
Study Completion form.....	408
Visit Report form	409
Unit UUIDs.....	409

Chapter 10 Customizing predefined lists 411

Overview of customizing predefined lists	412
Customizing predefined lists.....	413
Changing reason text for clearing a CRF or editing an item	413
Changing query text	414
Types of query text strings.....	415

Chapter 11 Performing trial administration activities 417

Overview 418

About InForm application administrators 419

About administrative data 420

 Users 420

 Rights 421

 Groups 425

 Sites 425

 Events 425

 Rules 425

 Configuration information 426

 System 426

 Sponsors 426

 Sequence numbers 426

Using the Admin tree in the Trial Objects window 427

Managing users 429

 Creating a user 429

 Assigning or updating a user password 431

 Managing group assignments 432

 Managing site assignments 435

 Assigning rights to a user 435

 Removing rights from a user 436

 Updating user status 436

Managing rights groups 438

 Creating rights groups 438

 Assigning users to rights groups 439

 Assigning display overrides to rights groups 440

Managing groups 443

 Types of groups 443

 Creating groups 444

 Assigning users to groups 445

 Creating item groups 446

 Assigning items to item groups 446

Managing sites 448

 Creating sites 448

 Assigning users to sites 449

Defining sponsors 451

 Creating a sponsor 451

Managing numbering 452

 Creating sequence types 452

 Creating sequences 452

 Specifying sequence number format 453

 Loading sequence number formats 455

Defining system configuration settings 456

 Setting configuration parameters with SYSCONFIG 456

Chapter 12 Updating study definition data 457

Overview 458

 Methods for updating the trial 458

 MedML tags for trial updates 458

Creating new form versions 459

Updating a study version 460

 Updating the STUDYVERSION tag 460

 Updating the STUDYVERSIONDOC tag 461

Activating a study version 462

Updating the STUDYVERSIONSITE tag	462
Revising without updating the study version	463
Revising a section on a form	463
Revising an item on a form	464
Metadata Update report	466

Chapter 13 Defining electronic trial documentation 467

Overview	468
Choosing an authoring tool	469
Creating a trial protocol	470
Modifying the trial protocol HTML files	471
Creating a documentation definition XML file	472
Examples of documentation definition XML files	473
Trial protocol documentation definition	473
CRF Help documentation definition	474
Visit Calculator and Sample Case Book documentation definition	475
Creating a Table of Contents file	476
Example of a Table of Contents file	476
Linking between document files	477
Example of a complete link	477
Example of a Table of Contents file with links	478
Trial documentation and study versions	479
Creating CRF Help	481
Modifying CRF data entry help files	481
Creating a FAQ placeholder	482
Creating a Rule Help placeholder	483
Revising trial documentation	485

Appendix A HTML Tags and Special Characters 487

Overview	488
HTML formatting tags	489
HTML special characters	490
Disallowed characters	493

Appendix B Creating custom reports 495

About custom reports	496
What is involved	496
Output formats	496
Reporting COM object properties by function	498
The Reporting COM object	500
Creating custom text reports	506
ASP basics for text reports	506
Filtering data queries	507
Simple text formatting	509
Creating custom graphs	510
ASP basics for reports with graphics	510
Filtering data for graphs	510
Formatting graphics	511
Loading a custom report into InForm application	514
Creating an XML file for custom reports	514

About this guide

In this preface

Overview of this guide.....	xiv
Documentation	xv
If you need assistance.....	xviii

Overview of this 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.

Audience

This guide is for trial designers, developers, and administrators such as:

- Clinical Data Managers (CDMs) and other trial design personnel who:
 - Design and implement electronic Case Record Forms (CRFs).
 - Specify automated data edit checks.
 - Manage libraries of trial definition components such as forms and edit checks.
 - Design electronic trial documentation, including protocols and CRF help.
- Application Engineers (AEs) and other trial development personnel who:
 - Develop and apply automated edit checks.
 - Design and implement custom databases for downloading trial data.
 - Develop custom reports.
 - Implement electronic trial documentation.
- Help desk and other administrative personnel who:
 - Set up and maintain users, sites, rights, and groups.
 - Set trial-wide configuration options.

Documentation

The product documentation is available from the following locations:

- **Oracle Software Delivery Cloud** (<https://edelivery.oracle.com>)—The complete documentation set.
- **My Oracle Support** (<https://support.oracle.com>)—*Release Notes* and *Known Issues*.
- **Oracle Technology Network** (<http://www.oracle.com/technetwork/documentation>)—The most current documentation set, excluding the *Release Notes* and *Known Issues*.

All documents may not be updated for every InForm release. Therefore, the version numbers for the documents in a release may differ.

Document	Description
<i>Release Notes</i>	The <i>Release Notes</i> document describes enhancements introduced and problems fixed in the current release, upgrade instructions, and other late-breaking information.
<i>Known Issues</i>	The <i>Known Issues</i> document provides detailed information about the known issues in this release, along with workarounds, if available.
<i>Secure Configuration Guide</i>	The <i>Secure Configuration Guide</i> provides an overview of the security features provided with the Oracle® Health Sciences InForm application, including details about the general principles of application security, and how to install, configure, and use the InForm application securely.
<i>Installation Guide</i>	The <i>Installation Guide</i> describes how to install the software and configure the environment for the InForm application and Cognos software.
<i>Setting Up a Trial with InForm Architect and MedML Guide</i>	The <i>Setting Up a Trial with InForm Architect and MedML Guide</i> describes how to design and implement trials in the InForm application using the InForm Architect application.
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 information is available from the InForm Architect user interface.
<i>Step by Step for CRCs and CRAs Guide</i>	The <i>Step by Step for CRCs and CRAs Guide</i> describes how to use the InForm application to: <ul style="list-style-type: none"> • Screen and enroll patients. • Enter, update, and monitor clinical data. • Enter and respond to queries. • Run trial management reports and clinical data listings.
Online Help	The online Help describes how to use and administer the InForm application. This information is available from the InForm user interface.

Document	Description
<i>Reporting and Analysis Guide</i>	The <i>Reporting and Analysis Guide</i> 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 trial management and clinical data packages available for Reporting and Analysis. It also provides detailed descriptions of each standard report that is included with your installation.
<i>Reporting Database Schema Guide</i>	The <i>Reporting Database Schema Guide</i> describes the Reporting and Analysis database schema.
<i>Portal Administration Guide</i>	The <i>Portal Administration Guide</i> provides step-by-step instructions for configuring and managing the InForm Portal application.
<i>InForm Utilities Guide</i>	<p>The <i>InForm Utilities Guide</i> provides information about and step-by-step instructions for using the following utilities:</p> <ul style="list-style-type: none"> • PFConsole utility • MedML Installer utility • InForm Data Import • InForm Data Export • InForm Performance Monitor utility • InForm Report Folder Maintenance utility
MedML Installer utility online Help	<p>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 trial components into the InForm database.</p> <p>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.</p> <p>This information is available from the utility user interface.</p>
InForm Data Import online Help	<p>The InForm Data Import online Help provides information about, and step-by-step instructions for using the InForm Data Import, which is used to import data into the InForm application.</p> <p>This information is available from the utility user interface.</p>

Document	Description
InForm Data Export online Help	<p>The InForm Data Export online Help provides information about, and step-by-step instructions for using the InForm Data Export, which is used to export data from the InForm application to the following output formats:</p> <ul style="list-style-type: none"> • AutoCode. • Customer-defined database (CDD). • Name value pairs. • Oracle Clinical. <p>This information is available from the utility user interface.</p>
<i>Third Party Licenses and Notices</i>	The <i>Third Party Licenses and Notices</i> document includes third party technology that may be included in or distributed with this product.

Documentation accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

If you need assistance

Oracle customers have access to support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info>, or if you are hearing impaired, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs>.

Finding InForm information and patches on My Oracle Support

The latest information about the InForm application is on the Oracle Support self-service website, My Oracle Support. Before you install and use the InForm application, check My Oracle Support for the latest information, including *Release Notes* and *Known Issues*, alerts, white papers, bulletins, and patches.

Creating a My Oracle Support account

You must register at My Oracle Support to obtain a user name and password before you can enter the site.

- 1 Open a browser to <https://support.oracle.com>.
- 2 Click the **Register** link.
- 3 Follow the instructions on the registration page.

Finding information and articles

- 1 Sign in to My Oracle Support at <https://support.oracle.com>.
- 2 If you know the ID number of the article you need, enter the number in the text box at the top right of any page, and then click the magnifying glass icon or press **Enter**.
- 3 To search the knowledge base, click the **Knowledge** tab, and then use the options on the page to search by:
 - Product name or family.
 - Keywords or exact terms.

Finding patches

You can search for patches by patch ID or number, product, or family.

- 1 Sign in to My Oracle Support at <https://support.oracle.com>.
- 2 Click the **Patches & Updates** tab.
- 3 Enter your search criteria and click **Search**.
- 4 Click the patch ID number.

The system displays details about the patch. You can view the Read Me file before downloading the patch.

- 5 Click **Download**, and then follow the instructions on the screen to download, save, and install the patch files.

Finding Oracle documentation

The Oracle website contains links to Oracle user and reference documentation. You can view or download a single document or an entire product library.

Finding Oracle Health Sciences documentation

For Oracle Health Sciences applications, go to the Oracle Health Sciences Documentation page at <http://www.oracle.com/technetwork/documentation/hsgbu-clinical-407519.html>.

Note: Always check the Oracle Health Sciences Documentation page to ensure you have the most up-to-date documentation.

Finding other Oracle documentation

- 1 Do one of the following:
 - Go to <http://www.oracle.com/technology/documentation/index.html>.
 - Go to <http://www.oracle.com>, point to the **Support** tab, and then click **Product Documentation**.
- 2 Scroll to the product you need, and click the link.

Finding prerequisite software for Oracle Health Sciences applications

Prerequisite software for Oracle Health Sciences applications is available from the following locations:

- Download the latest major or minor release from the Oracle Software Delivery Cloud (<https://edelivery.oracle.com/>).

For information on the credentials that are required for authorized downloads, click **FAQs** on the main page of the Oracle Software Delivery Cloud portal.

- Download subsequent patch sets and patches from My Oracle Support (<https://support.oracle.com>).

To find patch sets or patches, select the **Patches & Updates** tab.

If a previous version of prerequisite software is no longer available on the Oracle Software Delivery Cloud, log a software media request Service Request (SR). Previous versions of prerequisite software are archived and can usually be downloaded. After you open an SR, you can check its status:

- US customers: Call 1-800-223-1711.
- Outside the US: Check www.oracle.com/us/support/contact/index.html for your local Oracle Support phone number.

For more information on logging a media request SR, go to My Oracle Support for Document 1071023.1: Requesting Physical Shipment or Download URL for Software Media (<https://support.oracle.com/epmos/faces/DocumentDisplay?id=1071023.1>).

CHAPTER 1

Introduction to the InForm Architect application

In this chapter

InForm Architect application features	2
The InForm Architect application and MedML.....	3
About trial components.....	4
Starting with the Base trial.....	6
Planning a trial.....	8

InForm Architect application features

The InForm Architect application offers features that support setting up new trials and maintaining existing trials.

Setting up new trials

The InForm Architect application provides:

- Graphical editors for creating visits, designing and creating forms, and applying edit checks and calculations to form items.
- Automatic XML generation and loading.
- Graphical interface for trial administration activities.
- The ability to generate mappings in several formats to external data entities.
- Support for exporting data to an autocode utility and reimporting coded data to a trial.

Enhancing existing trials

The InForm Architect application provides:

- The ability to attach to running trials and regenerate their original XML.
- The ability to add new forms, items, and controls to the existing trial structure.
- User, site, rights, and group maintenance.

Debugging

The InForm Architect application provides:

- Development and test environments for edit checks and calculations.
- Dynamic form previews.

The InForm Architect application and MedML

The InForm application uses MedML tags as the means of defining how a trial definition is stored in the trial database. The InForm Architect application provides a simple, graphical way to define trial components. Every time you use the InForm Architect application to define a trial component, the software automatically generates the underlying MedML tags.

As you design and develop a trial, you save the MedML tags that the InForm Architect application generates as XML files. Optionally, you can load these files into the trial database so you can test them in a working InForm application test trial. When you have fully developed and tested a trial, you must load the final generated XML files into your production trial database.

The MedML Installer utility provided with the InForm application enables you to load the trial definition XML files into the database. For more information, see the *InForm Utilities Guide*.

Although the InForm Architect application can attach to a running trial, it is intended for use in a development and testing environment, not in production. Specifically, when you design a trial with the InForm Architect application and save the component definitions, you cannot use the trial database generated by the InForm application as the production database. You must reinstall the XML files generated by InForm on the production server by running the dbsetup script or the MedML Installer utility, as described in *Installation and Configuration Guide*.

About trial components

A trial definition consists of a set of components. This section describes the types of components that make up the definition of a trial.

Trial building blocks

The components of a trial have a hierarchical relationship with the trial at the top level and the controls on individual forms at the bottom. The building blocks that make up the lower-level components are reusable in higher-level components. Thus, the same form can appear in multiple visits, the same item can appear in multiple forms, and the same controls can appear in multiple items.

The major building blocks in a trial are as follows:

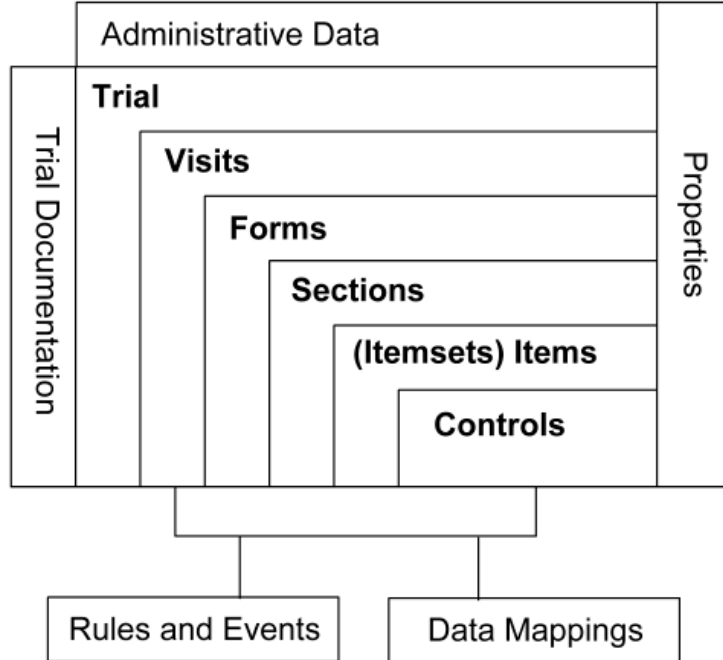
- A *trial* consists of a set of *visits*.
- Each visit consists of a set of *forms*, most commonly CRFs.
- Forms consist of *sections*.
- Each section has an *itemset* or one or more *items*. An itemset is made up of multiple items.
- Each item has one or more *controls*.

Each of these components is defined with a set of *properties*, which specify the characteristics and behavior of the component.

In addition to the building-block components of a trial, a trial definition is associated with the following related components:

- **Administrative data** specifies the users, sites, groups, and rights in a trial.
- **Trial documentation** provides online information about a trial protocol and details about CRF items.
- **Rules and events**, which work in the context of one or more specific items, enable validation or calculation of data items and automatic generation of queries.
- **Data mappings** specify mappings between the items in a trial database and any of the following destinations:
 - Customer-Defined Database (CDD).
 - Clintrial database.
 - Import file formatted for upload to the Oracle Clinical tool.
 - Calculated control designed as the destination of a value supplied by an autocode utility.

The following figure illustrates the building-block components and other components of a trial definition.



Starting with the Base trial

The InForm application is delivered with a collection of predefined, required components used in all trials. These components make up the Base trial. When you install the InForm application and the InForm Architect application, the installation includes the loading of the Base trial files. Information contained in the Base trial includes:

- System resources, including images and HTML files used throughout the InForm application.
- System configuration settings.
- Predefined rights and default rights groups.
- Default query and signature groups.
- Predefined administrative users with rights to perform user administration tasks.
- Default numeric sequences for screening, enrollment, and randomization numbers.
- System form definitions (for example, definitions of Comment, Data Value(s), Query, Audit Trail, and administrative screens).
- InForm application online Help.
- InForm application reports.

When you install the InForm application, the sample trial scripts provided with the installation load the Base trial files along with the files that define the sample trials provided with the InForm application. The Base trial files are the foundation on which the rest of the trial is built.

Customizing Base trial data

Most Base trial data **must not** be modified. To operate correctly, the InForm application depends upon the system settings and component definitions specified in Base trial files. The few exceptions to this rule are discussed in *Customizing Predefined Lists* (on page 411).

Learning with sample trials

Along with the Base trial definitions, the InForm application installation enables you to load several sample trials, each containing a complete set of forms for a hypothetical trial. You can use the sample trials for training, familiarization with the InForm application, and for performing installation qualification.

The sample trials provided with the InForm application are:

- **PFST**—Depression trial with trial components that illustrate InForm application features such as dynamic forms and visits, associated forms, and repeating forms. To install this trial automatically, click **Start > Programs > Oracle Health Science > InForm 4.6.5 > Install Sample Trial for Oracle**.
- **PFST45**—Depression trial with the product features of PFST45 and sample clinical data. To install this trial automatically, click **Start > Programs > Oracle Health Science > InForm 4.6.5 > Install Sample Trial 4.5 for Oracle**.

PFST45 includes a set of clinical data that you can load and use to test the reports delivered with the InForm Reporting and Analysis module and to develop and test custom reports. This data is available as an Oracle database extract. For information about how to load the clinical data into the PFST45 trial database, see *Installation and Configuration Guide*.

This trial is recommended as the best sample to investigate initially. For information about how to load the clinical data into the PFST45 trial database, see *Installation and Configuration Guide*.

When you are ready to load the components for a true trial, you can remove any sample trials by using the pfadmin utility, as described in *Installation and Configuration Guide*.

Planning a trial

This chapter describes strategies you can use when planning to create a trial definition with MedML. The section covers the planning and design of:

- Forms
- Edit checks (rules)
- Electronic CRF help
- Data mappings
- User workflows

Planning forms

When you know what data must be collected in a trial, you can plan how to present it on CRF forms.

Using annotated CRFs as a design tool

The InForm Architect application features an option to annotate case report forms. With the click of an icon, the CRF is annotated with control reference name definitions, control data type and data format definitions, element definitions, date ranges, and data mappings.

Choosing the right controls

Each data entry control in MedML is best suited to capturing a specific type of data. Keep the following recommendations in mind when designing data entry items:

Data entry control	Best used for this type of data
Pulldown list	A finite group of known answers, from which you want the user to select only one answer. Use a pulldown list control rather than a set of radio buttons when conserving space is important; for example, when the list of possible answers is longer than four items.
Radio buttons	A finite group of known answers. Use a set of radio buttons when the list of possible answers is short and the answers themselves are also short. Also, use radio buttons to enclose other controls—for example, a set of pulldown lists grouped with a text box control for capturing user-entered Other data.
Check boxes	A finite group of known answers from which the user can select one or more answers. Use check boxes when the list of possible answers is short and you want all options to be visible at the same time. Also, use check boxes to enclose other controls, as you would use radio buttons.
Text box	Descriptive or other data that cannot be captured with predefined labels. Since data entered in text boxes is not statistically useful until further analyzed, try to minimize the use of text boxes.

Forms design tips

This section describes additional form design guidelines and tips to consider.

Data type

When defining controls in which you specify a data type, usually with the `TYPE` property, consider the following:

- How will the data be analyzed after collection? Consider assigning sets of textual answers numeric values and coding them with a numeric `TYPE` if it will be convenient to handle them numerically in the statistical database. For example, you can assign Yes and No answers numeric values in the database by defining Yes and No data controls with a `TYPE` of `INTEGER` and values of 1 and 2.
- Will individual controls be included in compound controls such as lists or sets of radio buttons? All components of a compound control must have the same `TYPE` property.
- Will the submitted data be transmitted to a Customer-Defined Database? All components of multiple-selection controls (selections in a list box control or a set of check box controls) must be mapped to a target column with the `STRING` type.

Form length

To maximize system performance, keep forms short. Consider breaking longer forms into separate Form components. There is no predefined maximum number of items allowed on a form, but a good rule of thumb is to consider breaking the form into separate components if it contains about 20 items.

Known value sets

When the possible answers to a question belong to a known set of values, always use lists, radio button, or check box selection controls rather than having users enter the values in text boxes.

List size

Keep lists as short as possible, and attempt to break very long lists down into shorter ones. As a rule of thumb, a set of radio buttons should have four or fewer options. If you have more, consider using a pull-down list or breaking the control down further.

Nesting

MedML supports nesting controls within each other; for example, you can create an item that consists of a set of radio buttons, each of which is a pull-down list, and the last of which is a text control for entering additional descriptive information. The pull-down lists and text control are nested within the radio button control. MedML supports a maximum of four levels of nesting. To minimize complexity and assist performance, attempt to minimize the use of controls within controls, and never exceed the maximum number of nesting levels.

Edit checks

MedML enables you to create automated edit checks on CRF items. In the InForm application and the InForm Architect application, these checks are known as rules. You can create multiple rules on a single item, and you can create rules and calculations in which the outcome of the rule or calculation depends on the value of one or more data items on the same or other forms. When you design rules, bear in mind that each of the following choices can affect performance negatively:

- Multiple rules on an item.
- Many rules on a form.
- Multiple dependencies on items in the same or other forms.

Scrolling

For the convenience of the user, attempt to minimize scrolling by:

- Making text boxes wide enough to accommodate the longest data you expect users to provide.
- Making forms as short as possible.
- Placing the selections you expect users to choose most frequently at the top of pull-down lists.

Developing forms libraries

As you develop forms for multiple trials, you will find that some trial component definitions, including some whole forms, are common to many trials, and their definitions can be reused with only slight alterations. It is a good practice to maintain libraries of trial components so that you can reuse them from trial to trial.

Planning edit checks

By defining rules, you can validate the data entered for specific items on a form. As new data is entered in the item, or when existing data is changed, the rule attached to the item is run against the data. Depending on the outcome, different events can occur:

- The event can generate a new query.
- The event can close an answered query.
- The event can run one or more execution plans that:
 - Send e-mail.
 - Send an entry to a log file.

When you plan the edit checks for your trial, consider how you might use the following InForm application rule capabilities. For information on how to implement edit checks, see *Defining rules* (on page 279).

Types of rules

You can use rules to perform:

- **Validation checks.** If the value of an item fails a validation check, the InForm application issues an automatic query. Queries can be issued in Candidate (invisible to the site) or Opened state. In deciding what the initial state of queries should be, consider whether you want sponsor personnel to review automated queries before the site sees them.
- **Calculations** that determine the value of an item based on the value of one or more related items.
- **Unit conversions.**

Client-side and server-side edit checks

The InForm application enables client-side or server-side edit checks:

- **Client-side edit checks** are performed when a user exits a control, before submission of data. You can use this facility to check for conformance with minimum and maximum value ranges or to ensure that all parts of a multiple-control item (for example, a date field where all parts of the date are required) are filled in.
- **Server-side edit checks** are performed when a user submits data. Any edit checks that involve comparisons, calculations, checks for the existence of entered data, or specific data properties other than minimum or maximum values require processing on the server.

Generic and item-specific rules

You can design generic rules or rules targeted to a specific form and item:

- Generic rule design enables you to use the same rule script for multiple items on multiple forms. Additionally, you can pass different sets of arguments to parameters in a rule script for each item with which the rule is associated. This type of rule design is especially useful for tests such as range checks, which may be suited to many types of items in many contexts.
- For more complex checks and calculations, you can hard-code values and references to controls in a rule script and apply the rule to one specific item, form, and visit occurrence.

Dependencies

Edit checks can involve multiple items. For example, you can calculate the value of an item based on the value of one or more other items. You can check whether the value of one item has the appropriate relationship to the value of another; for example, whether the date of the second visit is at least two weeks later than the date of the first. All of the items whose value a rule looks at are called its *dependencies*.

Rule dependencies identify which item receives a query if a rule fails, and which item receives the result of a calculation. Dependency properties also determine whether an item has to have data entered in it for a rule to run. If an item is identified as an applied dependency, it receives a query if a rule fails or a calculation result if the rule is a calculation. If an item is identified as an applied or dependent dependency, it must have data entered before a rule can run. If an item is identified as a trigger dependency, the rule runs whether or not the item has data.

When you plan your trial's edit checks, consider which items are related to each other in such a way that you could identify the items as rule dependencies and use a rule to validate the relationship. Also, consider whether you can use rules to calculate the value of items.

Online help on edit checks

You can provide online Help that describes the edit checks being performed on an item. This help text appears along with CRF Help text when a user opens the Document window or links from an item to CRF help. For information on how to create edit check help, see *Specifying rule properties* (on page 294).

Planning CRF help

If the trial documentation for your trial includes detailed help about how to collect and enter CRF data, consider the following options:

- You can implement CRF help as an online document that is displayed to InForm application users in the Document window.

This level of implementation involves creating CRF help as a set of HTML documents, creating a table of contents for the set, and creating links from the table of contents to the individual documents. Additionally, you must set up a documentation definition XML file that loads the CRF help and its structure into the database.

- You can implement form-level or item-level links from CRFs to specific locations in online CRF help.

This level of implementation involves including HELPLINK components in the definitions of forms or items. These components point to specific files in the CRF help. If you define links to CRF item definitions, you must also create book marks in the CRF help and reference the bookmarks in the HELPLINK definitions.

- You can enable users to update CRF help by creating frequently-asked question (FAQ) entries that provide additional clarification about entering or updating CRF data.

This level of implementation involves creating FAQ placeholders in the CRF help, defining the CRF help document as one that contains FAQs, and assigning users the rights to maintain FAQs.

Note: Unlike other types of trial components, CRF help and electronic protocol documents require that you manually create the XML files to load the help and protocol definitions. The InForm Architect application does not support the creation CRF help and electronic protocols at this time.

Planning data mappings

The InForm application allows you to map the data stored in an InForm application trial database to external systems where you can use third-party analytical tools. The InForm application supports mapping to the following external data entities:

- Customer-Defined Database
- Clintrial software database
- Oracle Clinical software

Additionally, you can create mappings between data points in an InForm application trial so that you can export data to be processed by an external coding tool and then reimport it after coding to the target controls specified in the mappings.

Planning Customer-Defined Database mappings

A *Customer-Defined Database* (CDD) is a database that gives sponsor personnel access to the data being collected in a trial while the trial is ongoing and when it is completed. Sponsors can review the data as needed to spot results that might require protocol adjustments or warrant early statistical analysis.

The structure of a CDD reflects the types of data analysis a sponsor wants to perform. As such, it does not match the structure of the InForm application trial database. To deliver trial data in the sponsor-specified database structure, the InForm application enables you to create a set of CDD mapping definitions that specify:

- Where each mapped data point comes from in the source trial.
- Where each mapped data point goes in the CDD.
- How the data is organized in the CDD.
- Optional, supplemental text about the design of the components in the CDD definition.
- Optional label text that is transferred to the CDD along with the data values.

When you think about how to define CDD mappings, start with the database specifications developed by the sponsor. Consider how you want to be able to retrieve the data from the CDD, including:

- Do you want to map multiple data points to the same CDD table column?
- Do you want to repeat a particular data point in each row of the table?
- Do you want to combine data from different visits, forms, or sections in the same CDD table?
- How do you want date and time data to be accessed—in a single string, in two strings with date and time segments separated, or in six strings with each component separated?
- Do you want individual checkboxes or radio buttons to be in separate columns, or do you want a single column to hold the data for the checkbox or radio group?

Planning user setup

When planning the roles of different users in the InForm application, you should carefully examine the functions that those users currently perform within your organization. For example, who has supervisory authority? Who can authorize changes in the trial protocol or in collected data? Who enters the data? How is the workload distributed? You will probably want to define most of this functionality the same way in the InForm application.

Also, try to be aware of what rights and privileges depend on others. For example, if particular users have managerial responsibility over other users, does that also imply that they have signature authority over all changes that their subordinates might need to make?

Several default groups of rights that you might commonly use are already predefined in the InForm application. You can use these rights as a basis for examining the responsibilities and workflow within your organization, and add additional rights as needed.

Planning reports

The Reporting and Analysis module makes clinical and trial management data available in a set of predefined reports and enables report designers to create unlimited custom reports. When designing your trial, keep in mind that both data and labels are available as objects that can be included in reports.

Therefore, make sure that the names you give to trial objects are meaningful and are the ones you want to appear in reports. For example, consider how you want your reports to show user, site, and group names. Be aware that design notes are available as reporting objects.

The value of the Itemset Column Header property of an item serves as the default short name for the field in the clinical reporting model.

Note: You should ensure that Itemset Column Header definitions are unique within a trial. If the Cognos reporting system finds duplicate Itemset Column Header values, it combines the items in reports.

For additional reporting considerations, see *Reports* (on page 378).

CHAPTER 2

Quick tour

In this chapter

Overview	16
Starting and stopping the InForm Architect application.....	17
InForm Architect application window	18
Trial Objects window	19
Properties window	21
Output window	23
Design Workspace.....	24
Toolbars.....	39
Customizing the application workspace.....	45
Using editing and navigation features.....	49
Deleting trial components	56

Overview

This chapter gives you a quick tour of the InForm Architect application. It introduces you to the main application window, the workspace, and toolbars that you will use in setting up and managing trial components. Additionally, it describes how to start and stop the InForm Architect application.

If you are unfamiliar with how to work with applications that have multiple windows and toolbars, be sure to read the following sections:

- *Design Workspace* (on page 24)
- *Toolbars* (on page 39)
- *Customizing the application workspace* (on page 45)
- *Docking and undocking* (on page 47)

Starting and stopping the InForm Architect application

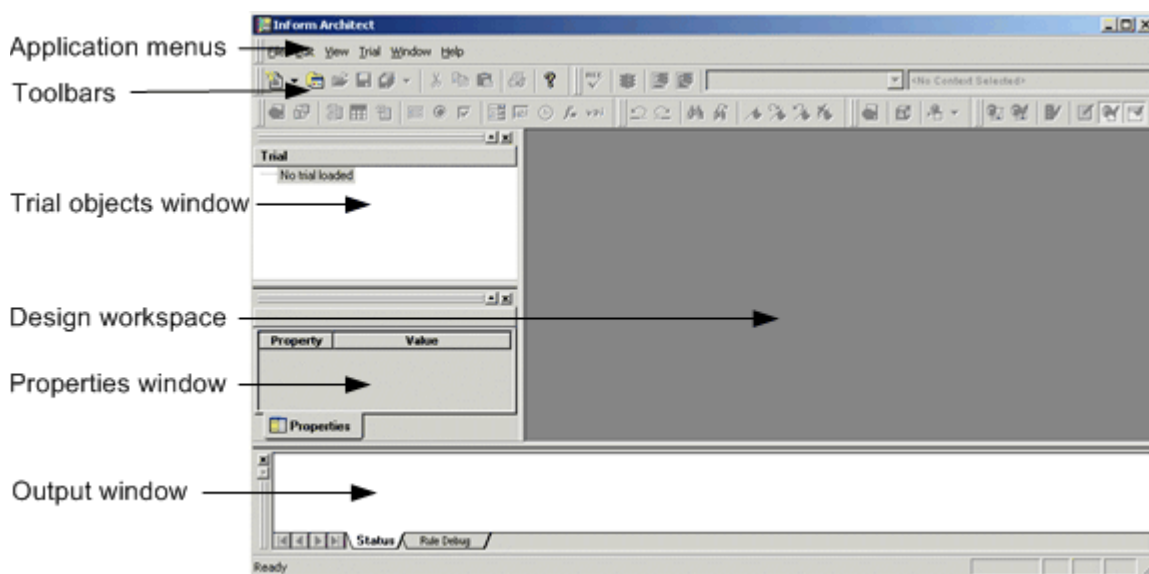
The InForm Architect application works with the InForm application server and database and attaches to a specific trial. You can create a trial with the InForm Architect application or you can work with an existing trial.

- To start the InForm Architect application—Click **Start > Programs > Oracle Health Science > Architect > Architect**.
- To stop the InForm Architect application—Select **File > Exit**.

Note: In order to start the InForm application service or to open a trial in the InForm Architect application, you must be a local Windows administrator for the machine where you are running the InForm Architect application.

InForm Architect application window

When you start the InForm Architect application, the application window opens, and you can see the toolbars and features that make up the default display:



- **Application menus**—Initiate frequently performed tasks.
- **Toolbars**—Perform additional functions you are likely to use frequently.
- **Trial Objects window**—Appears initially with the message “No trial loaded” and is a repository of the components defined for a trial.
- **Properties window**—Appears blank initially and is a properties editor for a specific component selected in the Trial Objects window or being worked on in the Design Workspace.
- **Design Workspace**—Appears blank initially and is a work area in which you can open and manipulate windows used for creating and defining trial components.
- **Output window**—Shows the status and results of any batch processing that you do; for example, opening a trial, saving a trial, inserting MedML into a trial, or testing rules against test data.

The panes of the application window are blank until you create or load a trial, as described in *Setting up a Trial* (on page 57). As you work on a trial, these panes make up a workspace that you can customize for your convenience. Each of these features is described in the sections that follow.

Trial Objects window

The Trial Objects window appears by default in the upper left corner of the application window. This window displays all components that have been defined for a specific trial, organized in a tree structure by component type. As you define new trial components, the Trial Objects window is updated, and the new components become available for use throughout the trial. Additionally, the Trial Objects window provides access to the trial administration screens that enable you to set up and maintain user, rights, group, and site definitions.

The Trial Objects window has the following features:

- When you create a new trial or load an existing trial, all trial components are displayed in the Trial Objects window. This repository includes the predefined components common to all trials that are loaded into the database during the base installation as well as any components that you define.
- Each type of trial component is represented by an icon and a tree control that you can expand or contract.

To display all components defined for a component type, click the **plus** icon (⊕).

To hide all individual components, click the **minus** icon (⊖).

- The components displayed in the Trial Objects window are reusable throughout the trial you are working on.

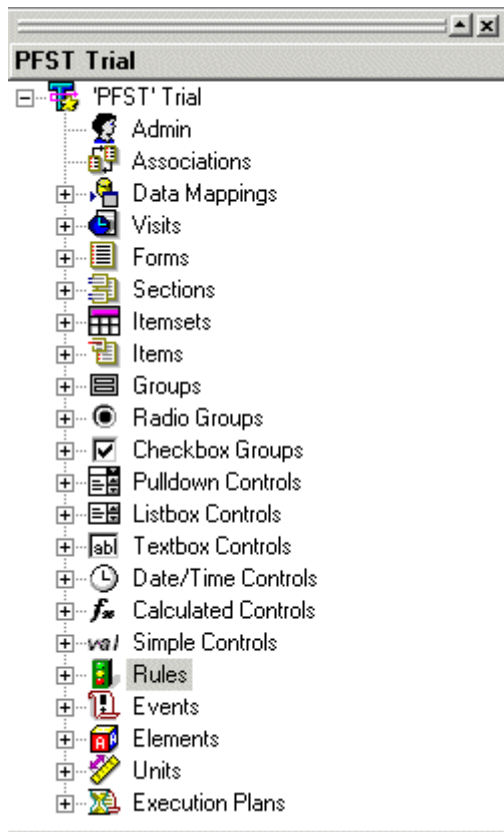
To add many types of components to the trial structures you are defining, you can drag them from the Trial Objects window onto the appropriate window in the Design Workspace.

- Use the Trial Objects window to open previously or partially defined components in the Design Workspace for editing.

To do this, double-click the component in the Trial Objects window. You can open forms, elements, units, data mappings, rules, events, and execution plans in this way.

- The Admin tree in the Trial Objects window provides access to the administration components you can maintain. When you double-click the Admin node, the User View administration screen opens in a browser view in the Design Workspace, and you can perform administration activities as you would within InForm application.

The following figure illustrates the Trial Objects window:



Properties window

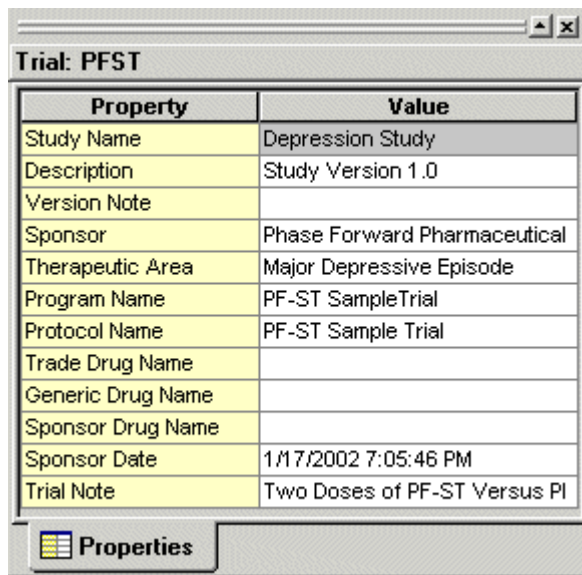
The Properties window is an editor in which you define or update the properties of a specific trial component. This window displays the properties of the component that is selected in the active window in the Design Workspace or of the component that is selected in the Trial Objects window. The Properties window appears by default in the lower left side of the application window. It consists of the Properties tab, which is always visible, and the Mappings tab, which is visible when you select a mappable control in the Form window.

Properties tab

The Properties tab is the default view of the Properties window. This tab is available for all types of trial components. It consists of a table in which the property name appears in the Property column and the value of the property appears in the Value column.

To edit the definition of the property, click the appropriate cell in the Value column and type or select the value you want to specify. If a property has a set of known values, when you click the value cell, a pull-down list appears, and you can select a value from the list. For some properties, you must select multiple values. In these cases, a list of checkboxes appears when you click the property's Value cell, and you can select each value that applies.

The following figure illustrates the Properties tab of the Properties window.



Property	Value
Study Name	Depression Study
Description	Study Version 1.0
Version Note	
Sponsor	Phase Forward Pharmaceutical
Therapeutic Area	Major Depressive Episode
Program Name	PF-ST SampleTrial
Protocol Name	PF-ST Sample Trial
Trade Drug Name	
Generic Drug Name	
Sponsor Drug Name	
Sponsor Date	1/17/2002 7:05:46 PM
Trial Note	Two Doses of PF-ST Versus PI

Mappings tab

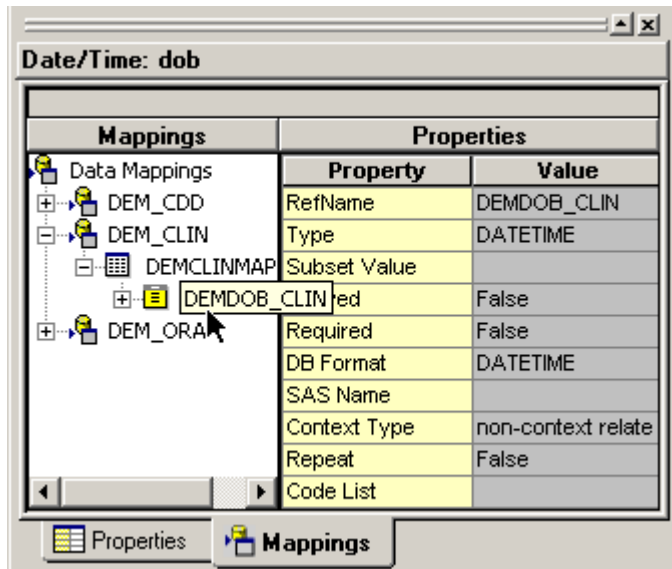
The Mappings tab of the Properties window is visible when a control that is or can be mapped to external systems is selected in the Form editor. This tab shows the data mappings that have been created for the control.

The left pane lists all external systems to which the selected control is mapped. When you expand the mapping nodes, you can see the tables and columns in which the mapping appears. For names that are too long for the visible pane, you can display a tool tip with the complete name by placing the cursor above the name.

To open the Data Mappings window, right-click the mapping node, table, or table column name and select Data Mappings Editor from the pop-up menu. The Data Mappings window opens with the selected node highlighted.

When you click a node in the left pane, the right pane displays the mapping properties of the selected node. This information is read only.

The following figure illustrates the Mappings tab of the Properties window.



Output window

The Output window displays the results of a batch process initiated in the InForm Architect application. This window displays the following tabs:

- **Status**—Displays messages that show the progress of trial creation, opening, and saving operations.
- **Rule Debug**—Displays messages that follow the process of running a rule against test data or selected data from the trial database.

The following figure illustrates the Status tab of the Output window.



Design Workspace

The Design Workspace is the large area to the right of the Trial Objects and Properties windows. This is the area in which you create and update trial components, by working with a set of trial component editors.

The component editors are a group of windows that you open in the Design Workspace to define and update trial components. You use each type of window for editing a different component or group of components:

This window...	Enables you to...
Protocol Editor	Set up the structure of a trial. A trial has one Protocol Editor window, in which you add, order, update, or remove visits and in which you add, order or remove forms.
Form Editor	Define forms and form components, including sections, items, and controls. Each form definition appears in a separate Form editor window, and you can work on multiple form definitions at one time.
Element Editor	Define and update elements, the lowest-level components used in form definitions. The Edit Element window is a table that lists all currently defined elements.
Rule Editor	Define rules (edit checks) and associate them with form components. Rules windows also enable you to create test data for debugging rule scripts. Each rule definition appears in a separate Rule editor window, and you can work on multiple rule definitions at one time.
Execution Plan	Define execution plans that can log an event to the Windows log or send email.
Data Mappings	<p>Define mappings between controls on a form and data mapping tables, columns and control paths. Each data mapping type is represented in a different window. To define mappings, you can:</p> <ul style="list-style-type: none"> • Use pop-up menus that appear when you right-click a node in the Data Mappings window. • Use the pop-up menu that appears when you right-click the Form RefName in the Form window. • Drag and drop controls from a Form window to a Data Mappings window.

Protocol Editor window

The Protocol Editor window enables you to create and update the visit and form structure of a trial. To open the Protocol Editor window, do either of the following:

- Select **Trial > Protocol Editor**.
- Click the Protocol Editor icon  in the Trial toolbar.

This window displays the following tabs: Protocol and MedML.

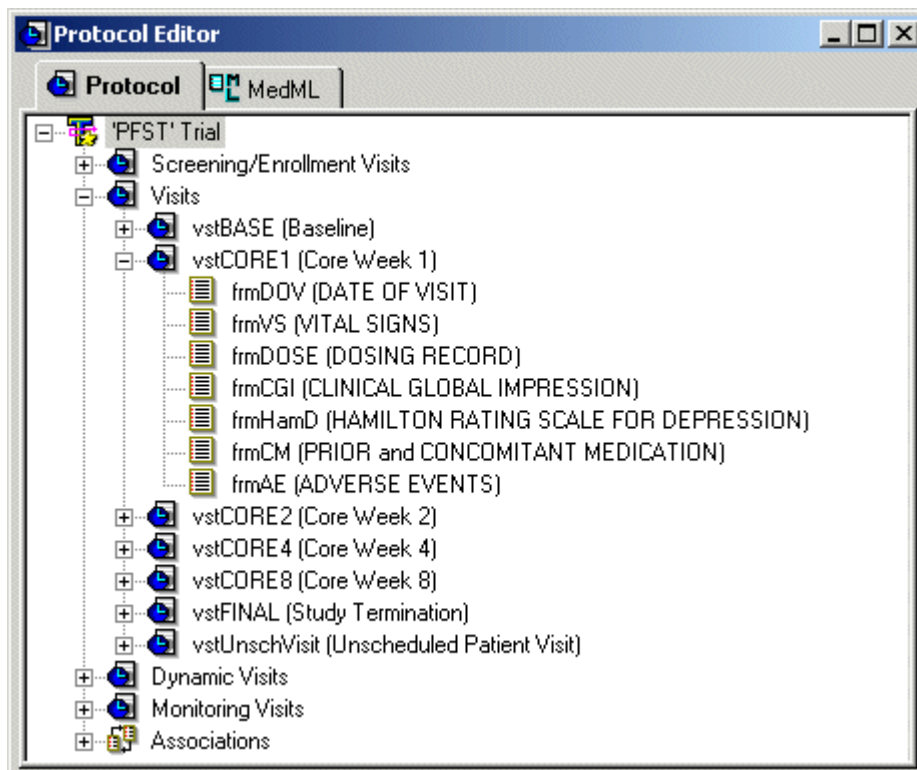
Protocol tab

The Protocol tab is a hierarchical view of the visits and forms in the trial. The trial and its visits are each represented by an icon and a tree control that you can expand or contract.

To display all visits in the trial or all forms in a visit, click the **plus** icon (+).

To hide all forms or visits, click the **minus** icon (-).

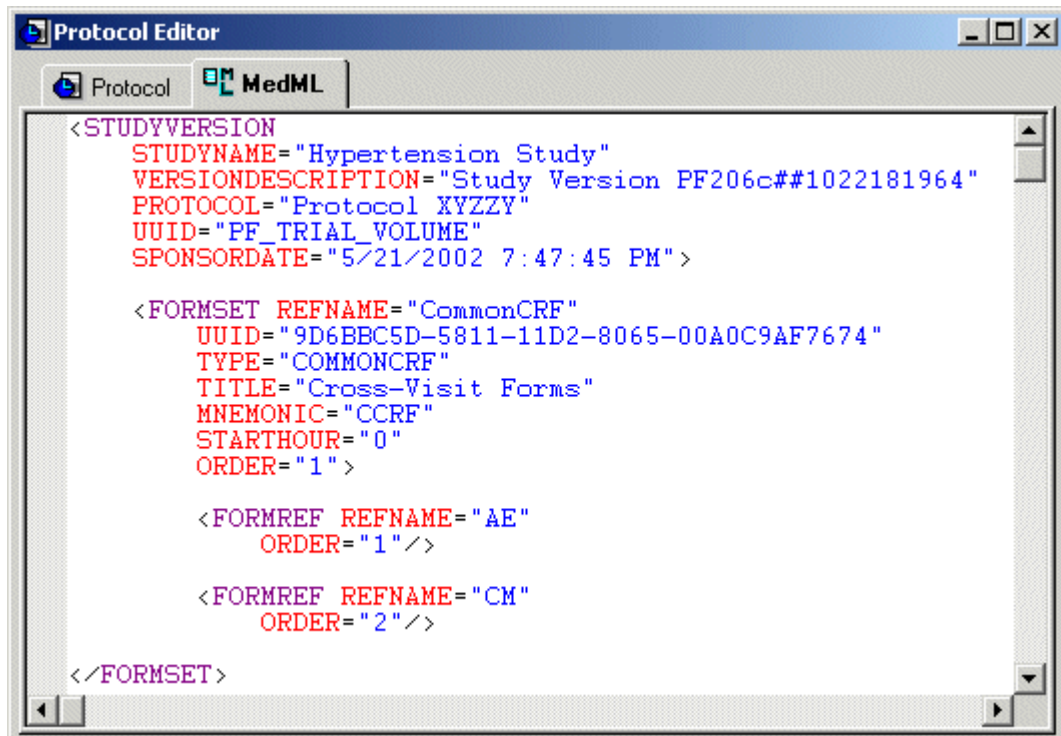
This tab is an editor in which you can create new visits, add existing forms to a visit, reorder visits and forms, redefine the properties of visits, and associate repeating forms. For details on how to perform these activities, see *Defining Visits* (on page 75).



MedML tab

The MedML tab of the Protocol Editor window provides a read-only display of the MedML tags used to define the trial structure. MedML is a standard developed by Oracle and used to exchange clinical data.

To open an online help file containing details about each MedML tag, select **Help > MedML Help**.



```

<STUDYVERSION
  STUDYNAME="Hypertension Study"
  VERSIONDESCRIPTION="Study Version PF206c##1022181964"
  PROTOCOL="Protocol XYZZY"
  UUID="PF_TRIAL_VOLUME"
  SPONSORDATE="5/21/2002 7:47:45 PM">

  <FORMSET REFNAME="CommonCRF"
    UUID="9D6BBC5D-5811-11D2-8065-00A0C9AF7674"
    TYPE="COMMONCRF"
    TITLE="Cross-Visit Forms"
    MNEMONIC="CCRF"
    STARTHOUR="0"
    ORDER="1">

    <FORMREF REFNAME="AE"
      ORDER="1"/>

    <FORMREF REFNAME="CM"
      ORDER="2"/>

  </FORMSET>

```

Form editor window

Form editor windows enable you to create, define, and update forms and their components, as well as use annotated forms for design review. Two tabs are available in Form editor window: Design and MedML.

Design tab of Form window

The Design tab of a Form editor window enables you to see the structure of a form and its components and to see a preview of the way a form will look in the InForm application. This tab has two panes:

- The top pane consists of a hierarchical representation of the structure of a form and its components. Each type of form component is represented by an icon and a tree control that you can expand or contract.

To display all components defined for a component type, click the **plus** icon (⊕). To hide all individual components, click the **minus** icon (⊖).

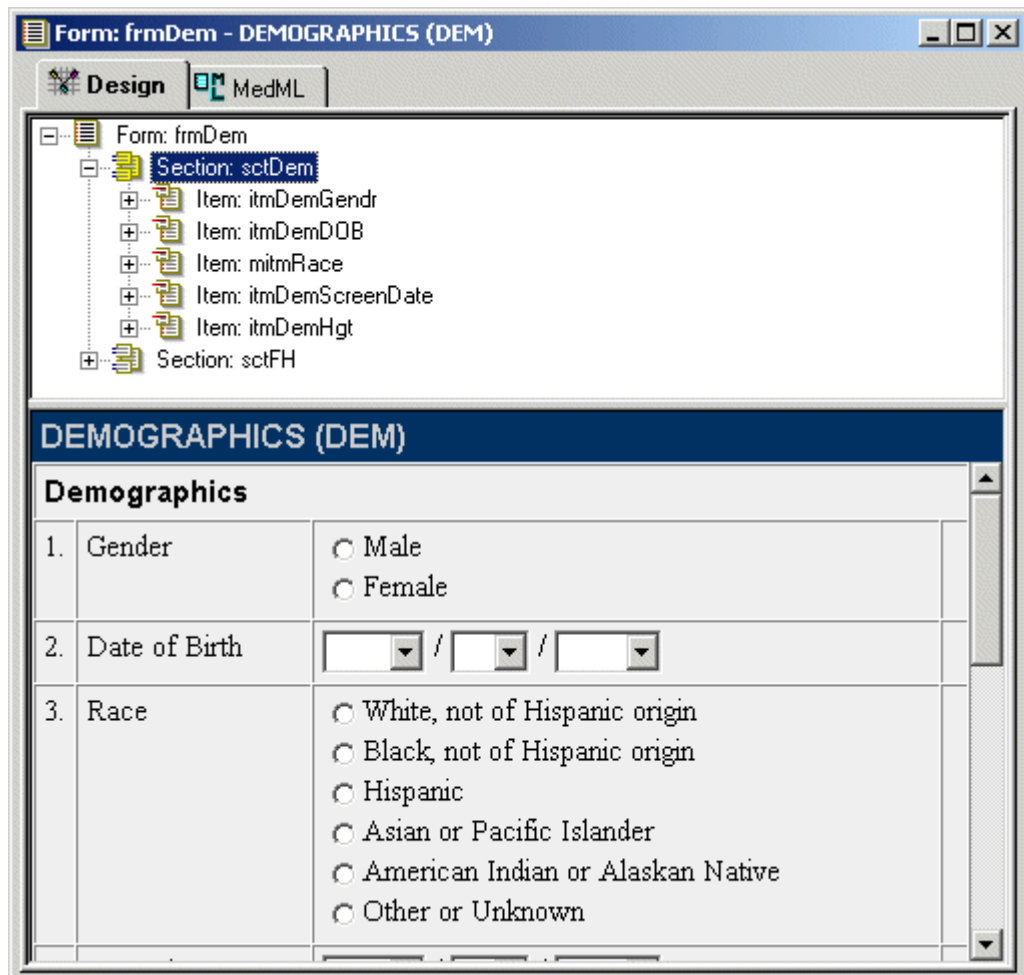
Along with representations of form components, this tab also shows any rule contexts that are attached to each form component.

To display the definition of an attached rule, double-click its icon in the Design tab.

- The bottom pane consists of a preview of the form as it will appear online in the InForm application.

With this tab of the Form editor window, you can:

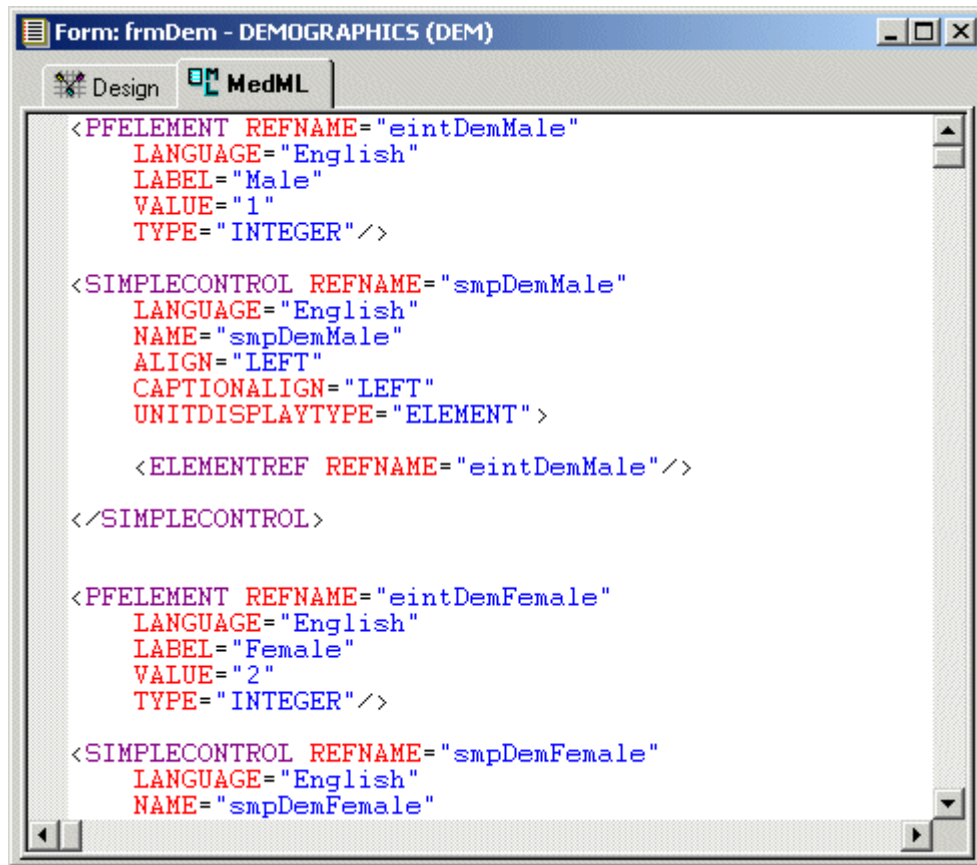
- Create a new form and define its properties.
- Add new or existing components to the form definition and define their properties.
- Reorganize or modify a form and its components.
- Define a column or control path for an existing data mapping.
- Annotate forms with metadata as well as data mappings.



MedML tab of Form window

The MedML tab of the Form editor window provides a read-only display of the MedML tags used to define the Form.

To open an online help file containing detailed information about each MedML tag, select **Help > MedML Help**.



```

Form: frmDem - DEMOGRAPHICS (DEM)
Design MedML
<PFELEMENT REFNAME="eintDemMale"
  LANGUAGE="English"
  LABEL="Male"
  VALUE="1"
  TYPE="INTEGER"/>

<SIMPLECONTROL REFNAME="smpDemMale"
  LANGUAGE="English"
  NAME="smpDemMale"
  ALIGN="LEFT"
  CAPTIONALIGN="LEFT"
  UNITDISPLAYTYPE="ELEMENT" >

  <ELEMENTREF REFNAME="eintDemMale"/>

</SIMPLECONTROL>

<PFELEMENT REFNAME="eintDemFemale"
  LANGUAGE="English"
  LABEL="Female"
  VALUE="2"
  TYPE="INTEGER"/>

<SIMPLECONTROL REFNAME="smpDemFemale"
  LANGUAGE="English"
  NAME="smpDemFemale"
  
```

Dragging and dropping form components

The Form editor window enables you to set up a form definition quickly by dragging form components from the Object Palette toolbar or the Trial Objects window into the Form editor window, and dropping them into the appropriate location in the Form structure. The Object Palette toolbar gives you access to templates for creating **new** form components. The Trial Objects window contains definitions of **existing** form components. For further information, see *Defining Forms* (on page 85).

Cutting, copying, and pasting

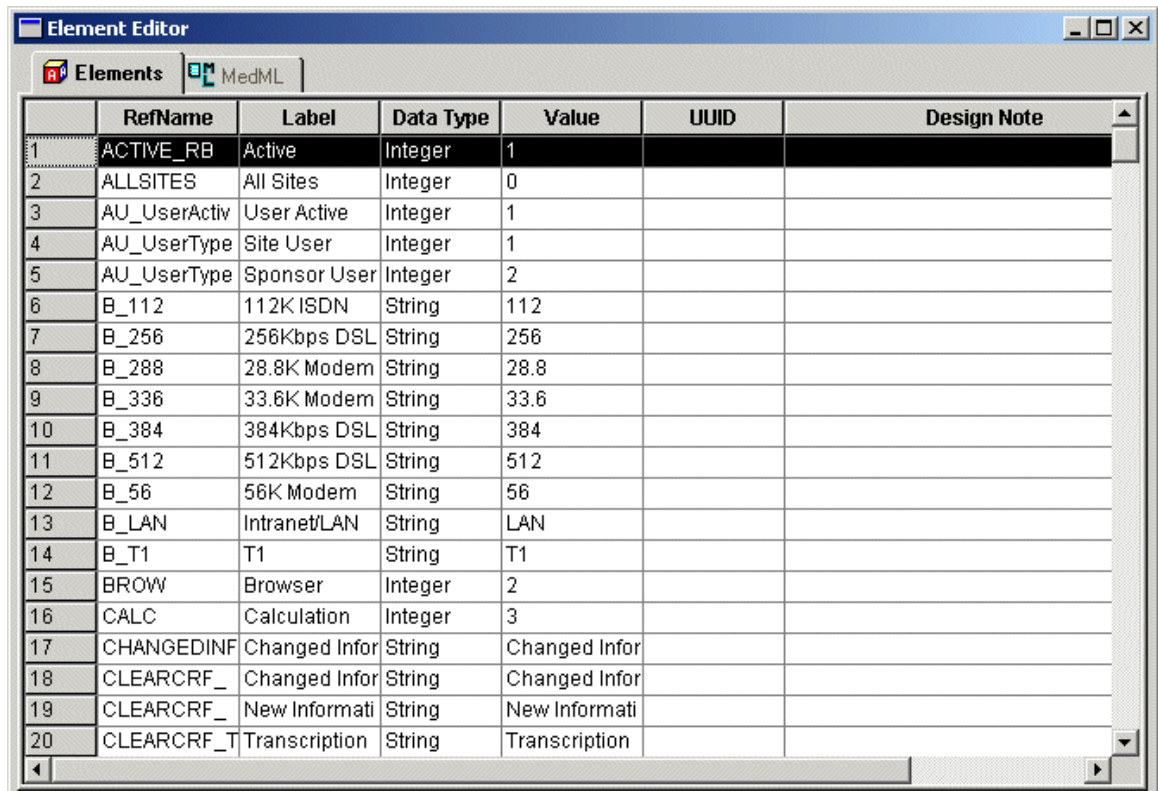
The InForm Architect application enables you to copy a form component from one location to another or to create a new component based on the definition of an existing one. For information, see *Cutting, copying, and pasting* (on page 51).

Element Editor window

The Element Editor window enables you to create and update the definitions of *elements*. *Elements* are the lowest-level components you can use in the definition of a form.

Elements tab

The Elements tab of the Element Editor window displays a table that contains the definitions of all elements available to the current trial. Use this tab to create definitions of new elements or to update the definitions of existing elements.



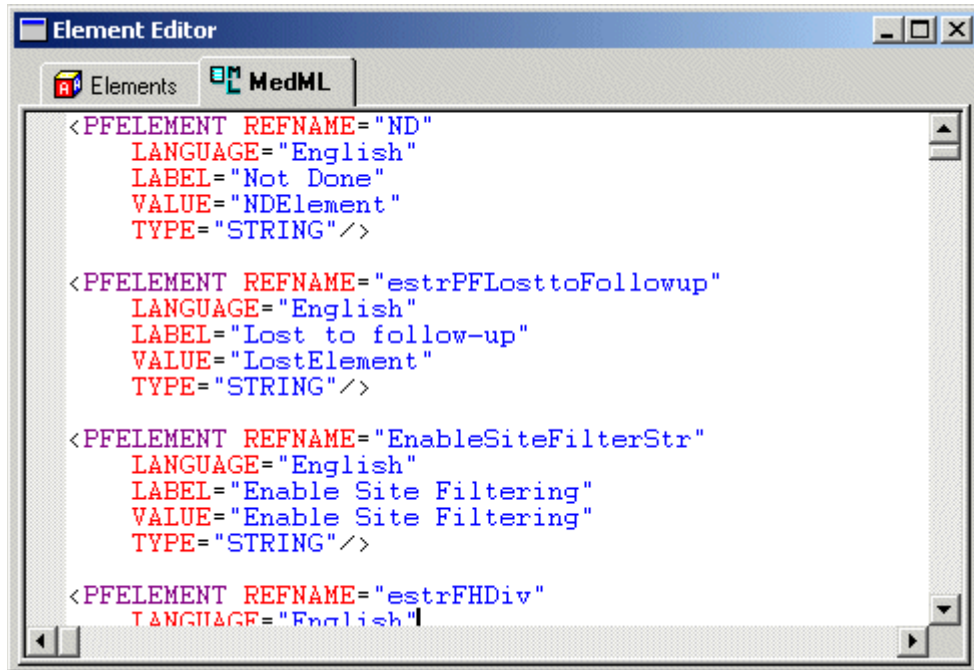
The screenshot shows the 'Element Editor' window with the 'Elements' tab selected. The window title bar includes 'Element Editor' and 'MedML'. The table below lists 20 elements with their respective RefName, Label, Data Type, Value, UUID, and Design Note.

	RefName	Label	Data Type	Value	UUID	Design Note
1	ACTIVE_RB	Active	Integer	1		
2	ALLSITES	All Sites	Integer	0		
3	AU_UserActiv	User Active	Integer	1		
4	AU_UserType	Site User	Integer	1		
5	AU_UserType	Sponsor User	Integer	2		
6	B_112	112K ISDN	String	112		
7	B_256	256Kbps DSL	String	256		
8	B_288	28.8K Modem	String	28.8		
9	B_336	33.6K Modem	String	33.6		
10	B_384	384Kbps DSL	String	384		
11	B_512	512Kbps DSL	String	512		
12	B_56	56K Modem	String	56		
13	B_LAN	Intranet/LAN	String	LAN		
14	B_T1	T1	String	T1		
15	BROW	Browser	Integer	2		
16	CALC	Calculation	Integer	3		
17	CHANGEDINF	Changed Infor	String	Changed Infor		
18	CLEARCRF_	Changed Infor	String	Changed Infor		
19	CLEARCRF_	New Informati	String	New Informati		
20	CLEARCRF_T	Transcription	String	Transcription		

MedML tab

The MedML tab of the Element Editor window provides a read-only display of the MedML tags used to define all of the elements available to the current trial.

To open an online help file containing detailed information about the MedML tags used to define elements, select **Help > MedML Help**.



Rule editor window

Rule editor windows enable you to create and update rule definitions and scripts and to associate rule definitions with form components. The following Rule editor window tabs are available: Rule Profile, Script, Test Cases, and MedML.

Rule Profile tab

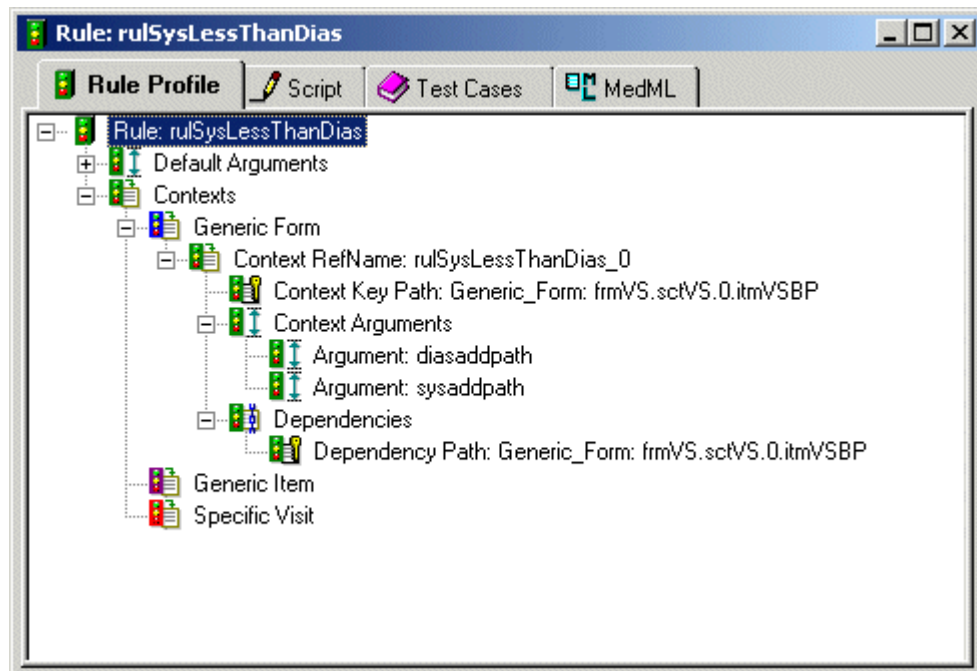
The Rule Profile tab enables you associate a rule with the form items it checks or depends on to perform a test or calculation. This tab is organized as a hierarchical list showing a rule and its contexts, dependencies, and arguments:

- **Contexts**—Defines the item it is evaluating or calculating and all items on whose values the rule script depends. A rule can be associated with different items on different forms and thus have many contexts. The key path of a rule context specifies the visit, form, section, itemset (if applicable), and item against which the rule runs.

There are three types of rule contexts:

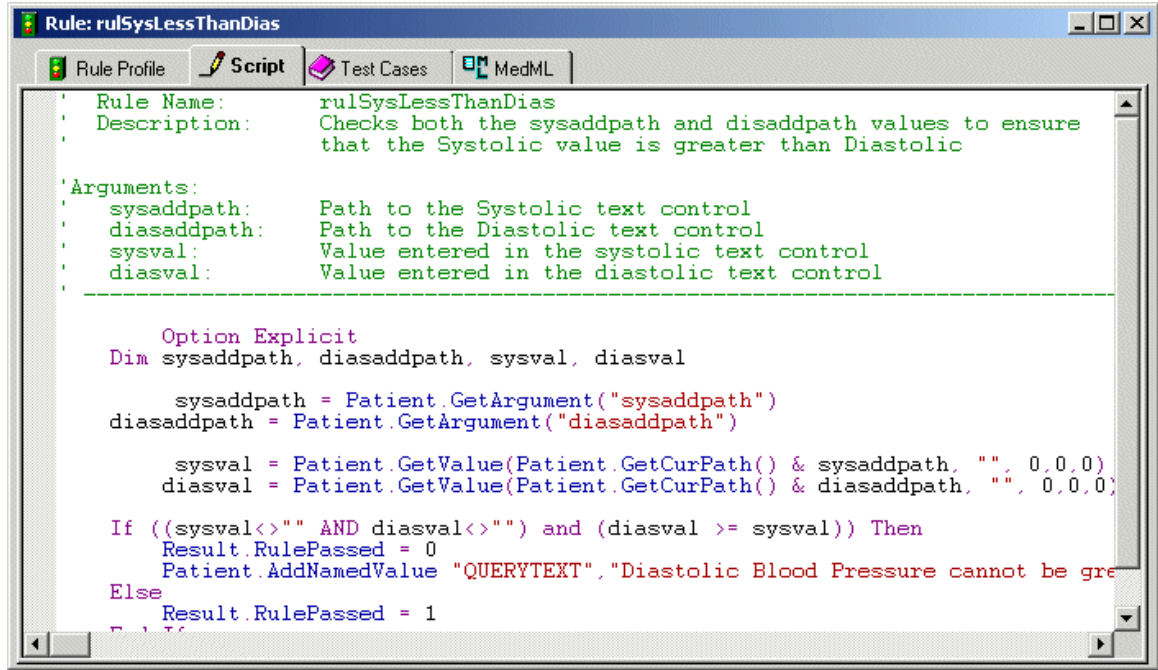
- **Generic Form**—Associates the rule with a form and runs the rule in every instance of the form in the trial. Use this context when you want the same rule to run on every instance of an item on a particular form throughout a trial.

- **Generic Item**—Associates the rule with a particular item and runs the rule with every instance of the item in the trial. Use this context when you want the rule to run on every instance of an item throughout the trial.
- **Specific Visit**—Associates the rule with one item in one visit. Use this context to attach a rule to a single item on a form for a specific visit.
- **Dependencies**—Part of the definition of a rule context is the set of items on which the rule script depends. A dependency consists of a key path that specifies the visit, form, section, itemset (if applicable), and item RefNames of the dependent item. The rule context is also the primary dependency. A rule can also have additional dependencies if its logic uses the values of more than one item.
- **Arguments**—Specify values that are passed to a rule script when it executes. A rule has a set of default arguments. Additionally, each context has its own set of arguments. The values of context-specific arguments override the values of the default arguments.



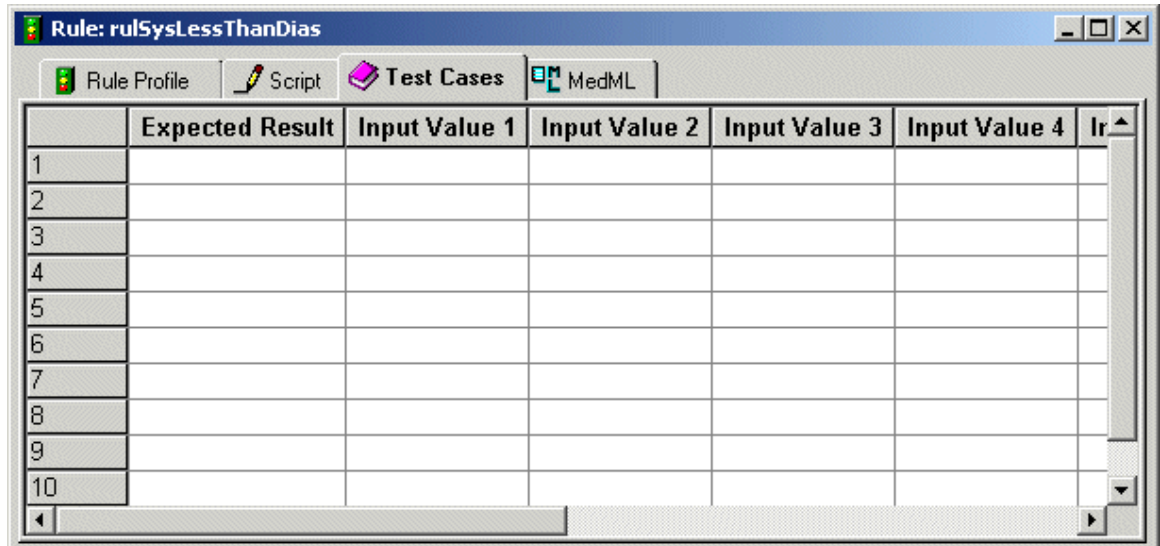
Script tab

The Script tab enables you to enter and edit the text of a rule script written in Visual Basic Script (VB Script). You can enter text directly, or you can cut and paste text from a word processor such as Microsoft Word. When you create a rule script, the MedML tab is updated to include the text of the script.



Test Cases tab

The Test Cases tab enables you to define test data for a rule so that you can run the rule script in test mode. In defining a test case, you specify the expected result and a value for each item that the rule checks. When you define a test case, the InForm Architect application adds it to the MedML definition of the rule. When you run a rule in test mode, the InForm Architect application substitutes the test data values for selected test cases or for each test case in succession and displays the test results in the Output window.

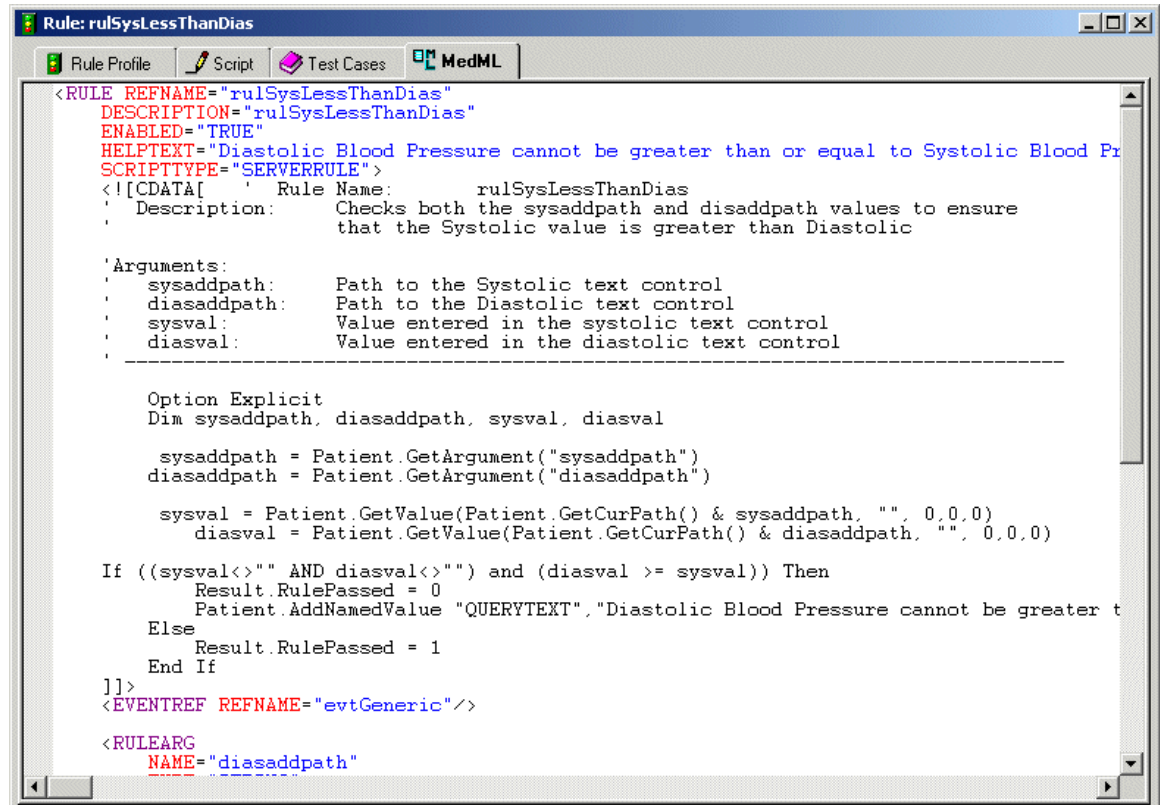


	Expected Result	Input Value 1	Input Value 2	Input Value 3	Input Value 4	Ir
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						

MedML tab

The MedML tab of the Rule editor window provides a read-only display of the MedML tags used to define the rule.

To open an online help file containing detailed information about each MedML tag, select **Help > MedML Help**.



```

Rule: rulSysLessThanDias
Rule Profile | Script | Test Cases | MedML
<RULE REFNAME="rulSysLessThanDias"
DESCRIPTION="rulSysLessThanDias"
ENABLED="TRUE"
HELPTXT="Diastolic Blood Pressure cannot be greater than or equal to Systolic Blood Pr
SCRIPTTYPE="SERVERRULE">
<![CDATA[
    Rule Name:      rulSysLessThanDias
    Description:    Checks both the sysaddpath and diasaddpath values to ensure
                   that the Systolic value is greater than Diastolic

    Arguments:
    sysaddpath:    Path to the Systolic text control
    diasaddpath:   Path to the Diastolic text control
    sysval:        Value entered in the systolic text control
    diasval:       Value entered in the diastolic text control
-----
    Option Explicit
    Dim sysaddpath, diasaddpath, sysval, diasval

    sysaddpath = Patient.GetArgument("sysaddpath")
    diasaddpath = Patient.GetArgument("diasaddpath")

    sysval = Patient.GetValue(Patient.GetCurPath() & sysaddpath, "", 0,0,0)
    diasval = Patient.GetValue(Patient.GetCurPath() & diasaddpath, "", 0,0,0)

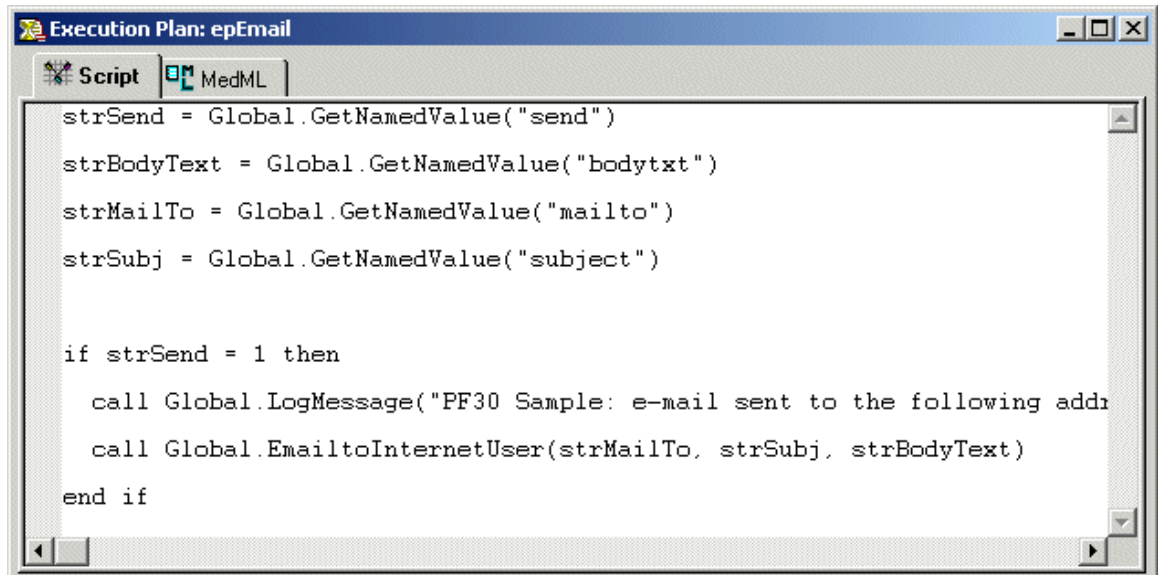
    If ((sysval<>" " AND diasval<>" ") and (diasval >= sysval)) Then
        Result.RulePassed = 0
        Patient.AddNamedValue "QUERYTEXT", "Diastolic Blood Pressure cannot be greater t
    Else
        Result.RulePassed = 1
    End If
]]>
<EVENTREF REFNAME="evtGeneric"/>
<RULEARG
    NAME="diasaddpath"
  
```

Execution Plan window

An Execution Plan window enables you to create and update an execution plan definition. The Execution Plan window consists of the Script and MedML tabs.

Script tab

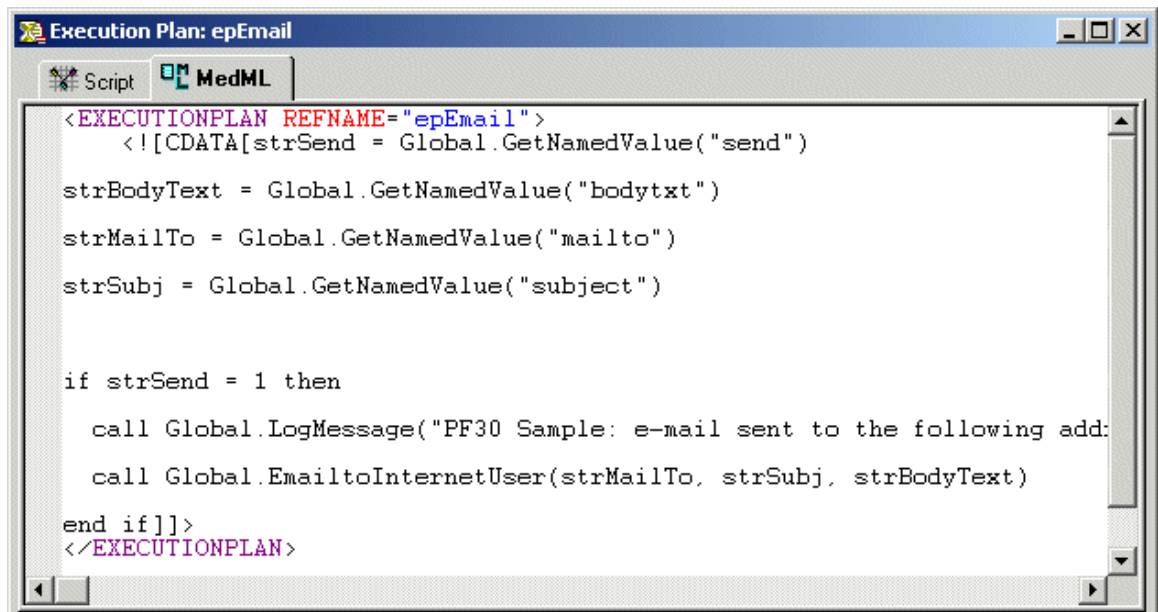
The Script tab enables you to view and edit an execution plan script.



MedML tab

The MedML tab provides a read-only display of the MedML tags that define the execution plan.

To open an online help file containing detailed information about each MedML tag, select **Help > MedML Help**.

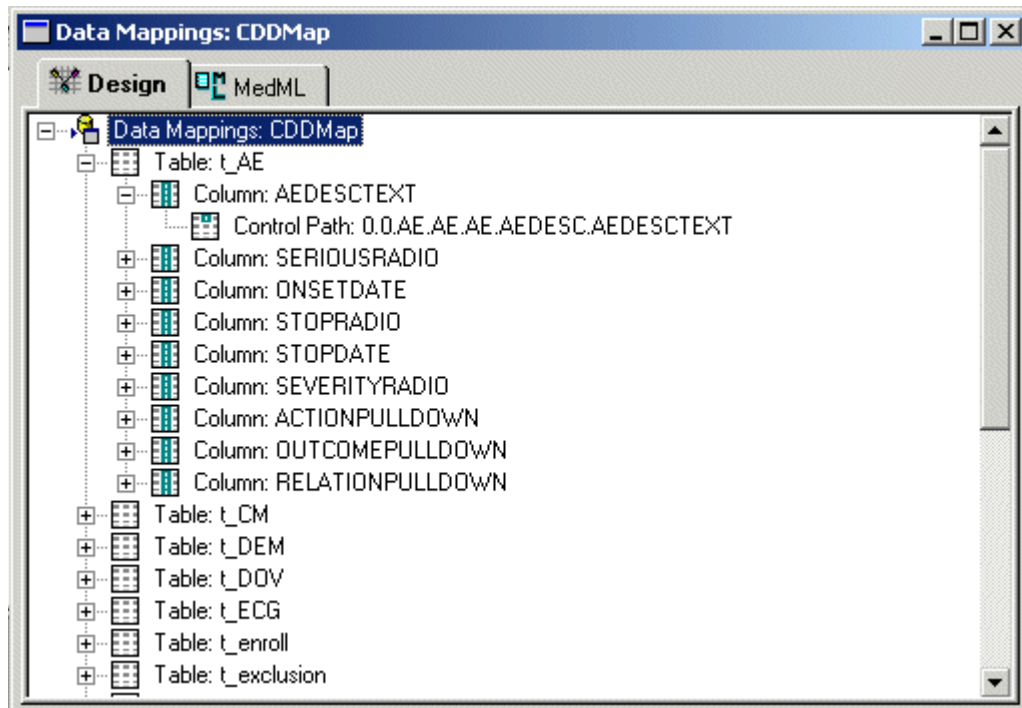


Data Mappings window

The Data Mappings window enables you to create, update, and view the data mappings between controls on a form and an external system or a calculated control designed to hold a code generated by an external autocoding tool. The Data Mappings window consists of the Design and MedML tabs.

Design tab

The Design tab represents the structure of a data mapping definition as a tree.



Right-click menus

Each component of a data mapping definition is associated with a right-click menu that you activate by right-clicking the component name in the Design tab. Use the commands on these menus to design, edit, or refine your data mapping definition.

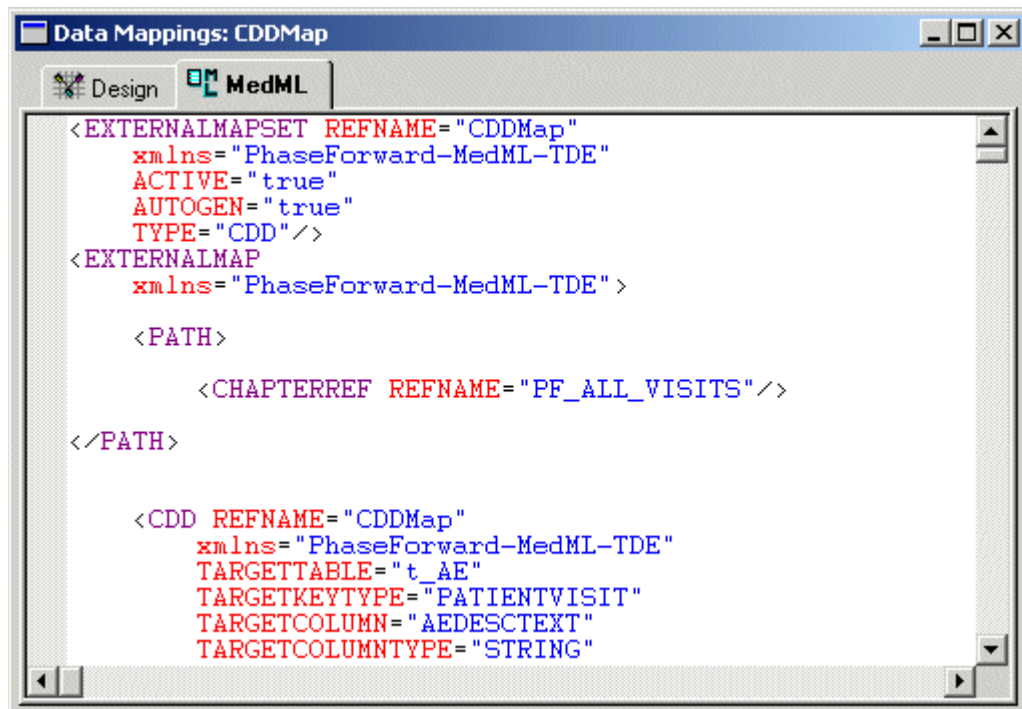
Dragging and dropping mapped controls

You can drag a control from a Form window onto a data mapping table or column to which you want to map the control, as an alternative to using the commands on the right-click menu for a table or column.

MedML tab

The MedML tab provides a read-only display of the MedML tags that define the data mapping. As you create, update, or delete the components of a data mapping definition, the InForm Architect application updates the corresponding MedML tags.

To open an online help file containing detailed information about each MedML tag, select **Help > MedML Help**.



```
<EXTERNALMAPSET REFNAME="CDDMap"
  xmlns="PhaseForward-MedML-TDE"
  ACTIVE="true"
  AUTOGEN="true"
  TYPE="CDD"/>
<EXTERNALMAP
  xmlns="PhaseForward-MedML-TDE" >

  <PATH>

    <CHAPTERREF REFNAME="PF_ALL_VISITS"/>

  </PATH>

  <CDD REFNAME="CDDMap"
    xmlns="PhaseForward-MedML-TDE"
    TARGETTABLE="t_AE"
    TARGETKEYTYPE="PATIENTVISIT"
    TARGETCOLUMN="AEDESCTEXT"
    TARGETCOLUMNTYPE="STRING"
```

Toolbars

The toolbars in the InForm Architect application give you shortcuts to some of the most frequently performed activities. This section introduces the toolbars and describes how you can use the buttons on each toolbar. The toolbars are:





- Main
- Edit
- Rules
- Trial
- Object Palette
- Settings







Main toolbar

The Main toolbar enables you to:

- Create a new trial, form, rule, event, execution plan, element, or unit.
- Open a trial or a MedML file.
- Save trial components.
- Cut, copy, or paste text or component definitions.
- Print.
- Display the InForm Architect application online Help.

The following table describes each button on the Main toolbar:





Button	Description
	<p>New—Create a new trial, form, rule, event, execution plan, element, or unit. When you click the New button, one of the following happens:</p> <ul style="list-style-type: none"> • If no trial is open, the Trial wizard opens so you can create a new trial. • If a trial is open, a Form editor window opens. When you click the arrow next to the button, a menu appears from which you can choose the component you want to create.
	<p>Open Trial—Open a trial. When you click the Open Trial button, the Open Trial dialog box appears, and you can select the trial you want to open.</p>
	<p>Insert MedML—Open an .xml or .rsp file. When you click the Open button, the Open dialog box appears, and you can select the MedML file you want to open.</p>
	<p>Save—Save the trial component in the active editor window as an .xml file. When you click the Save button, the Save dialog appears, and you can specify the name of the file in which you want to save the component definition.</p>





Button	Description
	<p>Save All—Save generated XML and files and response file for all trial components in cache, whether or not they are used in the active trial definition.</p> <p>When you click the arrow next to the button, a menu appears from which you can choose how you want to save the trial:</p> <ul style="list-style-type: none"> • Save All—Save generated XML files and response file for all trial components in cache, whether or not they are used in the active trial definition. • Save Study—Save all components that are being used by the current trial, organized in XML files and a response file that you can efficiently load. The next time you open the trial, only the components defined in the Protocol Editor window are available. This method of saving enables you to clean up components that have been defined but are not being used in a trial. • Save By Component—Save components that are being used by the current trial. This method of saving creates XML files for each event, form and rule and an XML file for the trial definition, along with a response file for loading all generated XML files.
	Cut —Cut the selected text or form component from the location where you have selected it.
	Copy —Copy the selected text, visit, or form component from the location where you have selected it.
	Paste —Paste the contents of the clipboard into a selected location.
	Print —Print the contents of the active window.
	About —Display the InForm Architect application online Help.

Edit toolbar

The Edit toolbar provides tools for editing the text in the MedML or Script tab of a component definition window.

The following table describes each button on the Edit toolbar.

Button	Description
	Undo —Reverse the previous action.
	Redo —Reinstate the action you previously reversed.
	Find —Search for a specified string in a text window (the MedML or Script tab of a component editor window). When you click the Find button, a Find dialog box opens so that you can specify search criteria.
	Find Next —Repeat the previous text string search.





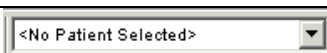
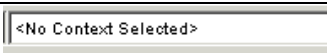
Button	Description
	Toggle Bookmark —Create or remove a bookmark at the cursor location in a text window. This button enables you to mark or unmark a specific line in the MedML or Script tab of a component editor window.
	Next Bookmark —Jump to the next bookmark in a text window.
	Previous Bookmark —Jump to the previous bookmark in a text window.
	Clear All Bookmarks —Clear all bookmarks from a text window.

Rules toolbar

The Rules toolbar helps you to create and test rule definitions by allowing you to:

- Use context-specific arguments and test data when you run a rule in test mode, or select the patient against whose data you want to run the rule.
- Check the RefNames in a rule definition.
- Run a rule script.

The following table describes each button on the Rule toolbar:




Button	Description
	Toggle RefName Check —Check the RefNames used in the current rule definition.
	Run Script —Run the current rule script interactively. You can also choose a patient and/or context with which to run the rule.
	Run Selected Testcases —Run the selected test cases from the current rule script.
	Run All Testcases —Run the entire current rule script, using data from the source specified in the pulldown lists in the Rules toolbar.
	Specify the patient whose data you want to use for testing a rule. Click the arrow to make the list drop down, then use the up and down arrow keys on your keyboard to scroll through the list.
	Specify the context containing the data you want to use to run a rule. If you select No Context selected , the InForm Architect application runs the rule by using the default arguments and values specified in the rule definition.

Trial toolbar

The Trial toolbar gives you access to the following trial component windows:

- Protocol editor
- Element editor window
- Mappings editor

The following table describes the buttons in the Trial toolbar:

Button	Description
	Protocol Editor —Open the Protocol Editor so that you can create, update, or delete visits in a trial.
	Element Editor —Open the Element Editor window so that you can create update data element definitions.
	Create Data Mappings —Open the Create Data Mappings dialog box so you that you can create a new data mapping definition. When you click the arrow next to the button, a menu appears from which you can choose the type of mapping to create.





Object Palette toolbar










The Object Palette toolbar enables you to create new visits or form components by dragging icons from the toolbar.

To create a new visit, drag the visit icon onto the appropriate location in the Protocol Editor window tree.

To create a new form component, drag its icon onto the appropriate location in the form design tree in a Form editor window.

The following table describes each button on the Object Palette toolbar.




Button	Creates a new...
	Visit —To create the first visit in a trial, drag the icon onto the trial icon in the Protocol Editor window. To create a subsequent visit, drag the icon onto the visit that the new visit will follow.
	Association —To create an association, drag the icon onto the Associations selection in the Protocol editor Design tab. To associate repeating forms, drag the icons representing the repeating forms from the Trial Objects list onto the new association.
	Section —To create a new section, drag the icon onto the section that the new section will follow in the active Form editor window.
	Itemset —To create a new itemset, drag the icon onto the section in which you want to include it in the active Form editor window.




Button	Description
	Item —To create the first item in a section, drag the icon onto the section icon in the active Form editor window. To create a subsequent item, drag the icon onto the item that the new item will follow.
	Group control —To create a new group control, drag the icon onto the item or control in which you want to include it in the active Form editor window.
	Radio group control —To create a new radio group control, drag the icon onto the item or control in which you want to include it in the active Form editor window.
	Checkbox group control —To create a new checkbox group control, drag the icon onto the item or control in which you want to include it in the active Form editor window.
	Pull-down list control —To create a new pull-down list control, drag the icon onto the item or control in which you want to include it in the active Form editor window.
	Text box control —To create a new text box control, drag the icon onto the item or control in which you want to include it in the active Form editor window.
	Date/time control —To create a new date/time control, drag the icon onto the item or control in which you want to include it in the active Form editor window.
	Calculated control —To create a new calculated control, drag the icon onto the item or control in which you want to include it in the active Form editor window.
	Simple control —To create a new simple control, drag the icon onto the item or control in which you want to include it in the active Form editor window.

Settings toolbar

The Settings toolbar enables you to specify system-wide InForm Architect application settings. These buttons are also accessible when you select **View > Settings**.

The following table describes each button in the Settings toolbar.

Button	Description
	Install MedML When Saving —Run MedML Installer when you save trial component definitions. MedML Installer processes the XML generated during the save and loads the trial component definitions into the trial database.
	Strict MedML Checking —Enforce completeness of component definitions when you insert MedML into a trial or when you install MedML when saving. If you do not select this button, MedML Installer allows you to load partially complete component definitions into the trial database; for example, when you are in an early trial design phase.
	Toggle Rule Display —Determine whether the Form editor window displays rule attachments graphically. When you activate this option, the representation of each item includes a node that shows the rules attached to the item, along with the visits to which the rules are attached.

Button	Description
	Toggle Annotated View —Display CRF with metadata including: control reference name definitions, control data type and data format definitions, element definitions, date ranges, and (optionally) data mappings.
	Toggle Annotated Data Mappings Display —Display annotated Data Mappings information, if you have created a data mapping definition.
	Toggle Annotated Data Mapping Table Display —Display annotated data mapping tables, if you have created a data mapping definition.

Customizing the application workspace

The InForm Architect application enables you to customize the appearance of the main application workspace for your maximum convenience. This section describes the customization options you can use:

- **Toolbars**—Display or hide toolbars, rearrange toolbar positions, move them in and out of the application window, and create custom toolbars that contain icons you choose.
- **Windows**—Display or hide windows, rearrange window positions, move them in and out of the application window, and minimize or maximize window size.
- **Settings**—Select **View > Settings** to affect Settings toolbar buttons and other options.

Note: The customizations that you make to the InForm Architect application workspace are for the duration of a session. They are not saved when you close and then reopen the application.

Customizing toolbars

This section describes how to customize the display of application toolbars.

Displaying and hiding toolbars

By default, all toolbars are visible when you start the InForm Architect application. To hide a toolbar, use either of the following methods:

- Select **View > Toolbars**; then click to clear the selection of the toolbar you want to hide.
- Click anywhere in the toolbar section of the main window with the right mouse button. In the pop-up menu that appears, click to clear the selection of the toolbar you want to hide.

To restore the display of a hidden toolbar, repeat the above instructions, but select the toolbar you want to display.

Rearranging positions

To move a toolbar to a different position in the toolbar area at the top of the application, drag the vertical handle at the left of the toolbar to the new location. When you release the mouse button, the toolbar snaps into position.



Docking and undocking toolbars

By default, toolbars are positioned (docked) in the toolbar area at the top of the application window. You can convert them into moveable windows and position them anywhere on the screen.

To undock a toolbar and change it into a window, drag its repositioning handle out of the toolbar area. When you release the mouse button, the toolbar's appearance changes to a window with a narrow title bar. You can use the title bar to drag the window anywhere you want on the screen.





To return a toolbar to its docked position, drag its title bar back into the toolbar area. When you release the mouse button, the toolbar snaps into the toolbar position closest to the cursor.

Note: To prevent a toolbar from being docked when you drag it near the toolbar area, hold down the **Ctrl** key while you drag the toolbar.

Choosing toolbar button appearance

The InForm Architect application enables you to choose between two views of toolbar buttons:

Example	Description
	Regular toolbar buttons consist of icons with no identifying text.
	Large toolbar buttons include a text label.

To toggle between the two toolbar button views, do either of the following:

- Select **View > Toolbars > Large Toolbar Buttons**.
- Right-click in the toolbar area of the InForm Architect application window then choose **Large Toolbar Buttons** from the pop-up menu.


Customizing windows

This section describes how to customize the appearance of the Output, Properties, and Trial Objects windows.

Displaying and hiding windows

By default, the Output, Properties, and Trial Objects windows are visible when you start the InForm Architect application.

To hide a window, use any of the following methods:

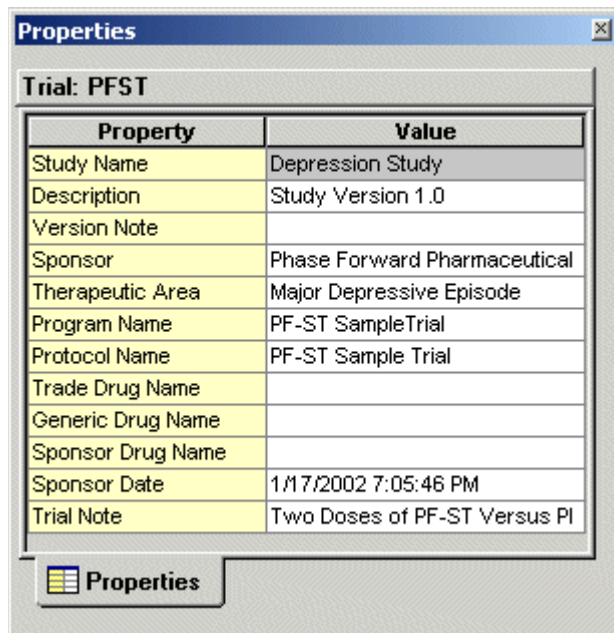
- Deselect the window name from the **View** menu.
- Right-click in the window, and choose **Hide Window** from the pop-up menu that appears.
- Click the **Close** button  in the upper right corner of the window.

To restore the display of a hidden window, select the window name from the **View** menu.

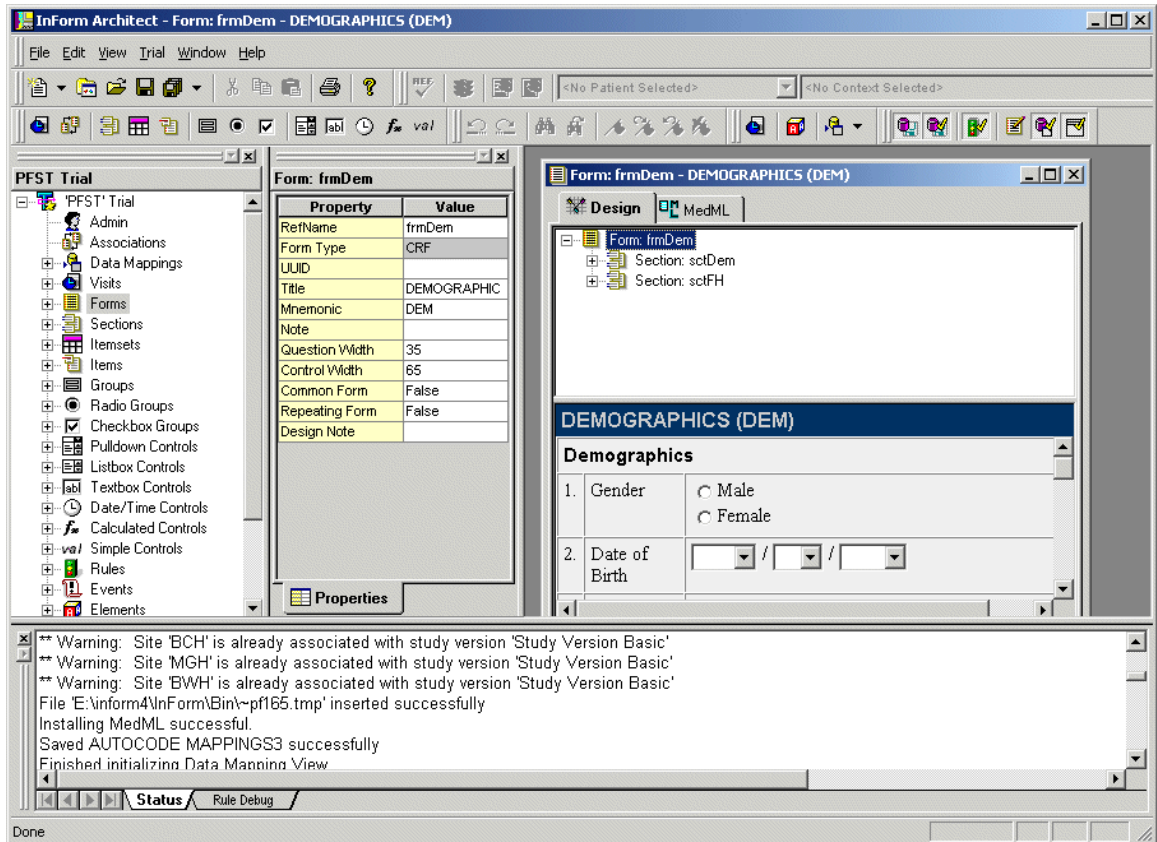
Docking and undocking windows

By default, the Output, Properties, and Trial Objects windows are positioned (docked) in panes of the application window. You can convert them into moveable windows and position them anywhere on the screen.

To undock one of these windows and change it into a moveable window, drag its repositioning handle out of the pane where it is located. When you release the mouse button, the window's appearance changes to a moveable window with a narrow title bar. You can use the title bar to drag the window anywhere you want on the screen.



If you release the mouse button when the window is near a defined pane, the window snaps to a docked position against the pane.



To return a window to its original docked position, drag its title bar back into the window's original pane. When you release the mouse button, the window snaps into the docked window position closest to the cursor.

Note: To prevent a window from being docked when you drag it near its pane, hold down the **Ctrl** key while you drag the window.

Maximizing and restoring

To expand a window so that it occupies the entire pane in which it is located, click the **Maximize** button in the upper right corner (☐).

To restore a maximized window to its original size, click the **Restore** button (☐).

Using editing and navigation features

The InForm Architect application features several tools for performing editing and navigation activities. This section describes how to:

- Modify trial component definitions, including renaming them.
- Cut, copy, and paste.
- Search for text strings.
- Work with bookmarks in text definitions.

Modifying trial components

In the InForm Architect application, you can modify trial components by:

- Changing the order or composition of the trial.
- Modifying descriptive and functional properties of the components.

Changing trial order or composition

You can reorganize the trial structure at the visit or form level by:

- Creating new components in place in the trial hierarchy.
- Dragging existing components from the Trial Objects window into their locations in the trial hierarchy.
- Removing components from the trial hierarchy.
- Moving components to new locations in the trial hierarchy.

For details on how to do each of these activities, see *Defining Visits* (on page 75) and *Managing forms and visits* (on page 99).

Changing property definitions

The means for modifying the definition of most trial components is the Properties window. The exception to this method applies to definitions of elements, which have a specialized definition window.

Using the Properties window

To modify a trial component definition, change the values of its properties as necessary.

To change a specific property:

- 1 In the **Design Workspace**, open the editor window containing component whose properties you want to edit, and select the component.

- 2 In the **Properties** window, click the **Value** cell that corresponds to the property you want to change:
 - To change a text property, edit the text in the **Value** cell, and press **Enter**.
 - To change a selected property, click the **Value** cell to reveal a pull-down list of selections then click the new selection in the pull-down list and press **Enter**.
 - To change a property requiring multiple selections, click the **Value** cell to reveal a pull-down list of selections then click each check box selection that applies and press **Enter**.

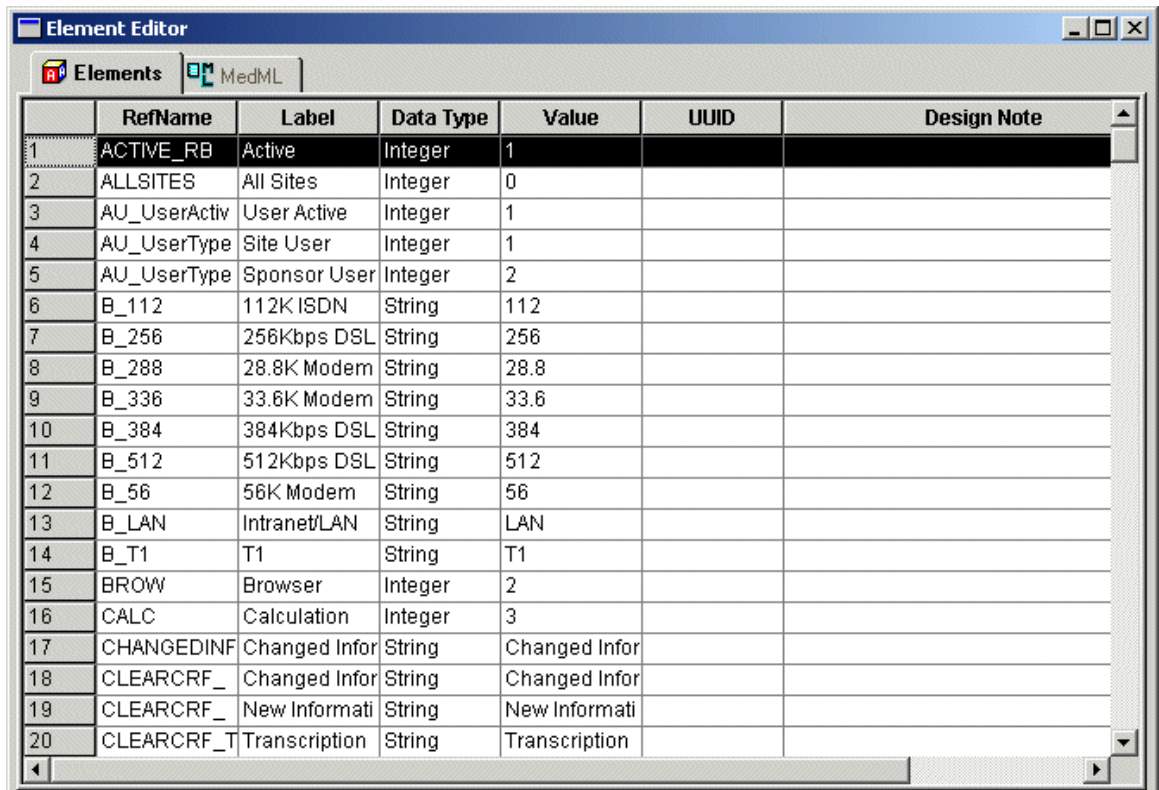
Modifying element definitions

To change element definitions, use the Element Editor window.

To open this window, do any of the following:

- Select **Trial > Elements**.
- Click the **Element Editor** button on the **Trial** toolbar.
- Double-click an element in the **Trial Objects** window.

To change a property in this window, click the cell you want to change and type or select a new value. For more information, see *Defining element components* (on page 134).



The screenshot shows the 'Element Editor' window with a table of element definitions. The table has columns for RefName, Label, Data Type, Value, UUID, and Design Note. The rows are numbered 1 through 20.

	RefName	Label	Data Type	Value	UUID	Design Note
1	ACTIVE_RB	Active	Integer	1		
2	ALLSITES	All Sites	Integer	0		
3	AU_UserActiv	User Active	Integer	1		
4	AU_UserType	Site User	Integer	1		
5	AU_UserType	Sponsor User	Integer	2		
6	B_112	112K ISDN	String	112		
7	B_256	256Kbps DSL	String	256		
8	B_288	28.8K Modem	String	28.8		
9	B_336	33.6K Modem	String	33.6		
10	B_384	384Kbps DSL	String	384		
11	B_512	512Kbps DSL	String	512		
12	B_56	56K Modem	String	56		
13	B_LAN	Intranet/LAN	String	LAN		
14	B_T1	T1	String	T1		
15	BROW	Browser	Integer	2		
16	CALC	Calculation	Integer	3		
17	CHANGEDINF	Changed Infor	String	Changed Infor		
18	CLEARCRF_	Changed Infor	String	Changed Infor		
19	CLEARCRF_	New Informati	String	New Informati		
20	CLEARCRF_T	Transcription	String	Transcription		

Renaming a component

To rename a component, change its **RefName** property. The InForm Architect application updates the component definition and changes the tabs in any open windows to show the new RefName.

Note: You cannot change the RefName of an element, unit, or form component if its definition has already been loaded into the database.

Cutting, copying, and pasting

The InForm Architect application makes it easy for you to reproduce component definitions that you use in more than one location by providing the ability to cut, copy, and paste the following:

- Form component definitions.
- Visit definitions (copy and paste only).
- Component properties in the Properties window.
- Text strings in rule and execution plan scripts.

Except as noted, you can cut, copy, and paste by using the standard Edit menu commands or shortcut keys:

Action	Edit menu command	Shortcut key
Cut	Cut	Ctrl + x
Copy	Copy	Ctrl + c
Paste	Paste	Ctrl + v

Form component definitions

The InForm Architect application enables you to copy the definitions of form components from one form to another or from one location within a form to another. You can either copy an existing component and use it as is in another location, or create a new component based on the composition of the original component. You can copy and paste components by using the Microsoft Windows Clipboard or by duplicating objects in the Trial Objects window.

Using the Microsoft Windows Clipboard method

When you use this method, the InForm Architect application creates the copied component in the new location. In most cases, the component in the new location is the same object as the original. However, if necessary for the integrity of the form structure, the InForm Architect application creates a new component with the same properties as the original but with the prefix “Copy of” added to the name.

For example, item definitions must be unique within a form. If you copy an item definition to a new location in the same form, the InForm Architect application creates a new item component in the new location. However, if you copy a radio group from one group control to another, the InForm Architect application simply creates a reference in the new group control to the original radio group definition.

When you copy component definitions that include child components (for example, a section that includes items), the InForm Architect application copies the child components along with the top-level components. Note that the child definitions are not new components but references to the original component definitions.

You can cut components from one location and paste them to another in the same manner.

When you are working in the Form editor, you can copy and paste a form, section, itemset, item, or control component as follows:

- 1 Select the component.
- 2 Select **Edit > Copy**. Alternatively, press **Ctrl+C**.
- 3 Select the component under which you want the copy to be included. For example, to copy an item definition, select the section or itemset where you want the copy to go.
- 4 Select **Edit > Paste**. Alternatively, press **Ctrl+V**.

Duplicating objects in the Trial Objects window

When you use this method, you copy forms or form components by selecting them in the Trial Objects window and choosing one of two Duplicate commands from a pop-up menu. The InForm Architect application creates a new component in the Trial Objects window. This component has the structure of the original component and a new RefName created from the name of the original component and the prefix “Copy of.”

To use the new component in a form, drag it from the Trial Objects window onto the appropriate location in the Form editor window.

You can choose whether any child components are copied as references to the original component or are created as new components in the same way as the parent component. If you choose to duplicate child components, the InForm Architect application creates new components down to the control level. It does not duplicate elements or units.

By using this method, you can duplicate forms or form components.

To create duplicate components and use them in a form or visit:

- 1 In the **Trial Objects** window, right-click the form or component you want to duplicate.

Note: To quickly locate the component you want to duplicate, right-click the component in the Form editor window, and click **Locate in Trial Window** on the pop-up menu. The InForm Architect application highlights the object in the Trial Objects window.

- 2 Choose one of the following from the pop-up menu:
 - To create a new component with references to the original definitions of all its subcomponents, choose **Duplicate**. In most cases, this level of copying is sufficient.
 - To create a new component containing all new subcomponents, choose **Duplicate With Children**.

The InForm Architect application creates new components named like the originals but with the prefix **Copy of**. If you create multiple copies, the InForm Architect application adds a copy number to the prefix.

- 3 Drag the new component to the appropriate location in the **Form editor** window, or, if the component is a form, in the **Protocol Editor** window.
- 4 In the **Properties** window, edit the component properties as necessary.

Restrictions

When you copy or duplicate form components, observe the following restrictions:

- You cannot paste or drag copied or duplicated components into a location that would create more than five levels of nesting.
- You cannot paste hierarchical components such as groups, radio groups, or check box groups into a location that would create a circular definition between parent and child components. For example, in the following structure:

Group1

 Radio Group1

 Checkbox Group 1

You cannot copy Group1 and paste it as a child component under Checkbox Group1.

Visit definitions

In the Protocol Editor, you can make a copy of a regular or dynamic visit and paste it within the same visit group or between the regular and dynamic visit groups. For example, if a visit that occurs in the fifth week of a trial has the same forms as an earlier visit, you can copy and paste the earlier visit.

When you copy and paste a visit in this way, the InForm Architect application creates a new visit with the same name as the original, prefixed by “Copy of.” This new visit contains the original visit’s forms and components. The InForm Architect application does not create duplicate forms and components. If you create multiple copies of the same visit, the name of the new visit contains the number of the copy. You can then use the Properties window to rename the new visit appropriately.

If you copy and paste a regular visit into the Dynamic visit group, or paste a Dynamic visit into the Visits group, the InForm Architect application automatically updates the value of the Dynamic property appropriately.

To copy and paste visit definitions, use the Edit menu commands or the standard Microsoft Windows shortcut keys.

Note: There is no cut capability for visit definitions.

Component properties



In the Properties window, you can cut, copy, and paste property definitions of all types of trial components by using Edit menu commands or the standard Microsoft Windows shortcut keys.

Text strings

You can cut, copy, and paste text strings in the Script tabs of the Rule and Execution Plan editor windows. Use Edit menu commands or the standard Microsoft Windows shortcut keys.

Searching for text strings

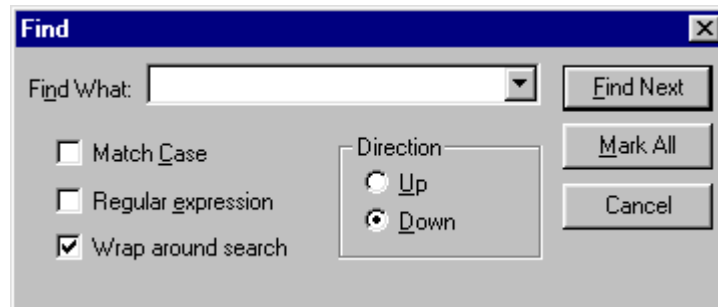
The Search tool in the InForm Architect application enables you to search for strings in text windows, that is, in MedML tabs and in the Script tab of the Rule editor window. This tool is accessible through the following buttons in the Edit toolbar:

Button	Description
	Find
	Find Next

Searching for a text string

To search for a text string:

- 1 Click the **Search** button in the **Edit** toolbar. The **Find** dialog box opens.



- 2 Specify the parameters of the search by using the check box and **Direction** options.
- 3 To start the search, click **Find Next**.
- 4 To have the tool find all occurrences of the string and create bookmarks where they occur, click **Mark All**.
- 5 To exit the search tool, click **Cancel**.





Repeating a search

To find successive occurrences of a text string:

- 1 Define the search criteria as specified in the previous section.
- 2 To find the next occurrence of the string, click the **Find Next** button in the Edit toolbar.
- 3 Repeat the search as often as necessary by clicking the **Find Next** button again.

Working with bookmarks

The bookmark tool enables you to mark the text in a text window and then jump to a bookmarked line. This tool is accessible with the following buttons in the Edit toolbar:

Button	Description
	Toggle bookmark
	Next bookmark
	Previous bookmark
	Clear all bookmarks

Creating a bookmark

To create a bookmark:

- 1 Position the cursor in the line of the text window where you want the bookmark to be created.
- 2 Click the **Toggle Bookmark** button in the **Edit** toolbar.

Removing a bookmark

To remove a specific bookmark:

- 1 Position the cursor in the text window line containing the bookmark you want to remove.
- 2 Click the **Toggle Bookmark** button in the **Edit** toolbar.

To remove all bookmarks, click the **Clear All Bookmarks** button in the Edit toolbar.

Navigating among bookmarks

To move to the next bookmark after the cursor location in a text window, click the **Next Bookmark** button in the Edit toolbar. To move to the previous bookmark, click the **Previous Bookmark** button.

Deleting trial components

When you are editing trial components in a component editor window in the Design Workspace, the InForm Architect application enables you to delete a selected component by choosing a command from a right-click pop-up menu or from an application menu.

When you delete a component, you remove it from the trial definition. This has the following implications:

- If you delete a component that is included in another component definition (for example, if you delete a text box control from a radio group), the composition of the parent component changes in every location where it is used. In the example, if you delete the text box control while editing the Vital Signs form, and the radio group is included in both the Vital Signs and the ECG forms, both forms now contain the radio group without the text box. The InForm Architect application warns you of this implication when you attempt to perform the deletion.
- Deleting a component from the trial definition does not remove the component from the collection of objects available for the trial. For example, if you decided to restore the deleted text box control to the radio group, you could drag its definition back onto the radio group from the Trial Objects window.
- When you save a trial by selecting **File > Save Trial > Save Study**, components you have deleted from the trial definition are not included in the trial MedML that the InForm Architect application generates and are not reloaded into the caches when you reopen the trial. In the example, unless the text box was used elsewhere in the trial than the radio group, it would not be reloaded into the caches after you save the trial.
- When you save a trial by selecting **File > Save Trial > Save All**, all defined components are included in the MedML that the InForm Architect application generates and are reloaded into the caches when you reopen the trial, whether or not they are included in the trial definition. Either this option or the Save By Component option is recommended while a trial definition is under development.

Note: You cannot delete definitions of element or unit components.

CHAPTER 3

Setting up a trial

In this chapter

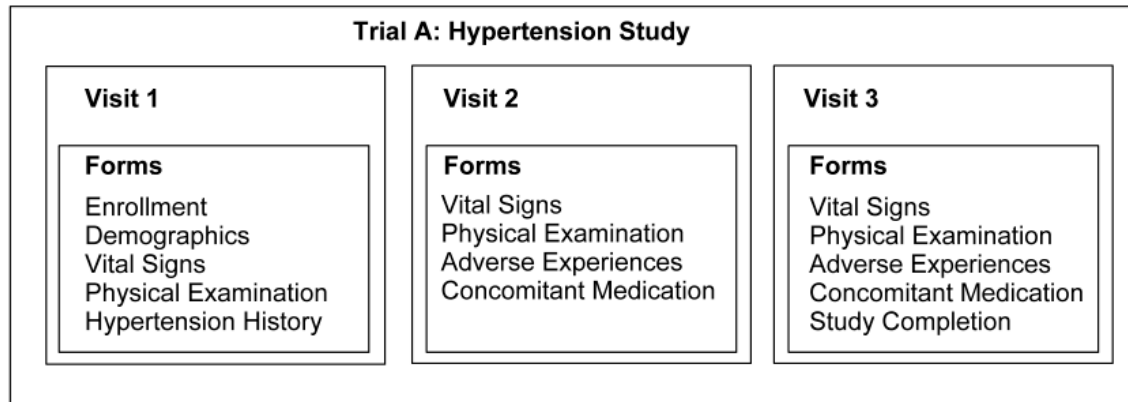
Overview	58
About trials.....	59
Creating a new trial.....	60
Opening an existing trial.....	64
Closing a trial	66
Inserting MedML into a trial.....	67
Saving a trial.....	69
Moving a trial into production.....	74

Overview

This chapter describes how to set up and manage a trial and how to perform trial administration activities such as management of users, sites, groups, and rights.

About trials

When an investigator collects data from a patient during a patient visit, the data are transferred to a set of *forms*. Collectively the forms for a particular patient visit to a site make up a *visit*. The collection of all trial visits is called a *trial*. The definition of a trial contains definitions of the trial's visits, which in turn contain the definitions of forms, in a hierarchical structure.




A trial definition represents all of a trial's forms and visits at a particular point in time. If a form changes, or a form or visit is added or dropped, a new trial definition is required.

You can map each trial to one or more sites and to a set of trial documentation. Each time a component of a trial changes, you create a new trial and map the new version to the sites and trial documentation to which it applies. With this mapping, you can ensure that when a protocol change requires a change in forms, the new forms come into effect as each site's IRB approves the protocol change.

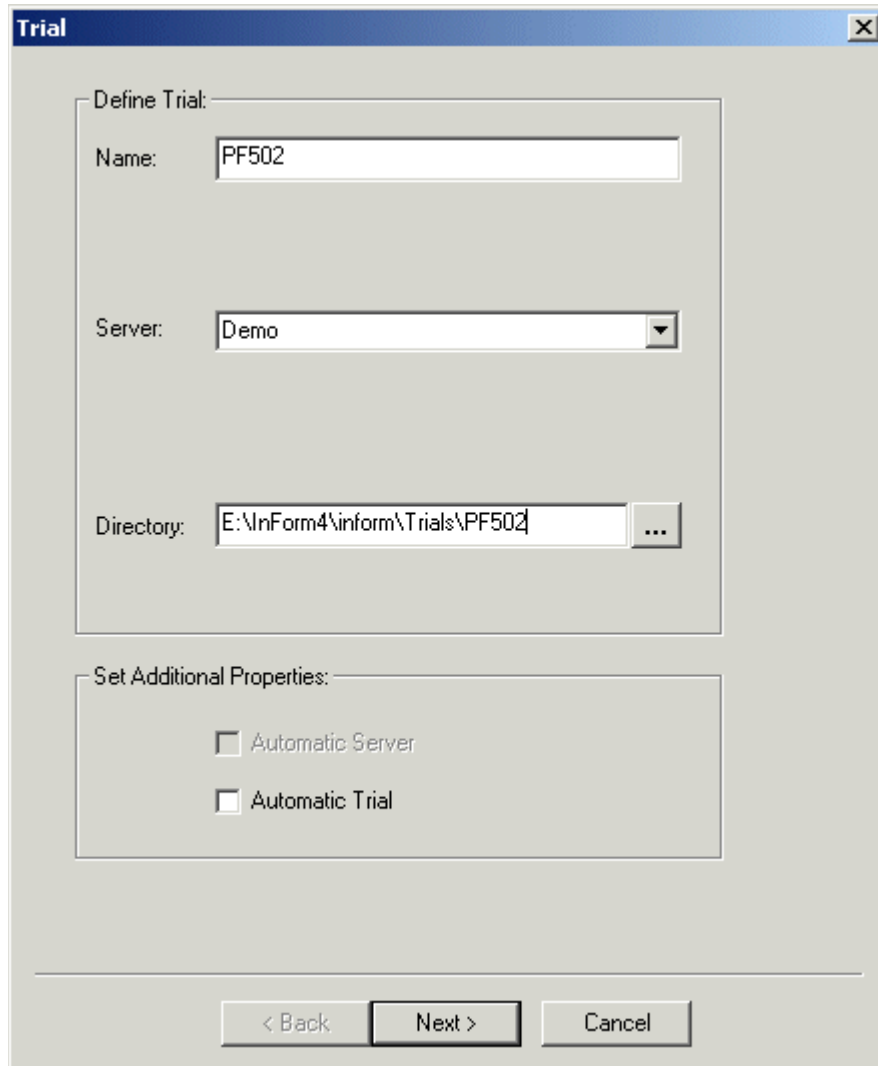
Creating a new trial

When you create a new trial, the InForm Architect application creates a trial definition, loads the Base trial, and opens the new trial to enable you to work on it.

To create a new trial:

- 1 Select **File > New > Trial**. If no trial is open, you can do the same thing by clicking the **New** button  in the **Main** toolbar.

If a trial is currently open, the InForm Architect application prompts you to specify whether you want to close the currently open trial. If you click **Yes**, the new trial wizard appears, with the **Trial** window visible.



Trial

Define Trial:

Name: PF502

Server: Demo

Directory: E:\InForm4\inform\Trials\PF502

Set Additional Properties:

Automatic Server

Automatic Trial

< Back Next > Cancel

- 2 In the **Trial** window, specify the following information:
 - In the **Name** box, enter the name of the trial. The name must have the following characteristics:
 - Maximum length of 16 characters.
 - Alphabetic first character.
 - No special characters except underbar ().
 - In the **Server** box, select the name of the server on which the trial runs. If you want to set up a new server that is not in the list, clear the name currently showing in the **Server** box, and enter the name of the new server.
 - In the **Directory** box, browse for or enter the name of the directory where the XML files defining the trial objects will be stored.
- 3 If you want the server to start automatically when you start the InForm Architect application, select the **Automatic Server** check box.
- 4 If you want the trial to start automatically when you start the InForm Architect application, select the **Automatic Trial** check box.
- 5 Click **Next**. The **Database** window appears.

- 6 If the trial will use a previously defined database connection, click **Use Existing DSN**, and provide the following information:
 - In the **Trial DSN** box, specify the name of the DSN for the trial.
 - In the **Report DSN** box, specify the name of DSN for the InForm application reports.
 - In the **UID** box, specify the name of the owner of the trial schema. UIDs must contain all alphabetic or all alphanumeric characters and begin with a letter. Do *not* use all numeric characters.
 - In the **Password** box, specify the password of the trial schema owner. Passwords must contain all alphabetic or all alphanumeric characters and begin with a letter. Do *not* use all numeric characters.

- 7 If you want to define a new database connection, click **Create New DSN**, and provide the following information:
 - In the **Connect String** box, specify the network service name for connecting to the InForm application database server.
 - In the **UID** box, specify the name of the owner of the trial schema. UIDs must contain all alphabetic or all alphanumeric characters and begin with a letter. Do **not** use all numeric characters.
 - In the **Password** box, specify the password of the trial schema owner. Passwords must contain all alphabetic or all alphanumeric characters and begin with a letter. Do *not* use all numeric characters.
 - In the **Verify Password** box, repeat the password.

Note: Including a trial identifier in the definition of the username and password is recommended.


- 8 Click **Finish**.

The InForm Architect application starts to create the trial. If a trial with the same name already exists, the InForm Architect application displays a warning message indicating that if you continue, all data in that trial will be lost.

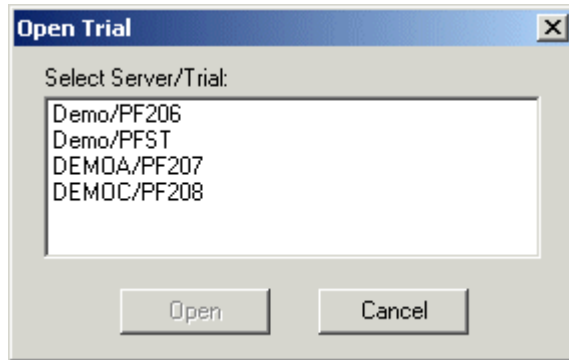
If there is no existing trial by the same name, or you indicate that the trial creation should proceed, the InForm Architect application creates the trial, loads the Base trial, and opens the trial for your use.

Opening an existing trial

To open a trial:

- 1 Select **File > Open Trial**. As a shortcut, you can click the **Open Trial** button  on the **Main** toolbar.

The **Open Trial** dialog box appears.



- 2 In the **Select Server/Trial** list, select the trial you want to open.
- 3 Click **Open**.

When you open a trial in the InForm Architect application, the application checks whether the trial is currently running:

- If the trial is not running, the InForm Architect application starts it.
- If the trial is running, the InForm Architect application attaches to it.

Note: If you install the InForm Architect application after installing the InForm application and starting a trial, you must stop and restart the InForm service before you can connect to the trial through the InForm Architect application.

Design and production mode

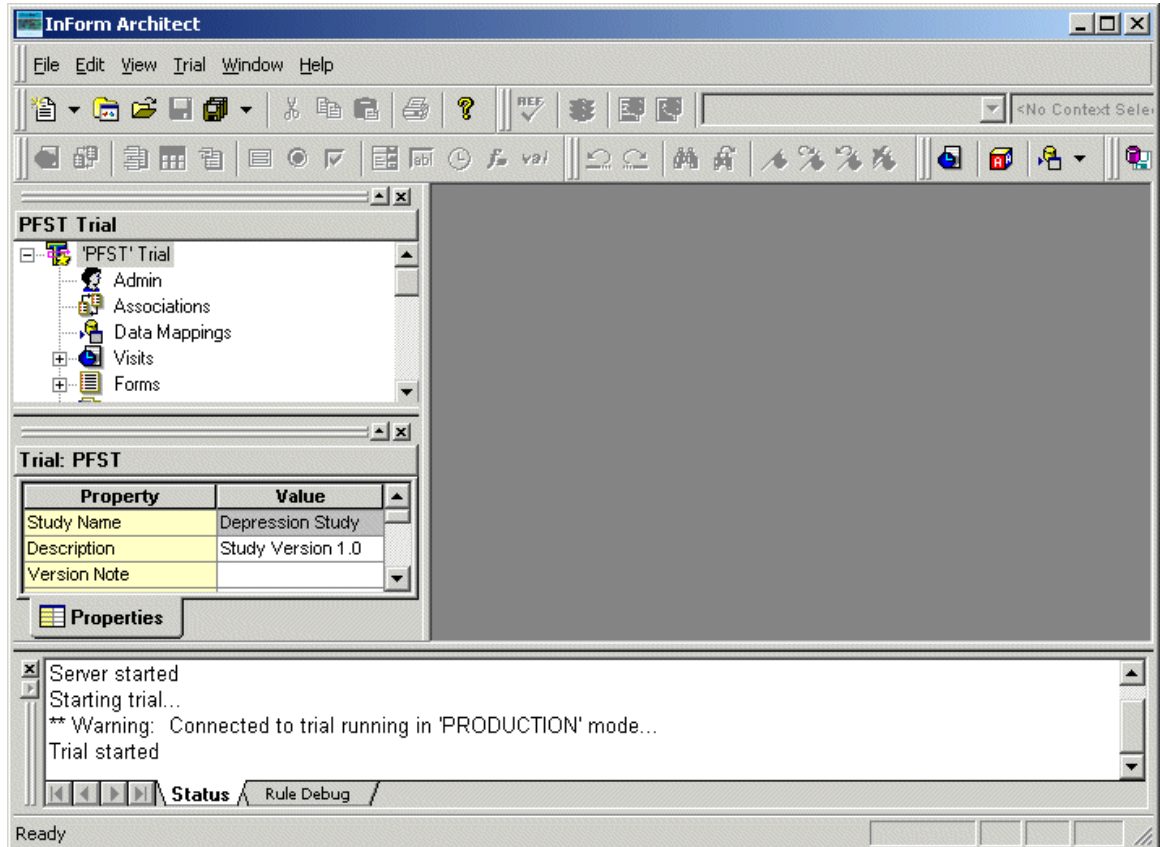
Trials can run in design or production mode:

- In **design mode**, the InForm application does not require that all metadata be consistent and complete when starting a trial. For example, you could start a trial in which not all controls referenced in a form have been defined and installed in the trial database. This mode enables trial designers to work with a trial in progress. When you open a trial with the InForm Architect application, the application starts it in design mode.
- In **production mode**, the InForm application checks to insure that all trial metadata are complete and consistent before permitting the trial to start. This mode prevents data corruption and system crashes in a production trial.

Design and production activities should always occur on different servers. You should never open a live production trial with the InForm Architect application, and the InForm Architect application should not be installed on a server hosting a live trial.

You can view and test trial design features that you are developing by starting the InForm application and working with the trial through the InForm application user interface. You should do this only if the trial is running in design mode.

If you open a trial that is running in production mode, the InForm Architect application displays a message to alert you of this fact.



If you receive this message:

- 1 Exit the InForm Architect application.
- 2 Stop the trial by issuing the following command at a Microsoft Windows prompt:


```
pfadmin stop trial trialname
```
- 3 Restart the InForm Architect application and open the trial.

Closing a trial

The InForm Architect application enables you to work with one trial at a time. Before you can open a different trial, you must close the trial you are currently working with. To close a trial:

- 1 Select **File > Close Trial**.
- 2 Click **OK**.

Inserting MedML into a trial

The InForm Architect application enables you to insert MedML component definitions into a running trial. You can insert either of the following types of files:

- An XML file containing component definitions
- A response file (.rsp extension) containing the names of XML files to insert

Once you have created a new trial or opened an existing trial, insert MedML by doing the following:

- 1 Select **File > Insert MedML**.
- 2 Use the **Open File** dialog box to select a previously defined XML or response file.
- 3 Click **Open**.

The InForm Architect application loads the component definitions into the trial, where they become available for selection in the **Trial Objects** window.

Specifying additional paths


When you create a .rsp file, you specify the location of each .xml file to load relative to the location of the .rsp file. The MedML Installer finds the XML files by using the path information in the .rsp file. If you attempt to insert individual XML files, the MedML Installer may be unable to resolve references to other files—for example, if you load user images in a set of user definitions.

To handle this situation, the InForm Architect application enables you to specify additional paths where XML files are located. When the MedML Installer processes an XML file, it searches each of the specified paths to find the files it requires.

To specify additional paths:

- 1 Select **View > Settings > Additional MedML Path**.
- 2 In the **Additional Path** dialog box, enter each additional path where MedML should search for files referred to in the XML files you want to process. Separate path names with semicolons.
- 3 Click **OK**.

Specifying strict MedML processing

To install trial component definitions into the database while saving trial components, the InForm Architect application activates the MedML Installer utility. During the trial design process, you may want to be able to load definitions that are not entirely complete. Later, you may want to ensure that the definitions you load are fully compliant with the MedML specification. The InForm Architect application accommodates both of these modes with the Strict MedML checking button  on the Settings toolbar:

- To specify that you want the MedML Installer to check that all required parts of a component definition are present when you save and install trial component definitions, select the **Strict MedML checking** button on the **Settings** toolbar. In this mode, the MedML Installer does not load a form if it does not have a section, a section if it does not have an item or itemset, an item if it does not have a control, and so on.
- To specify that you want the MedML Installer to load partial definitions of components, leave the **Strict MedML checking** button in the unselected state.

Installing the sample trial

If you set up either sample trial when you install the InForm application, the sample trial is available for opening when you start the InForm Architect application.

Alternatively, you can use the InForm Architect application to install the sample trial as follows:

- 1 Select **File > Insert MedML**.
- 2 Use the **Open File** dialog box to browse to the \InForm\Sample_Trial\FastStart directory in the InForm application installation.
- 3 Select the protocol.rsp file
- 4 Click **Open**.

Note: To see when the sample trial installation is complete, watch the messages in the output window.

Saving a trial

When you perform a Save operation, the InForm Architect application generates XML files defining some or all of the components in the open trial, according to the requested Save option, and saves the files in a location that you specify. The InForm Architect application offers the following options for saving a trial definition:


- Saving a copy of the trial definition XML.
- Saving all components in the Trial Objects window.
- Saving only the components that are actually used in the trial definition.
- Saving by component.
- Installing component definitions in the InForm application database as you save.

When you save trial component definitions by using the methods described in this section, the InForm Architect application generates XML files and response files that you can process with the MedML Installer to load the definitions into the InForm application database.

Note: Any new component definitions that you create during an InForm Architect application session exist in system cache only as long as the trial server is up. They are not loaded into the InForm application database unless you load the generated XML explicitly or when you save your work. To obtain the generated XML to load into the database, you must save component definitions. **If you stop the trial server before saving your work, you will lose it.** Therefore, saving frequently as you develop trial components is strongly recommended.

Saving a copy of the contents of an editor window

You can save a copy of the XML file for a trial definition or a specific form in a specified directory. To save a copy:


- 1 Select **File > Save Component** or **Save Component As**. As a shortcut, you can click the **Save** button  on the **Main** toolbar.
- 2 In the **Save Copy As** dialog box, specify the directory and file where you want to save the trial definition XML.
- 3 Click **Save**.

If the **Trial Design** window is active when you save, the InForm Architect application generates an XML file containing the definition of the trial. If a form window is active, the InForm Architect application generates an XML file containing the definition of the form.

Saving all components

When you save all components, the InForm Architect application saves generated XML files for all trial components in cache, whether or not they are used in the active trial definition. Each type of component is saved in a separate XML file; for example, there is an XML file for text box controls, one for pulldown list controls, one for rules, and one for events. Additionally, the InForm Architect application generates and saves a response file containing the names of all component XML files and saves the trial definition file for the active trial.

This method of saving trial components creates files that MedML Installer can process efficiently. You may find this method most useful when you plan to load a complete trial definition into a test environment. To save all components:

- 1 Select **File > Save Trial**. As a shortcut, you can click the **Save All** button  on the **Main** toolbar.
- 2 Then, choose **Save All**.
- 3 In the **Specify the Trial Directory** dialog box, specify the root directory in which to save the XML files for all defined components.
- 4 Click **OK**.

The following table describes the response and XML files created when you save all components. The XML files are shown in the order in which they appear in the response file, and thus the order in which they would be processed if you loaded the response file by using the MedML Installer.

Filename	Description
ADM_US.xml	User definitions
AllObjects.rsp	Response file containing the name of each saved XML file
CLCCTL.xml	Calculated control definitions
ConvertRuleStorage.xml	Unit conversion rule definitions
DATECTL.xml	Date/time control definitions
ELTCTL.xml	Simple control definitions
EXP_CH.xml	Execution plan definitions
FLTELT.xml	Float element definitions
FORM.xml	Form definitions
GPCTL.xml	Group control, radio group control, and check box control definitions
GROUP.xml	Query group, rights group, and signature group definitions, as well as the definition of the predefined manager user group for reporting
ITM.xml	Item definitions

Filename	Description
ITMSET.xml	Itemset definitions
LBCTL.xml	List box control definitions
NUMELT.xml	Numeric element definitions
PDCTL.xml	Pulldown control definitions
RIGHT.xml	Standard right definitions
RUL_CH.xml	Rule definitions
RUL_EV.xml	Event definitions
SECT.xml	Section definitions
SEQUENCE.xml	Sequence definitions
SITE.xml	Site definitions
SITEGROUP.xml	Sitegroup definitions
SPONSOR.xml	Sponsor definitions
STRELT.xml	String element definitions
StudyVersionName.xml	Trial definition and association with sites; filename is the version name of the trial definition
SVSITE.xml	Studyversion site definitions, associating a trial definition with a site.
SYSCONFIG.xml	System configuration settings
TDEDATAMAP.xml	Data mapping definitions (metadata)
TXTCTL.xml	Text control definitions
UNIT.xml	Unit definitions

Note: When you save all trial components repeatedly, the InForm Architect application creates backup copies of the most recently saved files. Each time you save, the InForm Architect application overwrites the backup with the previously saved version and overwrites the previously saved version with the new version.

Saving active trial components

When you save active trial components, the InForm Architect application saves generated XML files for only the trial components that are used in the current trial definition. Each type of component is saved in a separate XML file; for example, there is an XML file for text box controls, one for pulldown list controls, one for rules, and one for forms. Additionally, the InForm Architect application generates and saves a response file containing the names of all component XML files and saves the trial definition file for the active trial.

This method of saving trial components does not generate XML files for trial components that are part of the Base trial, or trial components that have been created but not included in the current trial definition.

This method of saving creates files that the MedML Installer can process efficiently. You may find this method most useful when you plan to load a complete trial definition into a clean production environment. You can also use this method during the development process to remove component definitions that you have created but that are not being used in the final trial definition. To save active trial components:

- 1 Select **File > Save Trial**. Alternatively, click the arrow next to the **Save All** button on the Main toolbar, and select **Save Trial** from the pulldown menu.
- 2 Select **Save Study**.
- 3 In the **Specify the Trial Directory** dialog box, specify the root directory in which to save the XML files for defined components that are used in the active trial.
- 4 Click **OK**.

Note: When you save active trial components repeatedly, the InForm Architect application creates backup copies of the most recently saved files. Each time you save, the InForm Architect application overwrites the backup with the previously saved version and overwrites the previously saved version with the new version.

Saving by component

When you save by component, the InForm Architect application generates the following XML files for components that are used in the active trial and saves them in the directory that you specify:


- One XML file for each event.
- One XML file for each form.
- One XML file for each rule along with its attachment definitions.
- One XML file for the trial definition.
- A response file, named StudyByComponent.rsp, that loads each form, event, and rule definition, along with the trial definition.

This method of saving trial components creates files that enable you to review the generated trial component XML in context. You may find this method to be the most convenient during the process of creating trial definition component definitions. To save by component:

- 1 Select **File > Save Trial > Save by Component**. Alternatively, click the arrow next to the **Save All** button on the Main toolbar, and then choose **Save By Component** from the pulldown menu.
- 2 In the **Specify the Trial Directory** dialog box, specify the root directory in which to save the XML files.
- 3 Click **OK**.

Installing trial component definitions in the database

When you save trial component definitions by using any of the methods described in this section, the InForm Architect application generates XML files and response files that you can process with the MedML Installer to load the definitions into the InForm application database. To load trial component definitions into the database, use either of the following methods:

- Use the MedML Installer to process the response file containing the names of all the component XML files you want to load.
- Click the **Install MedML When Saving** button  on the **Settings** toolbar. When this toggle button is depressed, the InForm Architect application generates the component XML files and response files appropriate to the save option you choose and also calls the MedML Installer to load the generated files into the database.

Working with a source control system

When you develop trial definitions with the InForm Architect application, it is strongly recommended that you work in conjunction with a source control system such as Microsoft Visual Source Safe. A source control system enables you to preserve versions of your trial definitions and prevents you from losing work because of local system failure.

As a general practice, you should check trial definitions into your source control system each time you perform a Save All, Save Study, or Save by Component operation.

Moving a trial into production

When your trial design is complete, you can install it in a test production or production environment. Before you move a trial from one environment to another, ensure that:

- The InForm application is installed on the production or test production server.
- A complete set of trial components is saved as MedML. When saving from the InForm Architect application:
 - Select **Strict MedML Checking**.
 - Deselect **Install MedML When Saving**.

To install a trial in a production or test production environment:

- 1 Save all trial components as MedML on the production machine.
- 2 Set up a new trial on the production server using the pfadmin command. For information about pfadmin, see *Installation and Configuration Guide*.
- 3 Install the basic trial components in the production trial using the dbsetup command. For information about dbsetup, see *Installation and Configuration Guide*.
- 4 Install the saved trial components in the production trial using the MedML Installer utility. For information about the MedML Installer, see *InForm Utilities Guide*.

CHAPTER 4

Defining visits

In this chapter

About visits	76
Creating a new visit.....	78
Renaming a visit	81
Reordering visits in a trial.....	82
Deleting a visit from a trial.....	83

About visits

In the InForm Architect application, visits are collections of forms. Most visits represent actual patient visits to an investigative site, and the forms are CRFs used to collect the data obtained from the patient at that visit. Additionally, the InForm Architect application supports the following types of specialized visits:

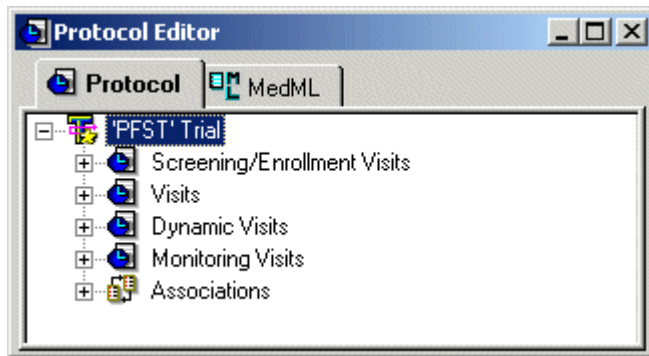
- **Screening and Enrollment**—In the InForm application, screening and enrollment is a two-step process. To register a patient on a Screening Log, users submit a screening form. To enroll a screened patient, users submit a form that captures enrollment criteria. The Screening visit, containing a screening form, and the Enrollment visit, containing an enrollment form, are required components of an InForm application trial.
- **Dynamic**—The InForm application supports the creation of visits that are generated automatically when patient data satisfies certain criteria that you test by attaching a rule to the relevant data item or items. For example, if you want to create a different sequence of visits for patients who are randomly assigned to receive different dosages of a drug in the third visit of a trial, you can do this by setting up dynamic visits.
- **Monitoring**—The InForm Architect application enables you to create the following types of forms to support site monitoring:
 - A Site Visit Report records data about a visit to a site.
 - A Regulatory Documentation Checklist records information about a review of site documents.

Each of these monitoring forms is included in a visit with special properties to identify it. Use the Visit Report default visit for Site Visit Reports, and use the Reg Docs default visit for Regulatory Documentation Checklist forms.

Along with these types of visits, the InForm Architect application supports the creation of *associations* between related types of forms. When forms are related in an association, data from both forms in the association is accessible when a user works with either form in a running trial, and special controls make it easy to navigate between associated forms.

When you create a new trial, the InForm Architect application includes definitions of each type of special visit as default visits and creates a node for associations. These visits occur in a specific order in the trial definition, and you should not change this order. You can add or remove forms in these visits as you would with regular visits. However:

- You can have only one screening form and one enrollment form per trial.
- You cannot remove forms from an association after its definition has been installed in the InForm application trial database.



All visits have one or more of the following major characteristics:

- **Scheduled** or **unscheduled**—Scheduled visits occur in a fixed relationship to the beginning of a patient's enrollment in a trial. Scheduled visits start a specified number of hours after enrollment. Unscheduled visits are not on a specific timeline.
- **Optional** or **required**—The optional and required characteristics specify whether a trial can be considered complete if a visit has no data. If a visit is required, the InForm application indicates incomplete status on summary screens and on the Case Book list and Time and Events schedule.

Note: Optional visits are not currently supported.

- **Repeating**—Repeating visits are unscheduled visits in which the user determines how many visits occur and on what schedule. Site visit reports are examples of repeating visits.

Creating a new visit

This section describes how to create a new visit and define its properties.

Creating a visit definition

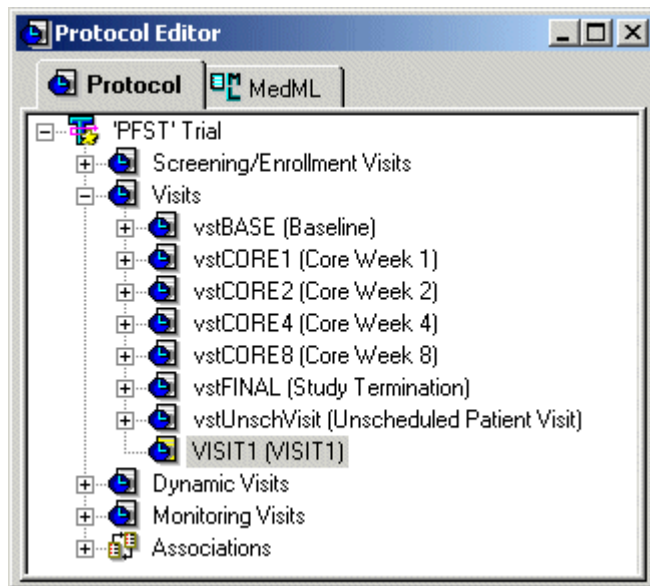
To create a new visit definition:

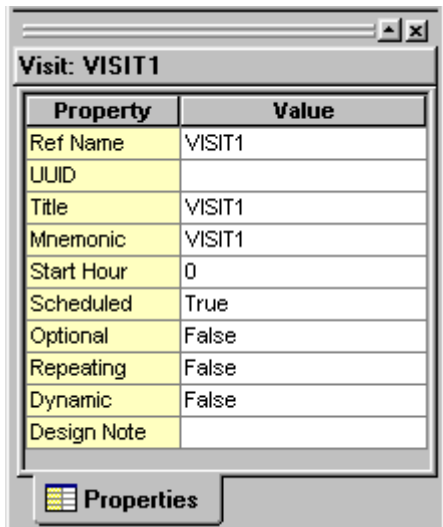
- 1 Select **Trial > Protocol Editor** to open the **Protocol Editor** window.
- 2 To add the first visit of the trial, click the trial name with the right mouse button and select **Add Visit** from the pop-up menu.

To add a subsequent visit, click the visit that you want the new visit to follow and repeat the process.

Alternatively, drag the **visit icon** from the **Object Palette** toolbar and drop it in the location you want.

The hierarchical representation of the trial is updated to show the new visit, and the Properties window displays a set of default properties.





Defining visit properties

To define the properties of a visit, select the visit icon in the **Protocol Editor** window and edit the properties for the visit in the **Properties** window:

Property	Description
RefName	<p>RefName of the component.</p> <p>REQUIRED.</p> <p>For rules about the use of RefNames, see <i>RefNames</i> (on page 89)</p> <p>Note that you cannot change the RefName of the following predefined visits: Screening, Enrollment, Reg Docs, or Visit Report.</p>
UUID	<p>String that uniquely identifies the component across all databases, trials, and machines.</p> <p>Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters. The InForm application uses UUIDs to identify trial components that require specialized processing. When you create a new trial, it comes with each of the special visit types identified in <i>About visits</i> (on page 76). Special visits have the following UUIDs, which must not be changed:</p> <ul style="list-style-type: none"> • Screening—D882CE38-0F42-11D2-A419-00A0C963E0AC • Enrollment—D882CE3A-0F42-11D2-A419-00A0C963E0AC • Site visit—PF_UUID_VISITREPORT_FORMSET • Regulatory documentation checklist—PF_UUID_REGDOCS_FORMSET <p>When creating a CRF visit, leave this property blank.</p> <p>REQUIRED. for screening, enrollment, site visit report, or regulatory documentation checklist visits.</p>

Property	Description
Title	Name of the visit as it appears in the Time and Events Schedule in the InForm application. REQUIRED.
Mnemonic	Abbreviation of the visit title. For CRF visits, this name appears at the top of a collection of forms in the visit timeline. REQUIRED.
Start Hour	Number of hours from the commencement of the trial. This number is used to: <ul style="list-style-type: none"> • Create the proposed schedule of visits shown in the Visit Calculator when a patient is enrolled. • Calculate when CRFs are expected to be completed in aging and cycle time reports. OPTIONAL
Scheduled	True or False , indicating whether the visit is scheduled. The InForm application includes scheduled visits in the sequence in which they occur on the Case Book list and Time and Events Schedule. Initially, a placeholder for unscheduled visits appears in the Case Book list and Time and Events Schedule. As users add new unscheduled visits, the individual visits are added to the Case Book List and Time and Events Schedule. True is the default. OPTIONAL
Optional	Internal use only. False is the default.
Repeating	True or False , indicating whether the visit occurs multiple times or only once. The InForm application users create repeating visits as necessary. This property is used to define unscheduled CRF visits or to define site report or regulatory document checklist visits. Repeating visits are generally unscheduled, that is, when the value of the Repeating property is True, the value of the Scheduled property is False. The default value of this property is False. OPTIONAL
Dynamic	True or False , indicating whether the visit is a dynamic visit. Dynamic visits are created automatically when patient data satisfies certain criteria that you test by attaching a rule to the relevant data item or items.
Design Note	Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only. OPTIONAL.

Renaming a visit

To rename a visit:

- 1 Open the **Protocol Editor** window.
- 2 Select the visit you want to rename.
- 3 In the **Properties** window, change the visit's **RefName**, **Title**, or **Mnemonic** properties, as appropriate.

Note: You cannot change the RefName of the following predefined visits: Screening, Enrollment, Reg Docs, or Visit Report.

Reordering visits in a trial

To change the order in which a visit occurs:

- 1 Open the **Protocol Editor** window.
- 2 Drag the icon of the visit you want to move and drop it onto the visit that you want it to follow.
To change a visit to the first visit of a trial, drop the icon onto the trial name.

Note: You can reorder only regular visits. The order of screening, enrollment, and monitoring visits is predefined, and a trial can have only one of each of these visit types. Dynamic visit order is determined by the sequence in which you create each visit. However, you can change a regular visit to dynamic or a dynamic visit to regular by dragging and dropping the visit onto the appropriate visit type.

Deleting a visit from a trial

To delete a visit:

- 1 Open the **Protocol Editor** window.
- 2 Do one of the following:
 - Select the visit you want to delete and press the **Del** key.
 - Click the visit you want to delete with the right mouse button, and from the pop-up window, choose **Delete Visit**.

The InForm Architect application prompts you to confirm that you want to proceed with the deletion.

- 3 Click **OK**.

CHAPTER 5

Defining forms

In this chapter

About forms	86
Creating a new form	94
Managing forms and visits.....	99
Defining a section	102
Defining an item	105
Defining an itemset.....	108
Using key items	113
Defining controls	119
Defining element components	134
Defining units	136
Annotating forms.....	139

About forms

Forms make up the data collection nucleus of a trial as implemented in the InForm application. This section describes the types of forms supported by the InForm Architect application and describes the components of a form.

Types of forms

An InForm application-based trial includes the following types of forms:

- **CRF**—Clinical Record Forms (CRFs) capture the data collected during a patient’s visits to an investigative site. CRFs can occur multiple times within the same visit (repeating forms) and can be associated with other forms (associated forms). The InForm Architect application supports the following types of CRFs:
 - A *regular form* contains data that is specific to the visit in which the form occurs. If the design of your trial requires you to collect the same data for multiple visits, each time the form occurs in a new visit, InForm application users can see and collect only the data for that specific visit.
 - A *common form* contains data that is cumulative from visit to visit. Each time the form occurs in a visit, it displays the data accumulated from all previous visits in which you collected that data, and InForm application users can add to the data in the current visit. Examples of forms that are often implemented as common forms are Concomitant Medication and Adverse Experience forms.
 - A *dynamic* form is generated automatically when patient data satisfies certain criteria that you test by attaching a rule to the relevant data item or items in another form. For example, if you want to collect additional data from a female patient who is pregnant, you can define a dynamic form that appears in the trial only if the response to a question determining whether the patient is pregnant is yes. For information about creating a dynamic form, see *Dynamic forms* (on page 353).
 - A *repeating* form occurs multiple times in the same visit. For example, if a protocol calls for multiple draws at intervals after administering a medication, you can capture the repeating data on multiple instances of a repeating form. Two repeating forms can be related in an *association*, which provides InForm application users with easy navigation between the forms and allows related data from each form to be displayed along with the other form. For information about creating repeating forms, see *Repeating forms* (on page 377). For information about associations, see *Associations* (on page 343).
- **Enrollment**—Forms used to capture initial eligibility information about a trial candidate and forms used to capture selection criteria used to determine whether a candidate can be enrolled in a trial are defined in the trial database as enrollment forms.
- **Regulatory Documents**—Forms used to check the status of a site’s regulatory documents.
- **Site Report**—Forms used to report on site visits.

You specify the form type when you define form properties.






Form definition hierarchy

A form definition includes several types of components, in the following hierarchy:

- **Sections** (on page 87)
- **Itemsets** (on page 88)
- **Items** (on page 88)
- **Controls** (on page 88)

Sections

Sections are modules within a form. Each section contains one or more questions. The questions can be organized as individual *items* or as groups of related items that repeat, or *itemsets*. The sample section illustrated in the following figure is made up of items.

VITAL SIGNS		Patient: HOM Patient No: PF001	
Vital Signs			
1.	<u>Weight</u>	<input type="text"/> <input type="radio"/> kg <input type="radio"/> lbs	
2.	<u>Temperature</u>	<input type="text"/> <input type="radio"/> C <input type="radio"/> F	
3.	<u>Sitting Blood Pressure</u>	<input type="text"/> / <input type="text"/> mm Hg	
4.	<u>Sitting Pulse Rate</u>	<input type="text"/> bpm	
5.	<u>Sitting Respiration Rate</u>	<input type="text"/> breaths per minute	
	BMI		

Itemsets

Itemsets are groups of questions that repeat; for example, a dosing record in which the same information is collected for each symptom.

Dosing Record Add Entry								
INSTRUCTIONS 1. Enter a new record whenever there is a change in dose 2. Enter Stop Date as the last date of a specific dose level, or the last date treatment was used for the study								
#	Start date	Stop date	Blister pack	Total # tablets per dose	# Doses per day	Doses missed	Reason	InclComments
1.a	Jun/1/2002	Jun/30/2002	572	2	3	Yes, how many? 1	Compliance	
1.b	Jul/1/2002	Jul/15/2002	729	2	3	No		

Items

Items are questions designed to collect data from a patient visit. Items are made up of text questions and data entry controls.

Pulse Rhythm:	<input type="radio"/> Regular <input type="radio"/> Irregular <input type="radio"/> Regularly Irregular
---------------	---

Item text
Data entry control

Controls

Controls are visual devices in which to enter data: for example, text boxes, selection lists, check boxes, or radio buttons. In the previous figure, the Regular, Irregular, and Regularly Irregular radio buttons are the controls for the Pulse Rhythm item.

Defining components in the hierarchy

When you create a form definition, you define each of these types of components in succession. To do this, you add each subcomponent to the definition of its parent and then refine the definition of the subcomponent by specifying its properties. For example, the sequence of steps for a form that contains one section might be:

- 1 Create a form definition.
- 2 Specify the form's properties.
- 3 Specify the properties of the default section that the InForm application includes with a new form.

- 4 Add items to the section.
- 5 Specify the properties of each item.
- 6 Add a radio control to the first item.
- 7 Specify the properties of the radio control.
- 8 Add simple controls to the radio control.
- 9 Specify the properties of each simple control.
- 10 Continue to add and define subcomponents until all controls are specified.

The sections of this chapter that describe how to create specific form components further explain the relationships between components and the properties that you must specify for each.

RefNames

RefNames are names that are used to identify component definitions. In a Form definition window, the hierarchical representation of a form and its components shows each component by RefName. RefNames are also used to uniquely identify the path to a specific control when attaching a rule definition. For more information, see *Defining rules* (on page 279).

The following rules apply to the use of RefNames:

- RefNames can have a maximum of 63 characters (except for rule RefNames, which can have a maximum of 255 characters).
- RefNames must be unique within a trial and component type. For example, no two item definitions in a trial could have the same RefName. However, a form and section can have the same RefName.
- RefNames are case sensitive.
- Once a definition containing a RefName is installed in the trial database, you cannot change the RefName. After this point, to create a new RefName, you must create a new object. The simplest way to do this is to copy or duplicate the original object. For more information, see *Cutting, copying, and pasting* (on page 51).

Oracle reserved words

The below table lists Oracle Reserved Words. These words may not be used as reenames or they will cause errors.

- ACCESS
- ADD
- ALL
- ALTER
- AND
- ANY
- AS
- ASC
- AUDIT
- BETWEEN
- BY
- CHAR
- CHECK
- CLUSTER
- COLUMN
- COMMENT
- COMPRESS
- CONNECT
- CREATE
- CURRENT
- DATE
- DECIMAL
- DEFAULT
- DELETE
- DESC
- DISTINCT
- DROP
- ELSE
- EXCLUSIVE
- EXISTS
- FILE
- FLOAT
- FOR
- FROM
- GRANT
- GROUP
- HAVING
- IDENTIFIED
- IMMEDIATE
- IN
- INCREMENT
- INDEX
- INITIAL
- INSERT
- INTEGER
- INTERSECT
- INTO
- IS
- LEVEL
- LIKE
- LOCK
- LONG
- MAXEXTENTS
- MINUS
- MLSLABEL
- MODE
- MODIFY
- NOAUDIT
- NOCOMPRESS
- NOT
- NOWAIT
- NULL
- NUMBER
- OF
- OFFLINE
- ON
- ONLINE
- OPTION
- OR
- ORDER
- PCTFREE
- PRIOR
- PRIVILEGES
- PUBLIC
- RAW
- RENAME
- RESOURCE
- REVOKE
- ROW
- ROWID
- ROWNUM
- ROWS
- SELECT
- SESSION
- SET
- SHARE
- SIZE
- SMALLINT
- START
- SUCCESSFUL
- SYNONYM
- SYSDATE
- TABLE
- THEN
- TO
- TRIGGER
- UID
- UNION
- UNIQUE
- UPDATE
- USER
- VALIDATE
- VALUES
- VARCHAR
- VARCHAR2
- VIEW
- WHENEVER
- WHERE
- WITH

UUIDs

Universal Unique Identifiers (UUIDs) ensure the unique identification of components across all servers, databases, and trials.

In most cases, assignment of a UUID property is unnecessary. However, for certain purposes, the InForm application requires the use of specific, well-known UUIDs. For example, if you want to enable users to define patient numbers, rather than allowing the InForm application to generate them automatically, you must use well-known UUIDs for the visit, form, section, item, and text control definitions that make up the specification of the patient number data entry field.

To implement some types of specialized InForm application functionality, specific UUIDs are required. For information about required UUIDs, see *UUIDs* (on page 402).

Note: The InForm Architect application automatically capitalizes UUID strings that contain lower-case characters.

Position and caption properties

When you define a component of a form, you specify its properties by editing their values in the Properties window. Many form component definitions include similar properties for specifying the appearance of the components on a form. This section describes several properties that are used frequently in form component tags.

Caption

This property enables you to specify caption text for a control. For example, you can use the Caption property to label a text box, as in the following example:

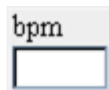


Caption Alignment

The Caption Alignment property enables you to specify the position of the caption in relation to the control. You can specify alignments of Left, Right, Top, and Bottom, as shown in the examples below:



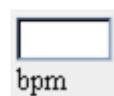
Left alignment



Top alignment



Right alignment



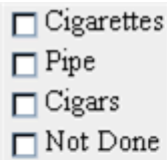
Bottom alignment

Layout

The Layout property specifies the orientation of the child controls with respect to each other within a compound control. You can specify any of the following Layout values:

- **Vertical**—Orients child controls vertically.
- **Horizontal**—Orients child controls horizontally. If the user resizes the browser window, the controls wrap to remain on screen.
- **NoWrap** (default)—Orients child controls horizontally, and does not wrap controls if the user resizes the browser window.

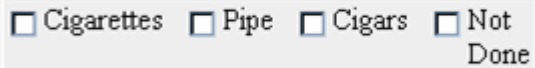
The following examples illustrate the Layout options:



Cigarettes
 Pipe
 Cigars
 Not Done

A vertical list of four checkboxes with their corresponding labels: Cigarettes, Pipe, Cigars, and Not Done.

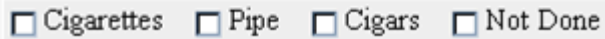
Vertical layout



Cigarettes Pipe Cigars Not Done

A horizontal list of four checkboxes with their corresponding labels: Cigarettes, Pipe, Cigars, and Not Done.

Horizontal layout



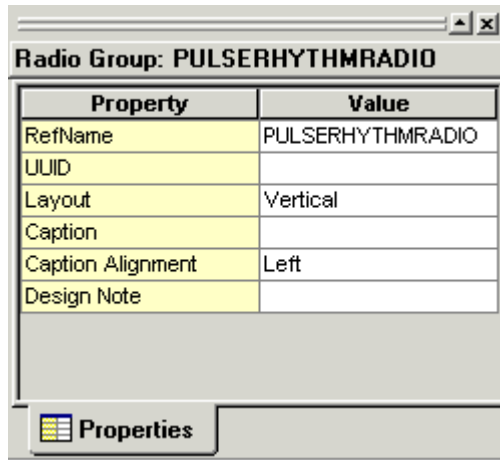
Cigarettes Pipe Cigars Not Done

A horizontal list of four checkboxes with their corresponding labels: Cigarettes, Pipe, Cigars, and Not Done.

NoWrap layout

Design notes

As you create and update forms and form components, you can include free-form notes about the components. These notes are associated with each component as properties. They are for documentation purposes only and are not displayed with the components in the InForm application. You can create design notes at the visit, form, section, itemset, item, and control level. Design notes can have a maximum of 255 characters. The following figure illustrates the Properties window for a radio group control, showing the presence of the Design Note property:



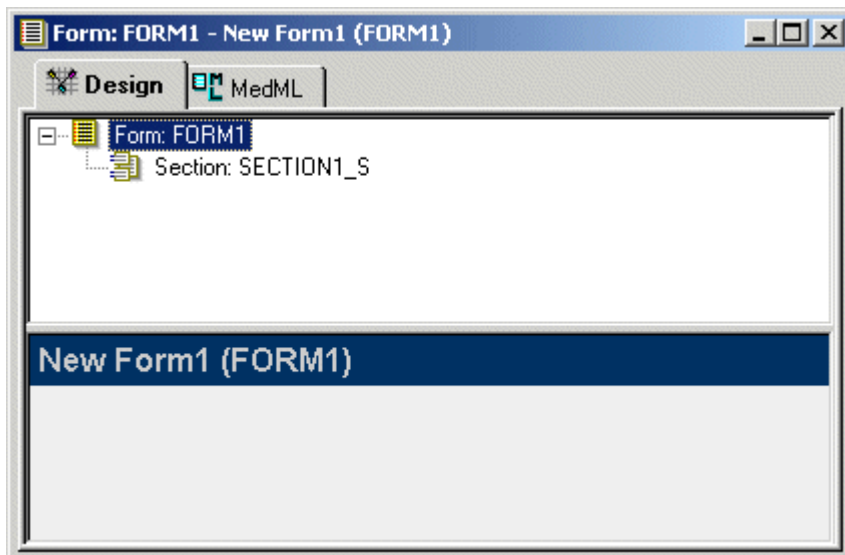
Creating a new form

Before you can add a form to a visit, you must create the form as a trial component. This section describes how to create a new form and how to define its properties. Once you perform the creation step, the form is available for inclusion in a trial visit, and you can define its properties and add its structural and data entry components.

Creating a form definition

Select **File > New > Form**. A new **Form** window appears in the application workspace. The upper pane shows a hierarchical representation of the form and its child components. When you create a new form, a default Section icon is included.

The lower pane shows a preview of the form as it will appear in the InForm application.

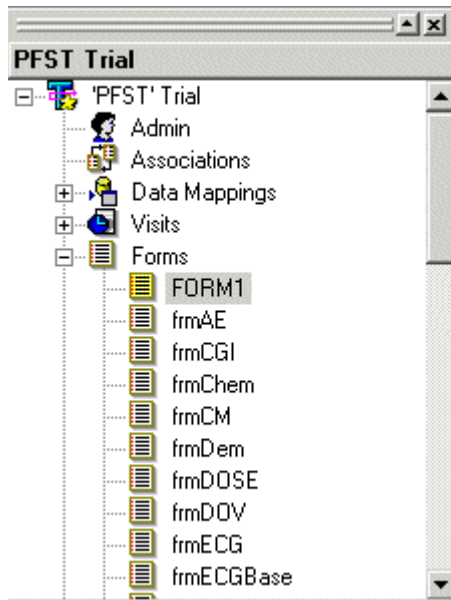


The **Properties** window shows the properties that describe a form definition.

Form: FORM1	
Property	Value
RefName	FORM1
Form Type	CRF
UUID	
Title	New Form1
Mnemonic	FORM1
Note	
Question Width	50
Control Width	50
Common Form	False
Repeating Form	False
Design Note	

Properties

The **Trial Objects** window is updated to show the addition of the new form object.



Defining form properties

To define the properties of a form, select the form icon in the **Form** definition window and edit the form's properties in the **Properties** window. As you specify properties that have an effect on the appearance of the form, the preview pane of the **New Form** window is updated to show the change.

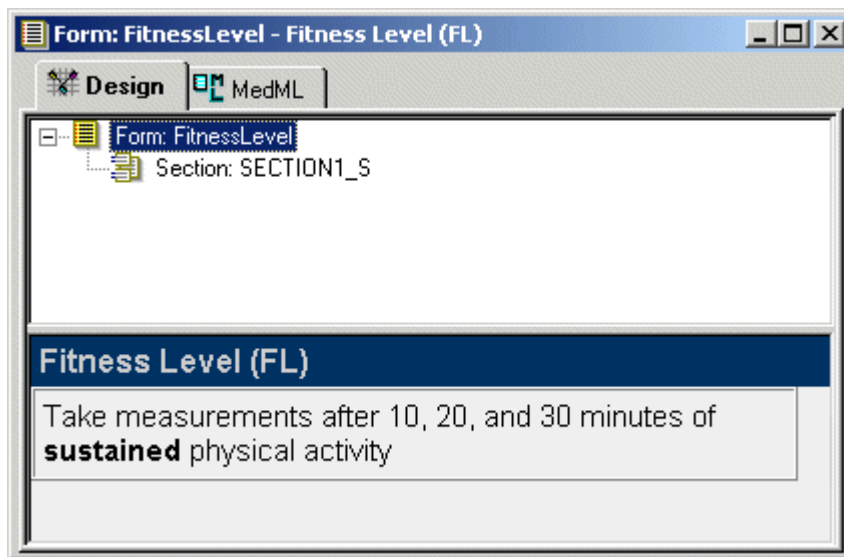
The following table describes the properties that you specify to create a form definition:

Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
Form Type	CRF. READ ONLY. Note that when you create a form in one of the specialized visit types (Screening/Enrollment or Monitoring), the InForm Architect application automatically assigns the appropriate Form Type property when you associate the form with the visit. These specialized form types are: <ul style="list-style-type: none"> • Enrollment—Used for Screening and Enrollment forms • Visit Reports—Used for Visit Report form • Regulatory Documents—Used for Regulatory Documentation forms

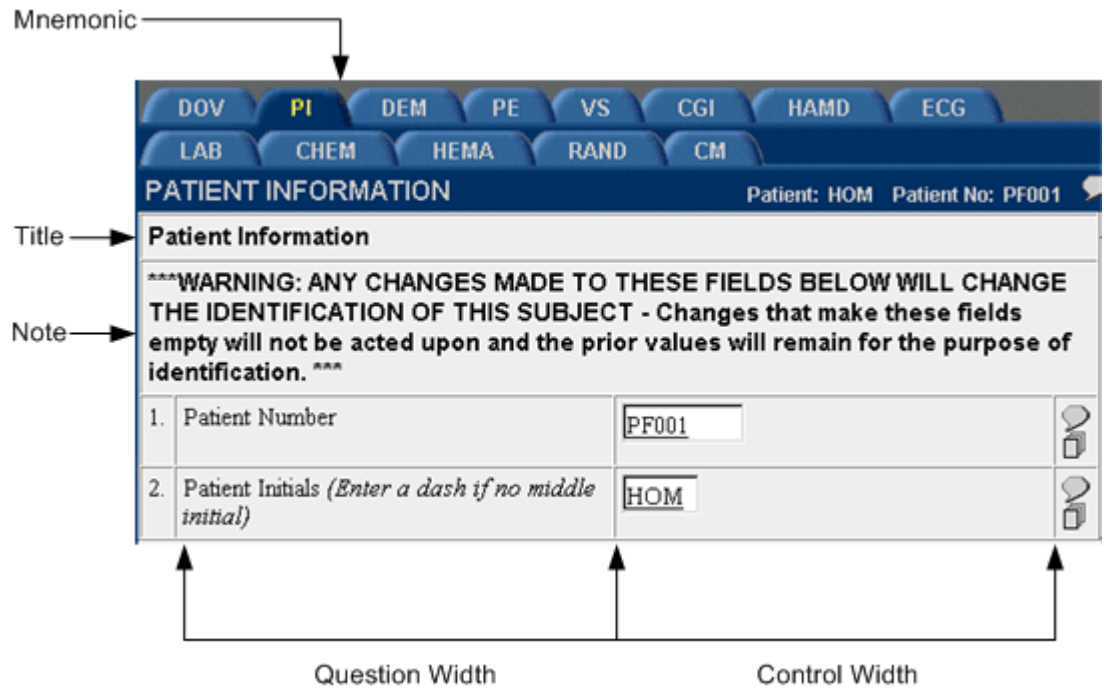
Property	Description
UUID	<p>String that uniquely identifies the component across all databases, trials, and machines.</p> <p>Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters.</p> <p>REQUIRED for certain types of forms, otherwise not necessary.</p> <p>For information about required UUIDs, see <i>UUIDs</i> (on page 402).</p>
Title	<p>Specifies the text of the form heading as it appears in the form's title bar online.</p> <p>REQUIRED.</p>
Mnemonic	<p>Specifies a short name or abbreviation for the form. This name appears on the visit navigation tab used to go to a specific form.</p> <p>REQUIRED.</p>
Note	<p>Specifies the text of a note to appear immediately below the Form heading. You can include HTML formatting characters to specify special fonts or emphasis. For a list of the HTML formatting tags and special characters supported by the InForm application, see the MedML online help.</p> <p>OPTIONAL.</p>
Question Width and Control Width	<p>Specify the percentage of the available screen occupied by item questions and by item controls. The percentages must total 100. The default percentages are both 50.</p> <p>OPTIONAL.</p>
Common Form	<p>True or False, specifying whether the form is a common, cumulative form. False is the default.</p> <p>OPTIONAL.</p>
Repeating Form	<p>True or False, specifying whether the form can be used multiple times within a visit. False is the default. For information about creating repeating forms, see <i>Repeating forms</i> (on page 377).</p> <p>OPTIONAL.</p>
Design Note	<p>Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only.</p> <p>OPTIONAL.</p>

Property	Description
Unique Key	<p>True or False, specifying whether items specified as Key Items for a repeating form must be unique to each instance of the form. This property applies only to repeating forms. False is the default.</p> <p>If Key Items are defined as unique keys and two instances of a form are submitted in the same visit with any of the same Key Item values, the InForm application rejects the input of the second instance. For information about defining Key Items, see <i>Using key items</i> (on page 113).</p> <p>OPTIONAL.</p>

The following figure shows how the new form window looks after properties are defined. The form is ready for inclusion of additional sections, items, and controls, as necessary.



The following figure illustrates how the properties of a form are used in the display of a CRF window in the InForm application.



Adding form subcomponents

A form definition must include at least one section. For information about adding a section to a form, see *Defining a section* (on page 102).

Managing forms and visits

The InForm Architect application enables you to add, reorder, or remove a form in a visit. This section describes how to perform these activities in the Protocol Editor window.

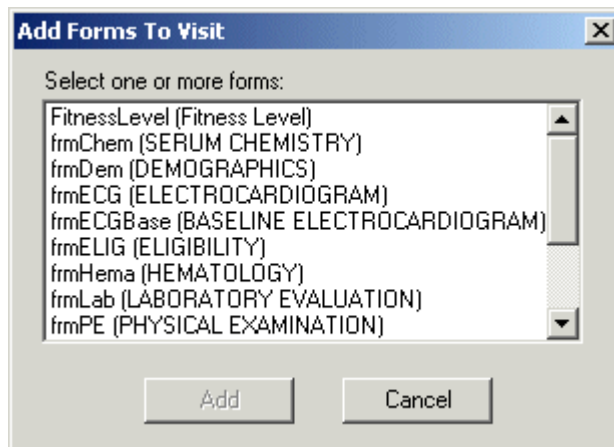
Adding a form to a visit

To add a form to a visit, right-click the visit in the Protocol Editor or drag and drop the form onto the visit.

Using the pop-up menu in the Protocol Editor

- 1 In the **Protocol Editor** window, select the visit to which you want to add a form. If the visit already includes one or more forms, open the visit by clicking its plus icon, and select the form after which you want the new form to be added.
- 2 Click the visit or form icon, or its description, with the right mouse button, and select **Add Form** from the pop-up menu.

The **Add Forms to Visit** dialog box appears, showing a list of defined forms that have not already been associated with the visit. Special forms such as Screening and Enrollment are not included in the list.



- 3 From the **Add Forms to Visit** dialog box, select the form you want to add.
- 4 Click **Add**.

Dragging and dropping the form

An alternative method for associating a form with a visit is to drag the form from the Trial Objects window. Drop the form in the Protocol Editor window on the appropriate visit or on the form you want it to follow within a visit.

Adding special forms to visits

The following visits can have only one form:

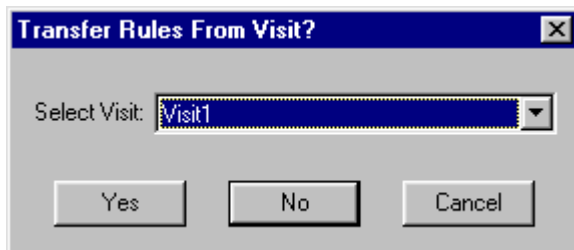
- Screening
- Enrollment
- Visit Report
- Reg Docs

If you attempt to add a second form to one of these visits, the InForm Architect application removes the current form and replaces it with the selected form after prompting you to confirm that you want to replace the current form.

Note: These specialized forms require predefined UUIDs. For information about required UUIDs, see *UUIDs* (on page 402). When you add a special form to its visit, the InForm Architect application assigns the appropriate UUID to the form.

Transferring rule contexts

If a form that you add to a visit has rules defined for any of its items, the InForm Architect application gives you the option of creating rule contexts for the form in the new visit based on the form's existing rule context definitions for the form. When you add the form to the visit, the following dialog box appears:

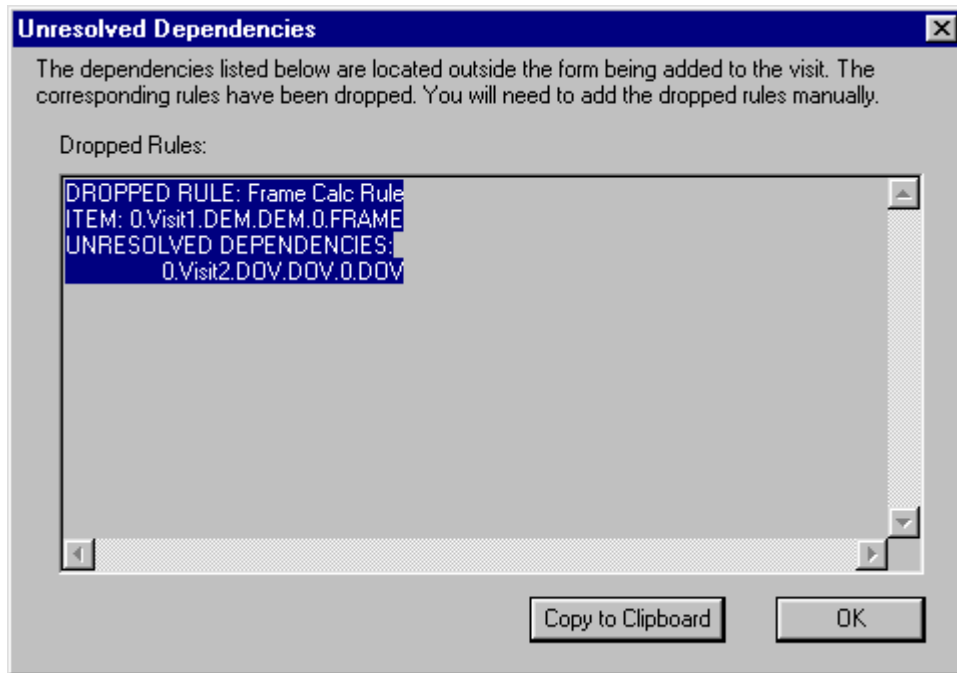


To generate new rule contexts:

- 1 In the **Select Visit** list, select the visit on which you want to base the contexts.
- 2 Click **Yes**.

The InForm Architect application copies the rule contexts from the existing visit and associates them with the form in the current visit, changing their names by appending the new visit name to the end of the original context name.

If any context references cannot be resolved—for example, if a context includes a dependency on an item outside the form—the InForm Architect application displays the Unresolved Dependencies dialog box. This message dialog box indicates which references could not be resolved and instructs you to make the associations manually:



For convenience, you can copy the message to the Microsoft Windows Clipboard by clicking the Copy to Clipboard button. You can add any context definitions and dependency that the InForm Architect application was unable to create automatically by following the procedure described in *Defining additional rule dependencies* (on page 314).

Reordering a form in a visit

To reorder a form within a visit, drag its icon onto the form after which you want it to appear. To move it to the first position in the visit, drag the icon onto the visit icon.

Removing a form from a visit

To remove a form from a visit, do either of the following:


- Click the form icon with the right mouse button, and select **Remove Form** from the pop-up menu.
- Select the form and click the **Delete** key.

Defining a section

A *section* is a module of a form. Each form must include at least one section. When you create a new form, the InForm Architect application provides a default section icon in the Form window workspace. This section of the manual describes how to create a form section and specify its properties and how to include an existing section in the definition of a form.

Creating a new section

You can create a new section only in the context of the form in which you want to include it. To define the properties of the default section provided when you create a new form, follow the steps described in *Defining section properties* (on page 102). If you are creating additional sections, first do the following:

- 1 Open the form in which you want to include the section: double-click its icon in the **Protocol Editor** window or **Trial Objects** window.
- 2 Drag the section icon  in the **Object Palette** toolbar into the open **Form** definition window. To order the section first in the form, drop the icon onto the **Form** node. To create a subsequent section, drop the icon onto the section that you want the new section to follow.

Defining section properties

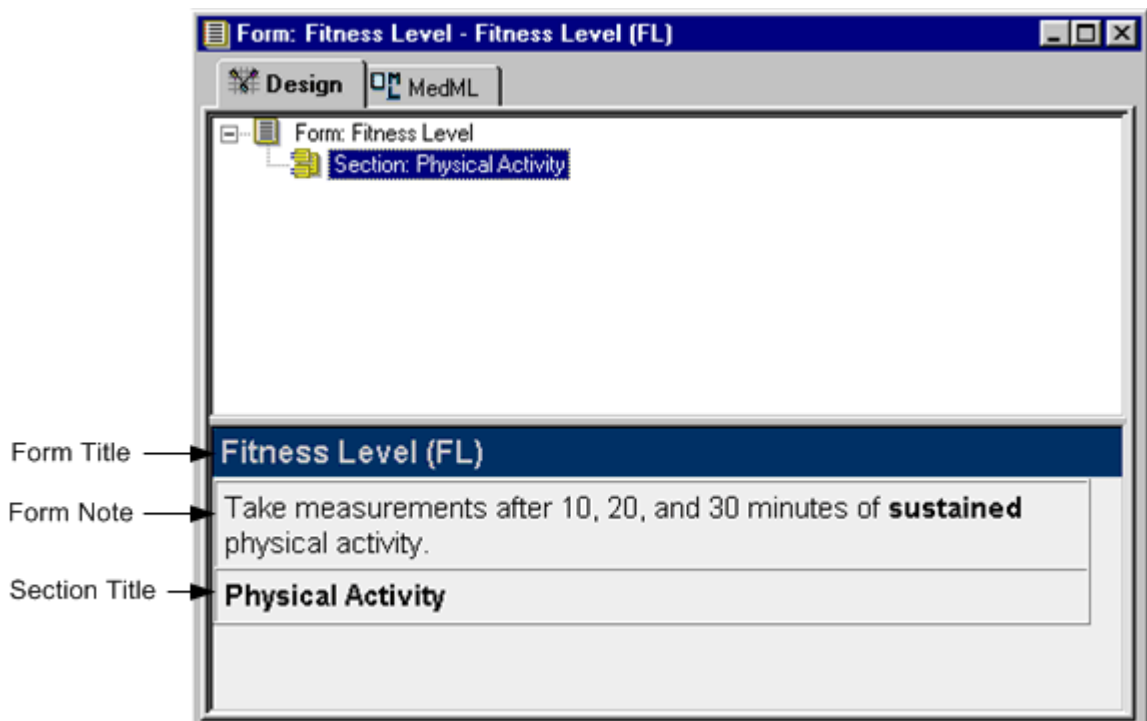
To define the properties of a section, select the section icon in the Form definition window and edit the section's properties in the Properties window. As you specify properties that have an effect on the appearance of the form, the preview pane of the New Form window is updated to show the change.

The following table describes the properties that you specify to create a section definition.

Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
UUID	String that uniquely identifies the component across all databases, trials, and machines. Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters. REQUIRED for certain types of sections, otherwise not necessary. For information about required UUIDs, see <i>UUIDs</i> (on page 402).
Title	Specifies the text of the section heading as it appears in the section's title bar online. OPTIONAL.

Property	Description
Note	Specifies the text of a note to appear immediately below the Section title. You can include HTML formatting characters to specify special fonts or emphasis. OPTIONAL.
Design Note	Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only. OPTIONAL.

The following figure shows how the new form window looks after a section and its properties are defined. The section is ready for inclusion of items or an itemset and controls, as necessary.



Adding section subcomponents

A section definition must include at least one item or itemset. The section can contain either one or more items or a single itemset. It cannot contain both, and it cannot contain more than one itemset.

To add items or an itemset to a section, follow the instructions in *Defining an item* (on page 105) or *Defining an itemset* (on page 108).

Including an existing section definition in a form


To include a previously defined section in a form:

- 1 Open the form in which you want to include the section.
- 2 Open the **Sections** node in the **Trial Objects** window.
- 3 Drag the section you want to include onto the form icon in the **Form** definition window.

Defining an item

An *item* is a question designed to collect data from a patient visit. Items are made up of text questions and data entry controls. A section must have at least one item. This section describes how to create an item and specify its properties and how to include an existing item in the definition of a section.

Creating a new item

- 1 Open the form in which you want to include the item: double-click its icon in the **Protocol Editor** window or **Trial Objects** window.
- 2 Select the section in which you want to create the item.
- 3 Drag the item icon  in the **Object Palette** toolbar into the open **Form** definition window. To order the item first in the section, drop the icon onto the section icon. To create a subsequent item, drop the icon onto the item that you want the new item to follow.

Defining item properties

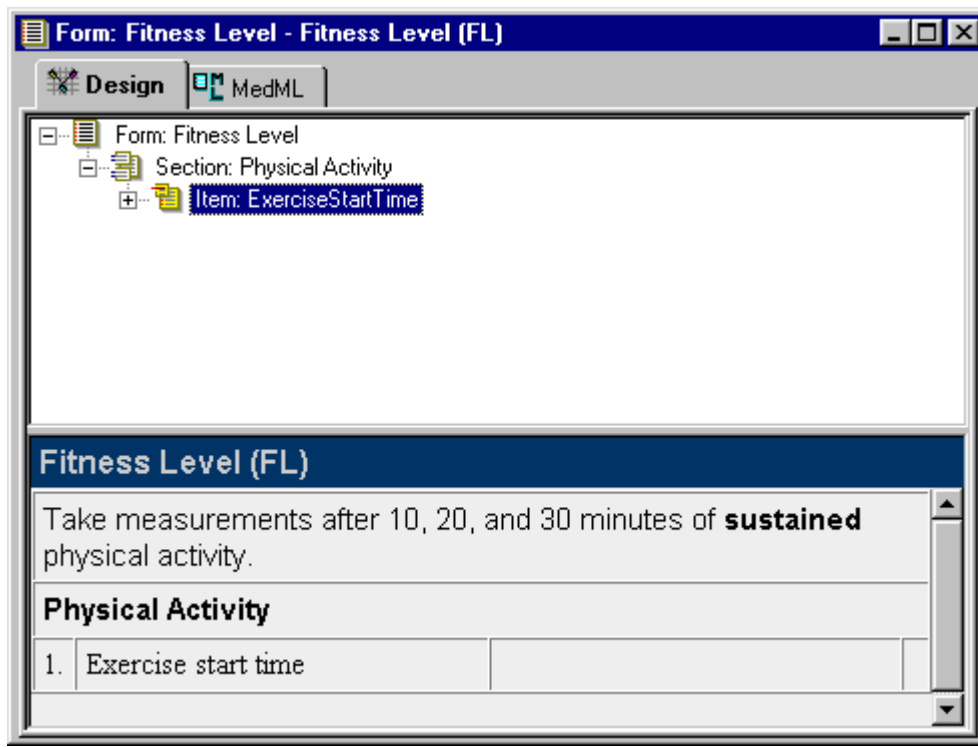
To define the properties of an item, select the item icon in the **Form** definition window and edit the properties of the item in the **Properties** window. As you specify properties that have an effect on the appearance of the form, the preview pane of the **New Form** window is updated to show the change.

The following table describes the properties that you specify to create an item definition:

Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
UUID	String that uniquely identifies the component across all databases, trials, and machines. Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters. REQUIRED for certain types of items, otherwise not necessary. For information about required UUIDs, see <i>UUIDs</i> (on page 402).
Question	Text of the user prompt. REQUIRED.

Property	Description
Itemset Column Header	<p>Specifies the following:</p> <ul style="list-style-type: none"> • A label to use as the Itemset column heading, if you are defining an item to be included in an itemset definition. • A default short name for the data item in the clinical reporting model. <p>When you specify an Itemset Column Header property, InForm application uses the Question text on the Add Entry page where users enter itemset details and uses the Itemset Column Header text on the CRF page in the itemset column headings.</p> <p>Note: If you are designing a trial with the InForm Architect application, do not use Subject Number as an itemset column header name. It will cause an error when you create the clinical model for the trial.</p> <p>OPTIONAL.</p>
Calculated	<p>Indicates whether the item contains a Calculated Control. Select either True or False (the default). If you specify that the Calculated property is True, the item is unnumbered, to set it off from items that require user entry.</p> <p>OPTIONAL.</p>
Item Required	<p>Indicates whether the item is required for data entry on the form to be complete. Select either True (the default) or False. If you select False, the item has a gray background on the CRF after submission to indicate that entry is not required.</p> <p>OPTIONAL.</p>
SV Required	<p>Indicates whether the item requires source verification. Select either True (the default) or False. If you select False, the item has a gray background on the SV view of the CRF to indicate that source verification is not required.</p> <p>OPTIONAL.</p>
Display Override	<p>Indicates whether the item is visible and editable. Select Read-Only to make the item visible but not editable. Select Editable to make the item always visible and editable. Select Hidden to make the item invisible on a form. The item display overrides can themselves be overridden through item-level overrides in rights group permissions. For details, see <i>Assigning display overrides to rights groups</i> (on page 440).</p> <p>REQUIRED.</p>
Design Note	<p>Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only.</p> <p>OPTIONAL.</p>

The following figure shows how the New Form window looks after an item and its properties are defined. The item is ready for inclusion of controls, as necessary.



Adding controls to an item

An item must include one control definition. A control definition can consist of:

- A single data entry control such as a text box
- A group consisting of multiple controls of the same type; for example, a set of radio buttons
- A group consisting of multiple controls of different types; for example, a set of radio buttons that includes a pulldown control, a text box control, and a simple radio control

The InForm Architect application supports several types of data entry controls. For information on how to define controls and include them in an item, see *Defining controls* (on page 119).

Including an existing item definition in a section

To include a previously defined item in a section:

- 1 Open the form in which you want to include the item.
- 2 Open the **Items** node in the **Trial Objects** window.
- 3 Drag the item you want to include onto the section icon in the **Form** definition window. To order the item first in the section, drop the icon onto the section icon. To create a subsequent item, drop the icon onto the item that you want the new item to follow.

Defining an itemset


An *itemset* is a group of related items that appears repeatedly in tabular form within a section.

The following rules apply to creating itemsets:

- Only one itemset definition can appear in a section.
- Regular, non-repeating item definitions cannot appear in a section that includes an itemset.

Creating a new itemset

To create an itemset:

- 1 Open the form in which you want to include the itemset: double-click its icon in the **Protocol Editor** window or **Trial Objects** window.
- 2 Select the section in which you want to create the itemset.
- 3 Drag the itemset icon  in the **Object Palette** toolbar into the open **Form** definition window. Drop the icon onto the section icon.

Defining itemset properties

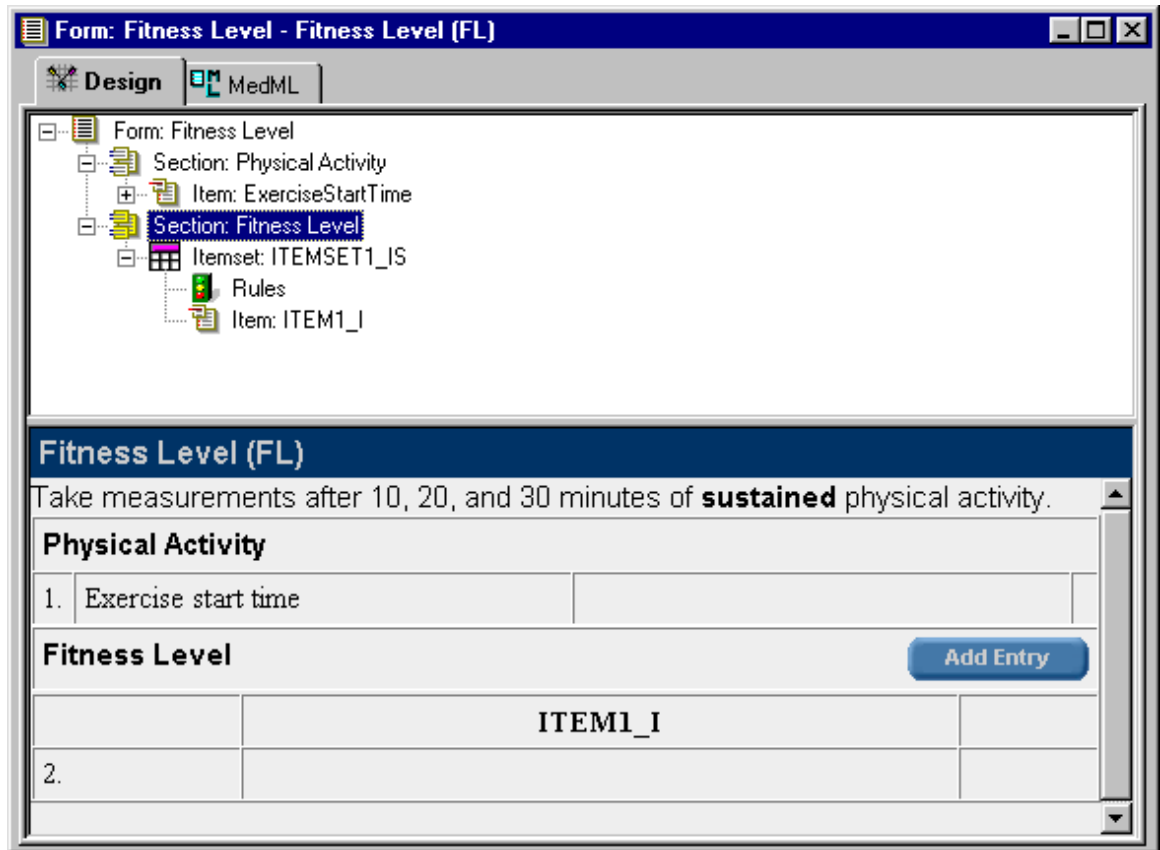
To define the properties of an itemset, select the itemset icon in the **Form** definition window and edit the itemset's properties in the **Properties** window. As you specify properties that have an effect on the appearance of the form, the preview pane of the **New Form** window is updated to show the change.

The following table describes the properties that you specify to create an itemset definition:

Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
UUID	String that uniquely identifies the component across all databases, trials, and machines. Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters. OPTIONAL, not needed for itemsets.
Item Required	Indicates whether the itemset must be completely filled in for data entry on the form to be complete. Select either True (the default) or False. If you select False, the itemset has a gray background on the CRF after submission to indicate that entry is not required. OPTIONAL.

Property	Description
SV Required	<p>Indicates whether the itemset requires source verification. Select either True (the default) or False. If you select False, the itemset has a gray background on the SV view of the CRF to indicate that source verification is not required.</p> <p>OPTIONAL.</p>
Display Override	<p>Indicates whether the itemset is visible and the items within it editable. Select Read-Only to make the itemset visible but make the items within it not editable. Select Editable to make the itemset always visible and the items within it editable. Select Hidden to make the itemset invisible on a form. The item display overrides can themselves be overridden through item-level overrides in rights group permissions. For details, see <i>Assigning display overrides to rights groups</i> (on page 440).</p> <p>REQUIRED.</p> <p>Note: To make some items within an itemset hidden, you must make the whole itemset editable, and then specify the individual items as hidden or non-editable.</p>
Design Note	<p>Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only.</p> <p>OPTIONAL.</p>
Unique Key	<p>True or False, specifying whether items specified as Key Items for an itemset must be unique to each row of the itemset. False is the default.</p> <p>If Key Items are defined as unique keys and two rows are submitted with any of the same Key Item values, the InForm application rejects the input of the second row. For information about defining Key Items, see <i>Using key items</i> (on page 113).</p> <p>OPTIONAL.</p>

The following figure shows how the new form window looks after an itemset and its properties are defined. The itemset includes one default item definition and is ready for inclusion of additional items, as necessary.



Adding items to an itemset

An itemset must include at least one item definition. The InForm Architect application automatically includes a default item icon in the hierarchical representation of the itemset in the Form window.

To add items to an itemset, use the process described in *Defining an item* (on page 105) to create the definition of each item you want to include in the itemset:

- To define the properties of the default item provided with the itemset definition, follow the instructions in *Defining item properties* (on page 105).
- To create additional item definitions, first follow the instructions in *Creating a new item* (on page 105). Instead of dropping item icons onto a section definition, drop them onto the icon of the itemset.

Viewing the items in an itemset

The Form window preview pane for a form that includes an itemset definition shows the itemset as it appears in the InForm application. Each item in the itemset is represented only by its column heading label. The InForm Architect application enables you to see each item with its controls when you select an itemset in the tree view presented in the top pane of the form window. To view a preview that shows individual itemset items:

- 1 Open the form in which you want to view itemset details.
- 2 In the top pane of the **Form** window, select the itemset. The preview pane changes to the **Add Entry** view of the itemset, as it would appear in the InForm application when you click the **Add Entry** button, showing the format of each item in the itemset.

Form: frmDOSE - DOSING RECORD (DOSE)

Design MedML

Form: frmDOSE
 Section: sctDOSE
 Itemset: itsDOSE

DOSING RECORD (DOSE)

START THE DOSING RECORD WITH THE DAY OF THE LAST VISIT AND END WITH THE DAY OF PRESENT VISIT (Account for all changes in dose, missed doses, and days off therapy on known dates)

Dosing Record [Add Entry](#)

INSTRUCTIONS

1. Enter a new record whenever there is a change in dose
2. Enter Stop Date as the last date of a specific dose level, or the last date treatment was used for the study

	Start date	Stop date	Blister pack	Total # tablets per dose	# Doses per day	Doses missed	Reason	InclComments
1.								

Form: frmDOSE - DOSING RECORD (DOSE)

Design MedML

Form: frmDOSE
 Section: sctDOSE
 Itemset: itsDOSE

DOSING RECORD (DOSE)

Dosing Record Entry

INSTRUCTIONS

1. Enter a new record whenever there is a change in dose
2. Enter Stop Date as the last date of a specific dose level, or the last date treatment was used for the study

Start date	<input type="text"/> / <input type="text"/> / <input type="text"/>
Stop date	<input type="text"/> / <input type="text"/> / <input type="text"/>
Blister pack number	<input type="text"/>
Total # tablets per dose	<input type="text"/>
# doses per day	<input type="text"/>

Including an existing itemset definition in a section

To include a previously defined itemset in a section:

1. Open the form in which you want to include the itemset.
2. Open the **Itemsets** node in the **Trial Objects** window.
3. Drag the itemset you want to include onto the section icon in the **Form** definition window.

Using key items

When designing a repeating form or an itemset, consider whether you want to use one or more key items to:

- Enhance repeating form navigation.
- Check for data uniqueness in specified items.

Enhancing navigation in repeating forms

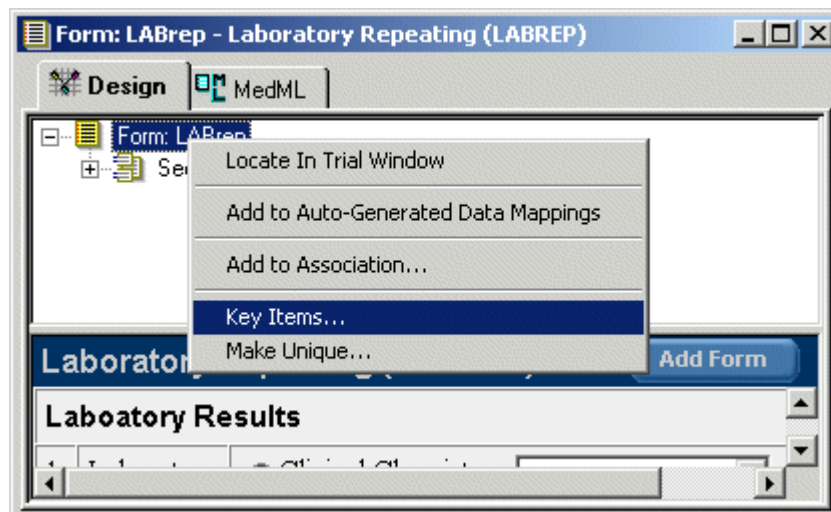
To make it easier for users to navigate through instances of a repeating form, you can identify one or more items in the form as *key items*. When you identify one or more key items, the InForm application displays navigation controls in the form header, along with a pulldown list of the key item values. Users can navigate to a specific instance of the form by selecting the instance based on the key item values.

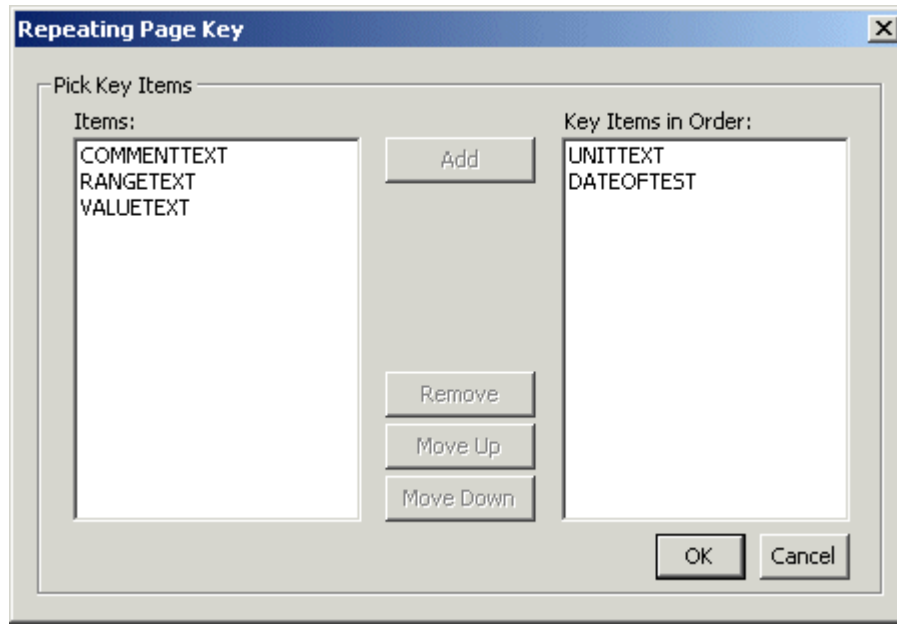
The screenshot shows the InForm application interface. At the top, there are tabs for 'DOV', 'VS', 'DOSE', 'CGI', 'HAMD', 'CM', 'AE', and 'LAB'. Below these is a 'DEM' tab. The main header displays 'PRIOR and CONCOMITANT MEDICATION (2 of 2)' with navigation arrows and a dropdown menu showing '2. Apr/10/2002 ; levoxy'. To the right of the header are buttons for 'New', 'Patient: HOM', and 'Patient No: PF001'. Below the header is a text area with the instruction 'Record all medication taken within 8 weeks prior to study start'. The main form area is titled 'Concomitant Medication' and contains three rows of data:

1. Medication (generic name preferred)	levoxy
2. Total Daily Dose with Units	50 mcg
3. Date Started	Apr / 10 / 2002

To introduce repeating form navigation with key items:

1. Open the repeating form for which you want to specify key items.
2. In the **Form editor** window, right-click the form name and select **Key Items** from the popup menu.



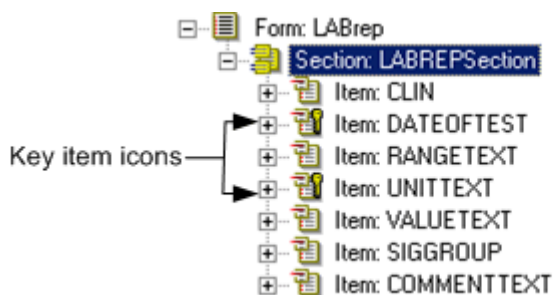


- 3 Click the desired items in the **Items** pane; then click the **Add** button to move the items to the **Key Items in Order** pane.

Note: Only items with the following types of controls can be key items and are included in the selection list: date time controls, pulldown lists, radio groups, and text boxes. Items defined as radio groups cannot be key items if they contain any type of nested compound controls: check box controls, group controls or other radio groups. Additionally, key items defined for an itemset cannot be key items in a repeating form that includes the itemset. These items are excluded from the selection list.

- 4 You can reorder the key items in the **Key Items in Order** pane by selecting one then clicking the **Move Up** or **Move Down** key. The order in which the items appear in the **Key Items in Order** pane determines the order in which they appear in the navigational pulldown list when the form is displayed during a trial.
- 5 You can remove key items from the **Key Items in Order** pane by selecting one then clicking the **Remove** button.
- 6 Click **OK** when you are done.

When you expand the form node in the **Form** editor window, note the key icon to the left of the items you designated as key items.



Checking for data uniqueness

When you select key items for a repeating form or an itemset, you can specify that the key items create a unique key, either separately or together. When key items are designated unique keys, if a user attempts to submit a repeating form or itemset row in which a unique key item value already exists, the InForm application accepts or rejects the new form instance or itemset row according to the following rules:

- When key items are individually unique, the InForm application evaluates each key item value separately to determine if it is different from previously entered values. If a new repeating form instance or itemset row contains a value that matches a previously entered unique key item, the instance or row is rejected. The following table gives an example in a repeating form instance or itemset where the DATE and PROCEDURE items have been designated separate unique keys:

DATE value	PROCEDURE value	Result
May 1, 2002	Hematocrit	Instance/row added
May 1, 2002	Hemoglobin	Instance/row rejected
June 14, 2002	Hematocrit	Instance/row rejected
June 15, 2002	Platelet count	Instance/row added

- When key items are unique together, the InForm application evaluates the key items in combination to determine whether the combination of items is different from a previously entered combination. If a new repeating form instance or itemset row contains a set of values that match a previously entered combination of key items, the instance or row is rejected. The following table shows how the DATE and PROCEDURE items are evaluated when they are designated a unique key combination:

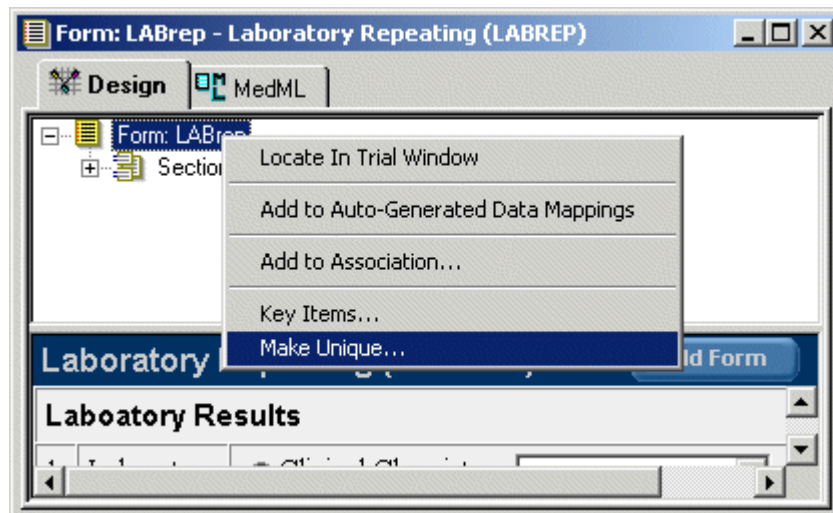
DATE value	PROCEDURE value	Result
May 1, 2002	Hematocrit	Instance/row added
May 1, 2002	Hemoglobin	Instance/row added
June 14, 2002	Hematocrit	Instance/row added
May 1, 2002	Hemoglobin	Instance/row rejected

Specifying separate unique keys

Specifying separate unique keys for a repeating form or an itemset involves the same process. The illustrations that follow use a repeating form as an example.

To specify unique keys:

- Identify the key items for the form or itemset, as described in *Enhancing navigation in repeating forms* (on page 113) and *Specifying key items for an itemset* (on page 116).
- In the **Form** editor window, right-click the form or itemset icon and select **Make Unique** from the popup menu.

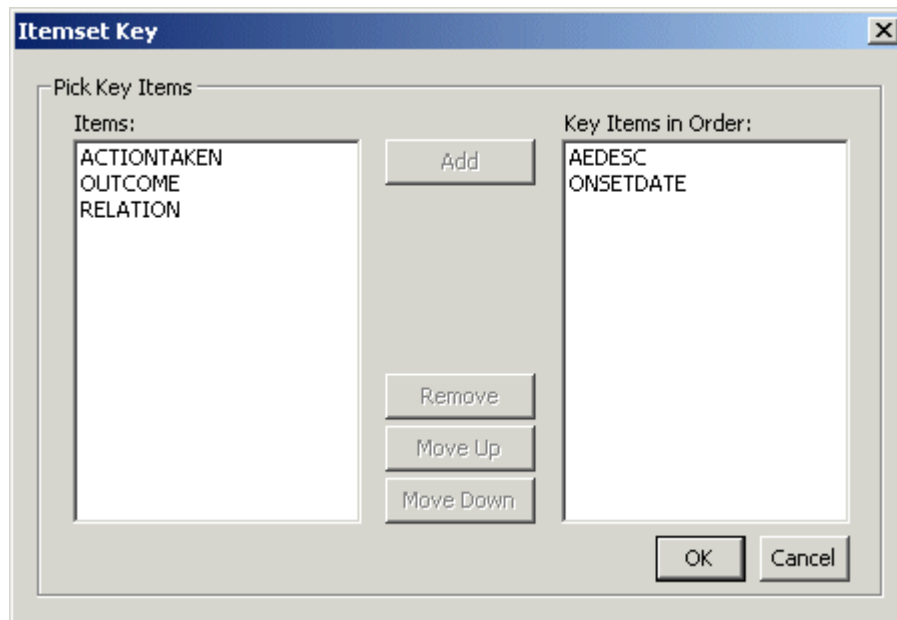
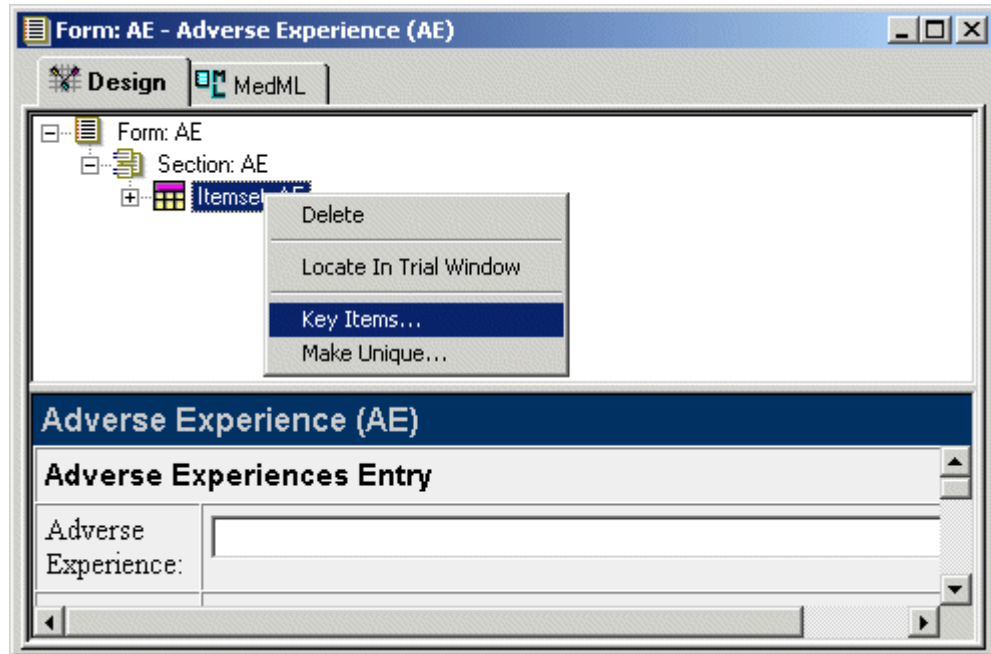


- 3 In the **Make Unique** dialog box, select the key items that you want to serve as unique keys when users add new form instances or new itemset rows.
- 4 Click **OK**.

Specifying key items for an itemset

To specify key items for an itemset:

- 1 Open the form containing the itemset for which you want to specify key items.
- 2 In the **Form** editor window, expand the nodes until the itemset is visible.
- 3 Right-click the itemset icon and select **Key Items** from the popup menu.



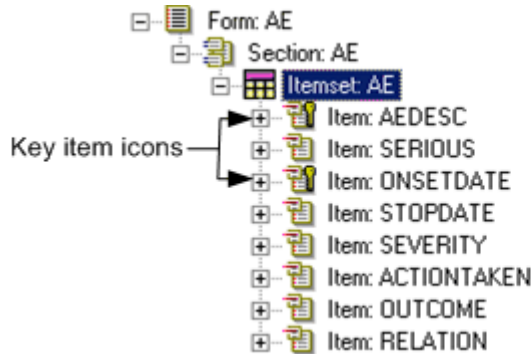
- 4 Click the desired items in the Items pane; then click the **Add** button to move the items to the **Key Items in Order** pane.

Note: Only items with the following types of controls can be key items and are included in the selection list: date time controls, pulldown lists, radio groups, and text boxes. Items defined as radio groups cannot be key items if they contain any type of nested compound controls: check box controls, group controls or other radio groups. Additionally, key items defined for an itemset cannot be key items in a repeating form that includes the itemset. These items are excluded from the selection list.

- 5 You can reorder the key items in the **Key Items in Order** pane by selecting one then clicking the **Move Up** or **Move Down** key.

- 6 You can remove key items from the **Key Items in Order** pane by selecting one then clicking the **Remove** button.
- 7 Click **OK** when you are done.

When you expand the itemset node in the **Form** editor window, note the key icon to the left of the items you designated as key items.



Specifying unique key combinations

To specify that the InForm application should evaluate a group of unique keys in combination:

- 1 Identify a set of key items that are unique keys, as described in *Specifying separate unique keys* (on page 115).
- 2 Set the value of the **Unique Key** property of the repeating form or itemset to True:
 - a In the **Form** window, select the form or itemset node, as appropriate.
 - b In the **Properties** window, select **True** as the value of the **Unique Key** property.

Defining controls

An item must include one control definition. A control definition can consist of:

- A single data entry control such as a text box
- A group consisting of multiple controls of the same type; for example, a set of radio buttons
- A group consisting of multiple controls of different types; for example, a set of radio buttons that includes a pulldown control, a text box control, and a simple radio control

The InForm Architect application supports the following types of data entry controls:

Control	Example	Description
Group	<input type="text"/> / <input type="text"/> mm Hg	A set of controls. Use a group control to combine multiple controls of different types or to create a compound control that consists of more than one text box control.
Radio group	<input type="radio"/> Male <input type="radio"/> Female	A set of controls from which a user must select only one value.
Checkbox group	<input type="checkbox"/> Cigarettes <input type="checkbox"/> Pipe <input type="checkbox"/> Cigars	A set of controls in which a user can select as many values as are applicable.
Pulldown control	<input type="text"/> ▾	A list of values from which a user can select only one. Pulldown controls can include a control in which the user selects units.
Text box control	<input type="text"/> <input type="radio"/> lbs <input type="radio"/> kg	A box in which a user enters a value explicitly. Text box controls can include a control in which the user selects units. This unit control can be presented as a radio control, a pulldown list, or a simple element.
Date/Time control	<input type="text"/> ▾ / <input type="text"/> ▾ / <input type="text"/> ▾ <input type="text"/> ▾ : <input type="text"/> ▾ 24-hour clock <input type="text"/> ▾ / <input type="text"/> ▾ / <input type="text"/> ▾ <input type="text"/> : <input type="text"/> 24-hour clock [hh:mm]	<p>A set of controls in which a user selects date or time values. A Date/Time control can include pulldown controls for day, month, year, hour, minute, and second.</p> <p>You can also define the time portion of a Date/Time control as a set of text controls.</p>

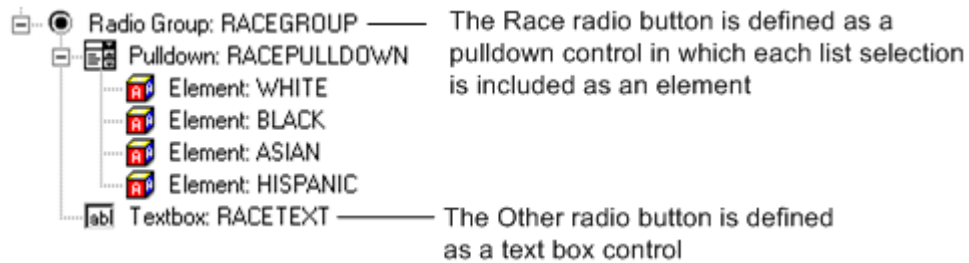
Control	Example	Description
Calculated control	Frame	A control whose value is determined by a calculation based on the value entered in one or more other controls. An item containing a calculated control is unnumbered and read-only. A calculated control can include a unit component.
Simple control	Premature Ectopic Atrial Beats	A wrapper that contains an element definition and is used to include the element definition in a group, radio group, or check box group control. A simple control can include a unit component.

Nesting controls

The InForm Architect application enables you to create compound controls that are composed of nested groups of different types of controls. For example, the control illustrated in the following figure contains two subcontrols that are nested within the overall radio group:

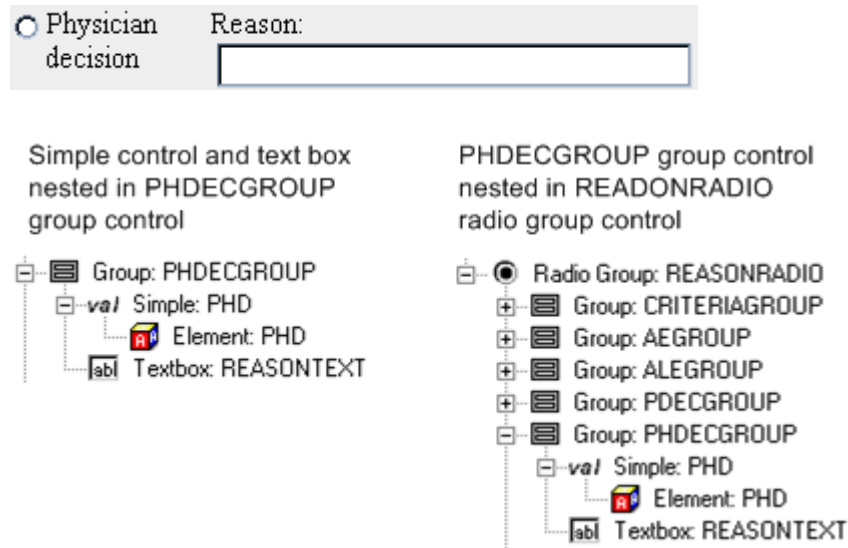


The component structure of this radio control is illustrated in the following figure:



In general, you nest control definitions whenever you define a control structure that has more than one component; for example, when you define a set of radio buttons or check boxes, each radio button or check box is defined as a separate control nested within the radio group or check box group.

Within a compound control structure, you need to create another level of nesting when you want the radio button or check box and the control that goes with it to have different captions. For example, in the following control, the Physician decision radio button label and the text box for entering the data each have a caption and are defined as separate controls within a group control called PHDECGROUP. The PHDECGROUP is in turn nested within a radio group.



Note: When defining compound controls, note that the InForm application supports a maximum of five levels of nesting. Although five levels are supported, as a design practice, you should attempt to minimize the number of nested levels to help performance.

Creating a new control—general instructions

The method for creating a new control is similar for each type of control:

- 1 Open the form in which you want to include the control.
- 2 Drag the appropriate **Object Palette** toolbar icon into the open **Form** definition window. Drop the icon onto the item or control in which to include the new control. To add the new control to an existing set of controls, drop the icon onto the control that you want the new control to follow.
- 3 Define the properties of the new control. As you specify properties that have an effect on the appearance of the form, the preview pane of the **Form** definition window is updated to show the change.
- 4 If the new control is a radio, check box, or group control, define its child controls in the same way. If the new control is a list, define its child elements. For more information, see *Defining element components* (on page 134).

The following sections indicate which icon to use to create a new control, describe how to define the control's properties, and indicate the types of child controls or elements the control can include.

Using an existing control definition

To include a previously defined control in a new item or control definition:

- 1 Open the form in which you want to include the control.
- 2 Locate the item or control in which you want to include a previously defined control, and highlight it.
- 3 Open the appropriate control tree in the **Trial Objects** window.
- 4 Drag the control you want to include onto the highlighted control in the **Form** definition window.

Defining a group, radio, or checkbox control




The InForm Architect application supports the following types of controls for grouping similar items:

- *Group controls* enable you to treat a set of controls as a single component for the purpose of including them in a nested control or item definition. A group control can be composed of any types of controls, including other group controls.
- Use a *radio group* to list a set of choices from which a user must select only one choice. Radio groups are appropriate when a small number of choices is available (for example, Yes, No, or Not Done), or when the definition of a single choice is expressed with another type of control; for example, a pulldown list. A radio group control can be composed of any types of controls, including other radio group controls.
- Use a *checkbox group* to define a control in which a user can select as many choices as are applicable. A checkbox group control can be composed of any types of controls, including other checkbox group controls.

These types of grouped controls have identical properties for the purpose of definition, therefore, this section describes all three types together.

Creating a new grouped control

To create a new group, radio group, or checkbox group control, drag and drop the following icon onto an item or onto another control:

Control	Icon
Group	
Radio group	
Checkbox group	

Defining grouped control properties

To define the properties of a group control, select the group control icon in the Form definition window and edit the group control's properties in the Properties window.

The following table describes the properties that you specify to create a group, radio group, or checkbox group control definition:


Property	Description
RefName	<p>RefName of the component.</p> <p>REQUIRED.</p> <p>For rules about the use of RefNames, see <i>RefNames</i> (on page 89)</p>
Layout	<p>Specifies how the controls included as children of the grouped control are oriented:</p> <ul style="list-style-type: none"> • Vertical—Controls are oriented vertically. • Horizontal—Controls are oriented horizontally • NoWrap (default)—Controls are oriented horizontally, and when the browser window is resized, they do not wrap. <p>OPTIONAL.</p>
UUID	<p>String that uniquely identifies the component across all databases, trials, and machines.</p> <p>Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters.</p> <p>OPTIONAL, not needed for grouped controls.</p>
Caption	<p>Specifies the text of the caption that appears on the screen with the component.</p> <p>OPTIONAL.</p>
Caption Alignment	<p>Specifies the position of the caption relative to the controls that make up the component. You can specify Left (the default), Right, Top, or Bottom.</p> <p>OPTIONAL.</p> <p>Note: The controls for key items for repeating forms and itemsets must not use a Caption Alignment property of top or bottom. Only left and right are currently supported.</p>
Design Note	<p>Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only.</p> <p>OPTIONAL.</p>

Property	Description
Selection Value	Visible only when the control is added as a child control in a radio group or checkbox group. Specifies the value of the child control stored when a user selects the control. OPTIONAL. If you do not specify a selection value, when a user selects the control in the radio group or checkbox group, the value stored in the database is a string consisting of the characters !pf! and the DBUID path of the selected control.

Defining a pulldown control

Use a *pulldown control* to create a list of choices from which a user must select one choice. Pulldown controls are composed of elements and can include units as a user selection or a label.

Creating a new pulldown control

To create a new pulldown control, drag and drop the following icon onto an item or onto another control: 

Defining pulldown control properties

To define the properties of a pulldown control, select the icon of the control in the **Form** definition window and edit the control's properties in the **Properties** window:


Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
UUID	String that uniquely identifies the component across all databases, trials, and machines. Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters. Optional, not needed for pulldown controls.
Caption	Specifies the text of the caption that appears on the screen with the component. OPTIONAL.
Caption Alignment	Specifies the position of the caption relative to the controls that make up the component. You can specify Left (the default), Right, Top, or Bottom. OPTIONAL.

Property	Description
Elements	Specifies the items to include in the pulldown control. If the list items you want to include have not yet been defined, you must first define them before you can include them in the pulldown control. For more information, see <i>Defining element components</i> (on page 134). At least one is REQUIRED.
Units	Specifies the unit type. Select the unit from the pull-down list. For information on how to create a new unit type, see <i>Defining units</i> (on page 136). OPTIONAL.
Unit Display Type	Element , indicating that the selected unit will be displayed as a label. OPTIONAL.
Design Note	Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only. OPTIONAL.
Selection Value	Visible only when the control is added as a child control in a radio group or checkbox group. Specifies the value of the child control stored when a user selects the control. OPTIONAL. If you do not specify a selection value, when a user selects the control in the radio group or checkbox group, the value stored in the database is a string consisting of the characters !pf! and the DBUID path of the selected control.

Defining a text box control

Use a *text box control* to create a box in which a user enters data explicitly. Text box controls can include units as a user selection or a label.

Creating a new text box control

To create a new text box control, drag and drop the following icon onto an item or onto another control:  The icon is a small square with a border containing the text 'l:abl'.

Defining text box control properties

To define the properties of a text box control, select the text box control icon in the **Form** definition window and edit the text box control's properties in the **Properties** window. As you specify properties that have an effect on the appearance of the form, the preview pane of the **New Form** window is updated to show the change.

Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
UUID	String that uniquely identifies the component across all databases, trials, and machines. Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters. OPTIONAL, not needed for text box controls.
Caption	Specifies the text of the caption that appears on the screen with the component. OPTIONAL.
Caption Alignment	Specifies the position of the caption relative to the controls that make up the component. You can specify Left (the default), Right, Top, or Bottom. OPTIONAL.
Height	Specifies how many lines of text the control displays. Default is one OPTIONAL.
Length	Specifies how wide the text box is in characters. To make all characters visible (that is, to avoid making users scroll to see all the text), make sure that this value is greater than or equal to any value you specify for the Max Length property. REQUIRED.
Max Length	Specifies how many characters a user can enter into the text box. If Max Length is greater than Length, the InForm application scrolls the extra characters. The default is the length specified in the Length property. OPTIONAL.
Data Type	Type of data a user can enter into the text box: Float, Integer, or String. String is the default. OPTIONAL.
Precision	Specifies the number of characters a user must enter after the decimal point, if you specified Float in the Data Type property. OPTIONAL.


Property	Description
Min Value	Specifies the minimum range of the value entered in the text box. OPTIONAL, available if the Data Type is Integer or Float.
Min Property	Specifies the operator to use in comparing the value entered in the text box with the value specified for Min Value. You can specify <code>GreaterThan</code> or <code>GreaterThanOrEqual</code> . For example, if you specify 50 as the Min Value and <code>GreaterThanOrEqual</code> as the Min Property, a data item must have a value of 50 or more, or the InForm application generates an error message. The use of the Min Value and Min Property takes the place of a range-checking rule on the item. REQUIRED if you specified a Min Value property for a text box control with a Data Type of Integer or Float.
Max Value	Specifies the maximum range of the value entered in the text box. OPTIONAL, available if the Data Type is Integer or Float.
Max Property	Specifies the operator to use in comparing the value entered in the text box with the value specified for Max Value. You can specify <code>LessThan</code> or <code>LessThanOrEqual</code> . For example, if you specify 100 as the Max Value and <code>LessThan</code> as the Max Property, a data item must have a value less than 100, or the InForm application generates an error message. The use of the Max Value and Max Property can take the place of a range-checking rule on the item. REQUIRED if you specified a Max Value property for a text box control with a Data Type of Integer or Float.
Units	Specifies the unit type, if you want a user to be able to choose the units associated with the value entered in the text box. Click the check box associated with each type of unit to display. For information on how to create a new unit type, see <i>Defining units</i> (on page 136). OPTIONAL.
Unit Display Type	If you selected a unit type, use the Unit Display property to select how units are displayed: <ul style="list-style-type: none"> • Element (the default)—Displays a single unit type as a label. • Pulldown—Displays multiple unit types as a pulldown list from which a user can select. • Radio—Displays multiple unit types as a set of radio buttons from which a user can select. OPTIONAL.
Design Note	Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only. OPTIONAL.

Property	Description
Selection Value	Visible only when the control is added as a child control in a radio group or checkbox group. Specifies the value of the child control stored when a user selects the control. OPTIONAL. If you do not specify a selection value, when a user selects the control in the radio group or checkbox group, the value stored in the database is a string consisting of the characters !pf! and the DBUID path of the selected control.

Defining a date/time control

Use a *date/time control* to capture entry of a date, a time, or both date and time.

Creating a new date/time control

To create a new date/time control, drag and drop the following icon onto an item or onto another control: 

Defining date/time control properties

To define the properties of a date/time control, select the date/time control icon in the Form definition window and edit the date/time control's properties in the Properties window:

Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
UUID	String that uniquely identifies the component across all databases, trials, and machines. Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters. REQUIRED. for certain types of date/time controls, otherwise not necessary. For information about required UUIDs, see <i>UUIDs</i> (on page 402).
Caption	Specifies the text of the caption that appears on the screen with the component. OPTIONAL.
Caption Alignment	Specifies the position of the caption relative to the controls that make up the component. You can specify Left (the default), Right, Top, or Bottom. OPTIONAL.

Property	Description
Start Year, End Year	First and last year to include in the pulldown list for year. REQUIRED..
Display Month, Display Day, Display Year, Display Hour, Display Minute, Display Second	These properties determine which date and time pulldown lists are rendered when the control is created online. The default value for the date properties Display Month, Display Day, and Display Year is True. The default value for the time properties Display Hour, Display Minute, and Display Second is False. OPTIONAL.
Require Month, Require Day, Require Year, Require Hour, Require Minute, Require Second	These properties determine which pulldown lists require entry for the item to be complete. The default value for all Require properties is False. OPTIONAL.
Allow Unknown Month, Allow Unknown Day, Allow Unknown Year, Allow Unknown Hour, Allow Unknown Minute, Allow Unknown Second	These properties determine whether each pulldown list includes a value of “UNK”, allowing a user to specify that a date or time value is unknown. The default value for all Unknown properties is False. OPTIONAL.

Property	Description
Consistency Check	<p>Specifies whether the InForm application checks for internal consistency among the entered components of the date and time. True is the default. When the value of Consistency Check is True:</p> <ul style="list-style-type: none"> • If any date or time component value is entered, all higher-order components displayed in the control must also have an entered, numeric value. For example, if a user enters a day value in a month/day/year date time control, the user must also enter the month and year. If the datetime control includes both date and time components and a user enters a time value, the user must also enter the date portion. • The InForm application treats a blank control and a control with the value of UNK identically. For example, if a user enters a numeric month and enters UNK for the year, the entry generates an error. • The consistency check applies to optional as well as required date time components; For example, if the time portion of a control is optional, the InForm application generates an error if a user enters the minutes without the hour. • If the Year component of the datetime control is not displayed, the consistency check is disabled. <p>OPTIONAL.</p>
Design Note	<p>Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only.</p> <p>OPTIONAL.</p>
Selection Value	<p>Visible only when the control is added as a child control in a radio group or checkbox group. Specifies the value of the child control stored when a user selects the control.</p> <p>OPTIONAL.</p> <p>If you do not specify a selection value, when a user selects the control in the radio group or checkbox group, the value stored in the database is a string consisting of the characters !pf! and the DBUID path of the selected control.</p>
Enter Time as Text	<p>Specifies whether the time portion of the control is represented as a set of text boxes instead of the default set of pulldown lists.</p> <p>OPTIONAL. The default is False (display as pulldown lists).</p>


Defining a calculated control

Use a *calculated control* to define a read-only value that is based on the value that a user enters into one or more other controls, or on a rule that generates a value on submission of a form. For example, use a calculated control to record a system-generated randomization drug kit number.

Note: A calculated control cannot be part of the key item on a repeating form or itemset.

Creating a new calculated control

To create a new calculated control:

- 1 Set the **Calculated** property of the item in which you want to define the calculated control to **True**.
- 2 Drag and drop the following icon onto an item or onto another control: 

Defining calculated control properties

To define the properties of a calculated control, select the calculated control icon in the **Form** definition window and edit the calculated control's properties in the **Properties** window. As you specify properties that have an effect on the appearance of the form, the preview pane of the **New Form** window is updated to show the change.

Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
UUID	String that uniquely identifies the component across all databases, trials, and machines. Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters. REQUIRED. for certain types of calculated controls, otherwise not necessary. For information about required UUIDs, see <i>UUIDs</i> (on page 402).
Caption	Specifies the text of the caption that appears on the screen with the component. OPTIONAL.
Caption Alignment	Specifies the position of the caption relative to the controls that make up the component. You can specify Left (the default), Right, Top, or Bottom. OPTIONAL.

Property	Description
Units	Specifies the unit type. Select the unit from the pull-down list. For information on how to create a new unit type, see <i>Defining units</i> (on page 136). OPTIONAL.
Unit Display Type	Element , indicating that the selected unit will be displayed as a label. OPTIONAL.
Design Note	Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only. OPTIONAL.


Creating a calculation rule

For the InForm application to calculate the value of a calculated control, you must define a calculation rule on the control. For information on how to do this, see *Writing a rule script* (on page 295).

Defining a simple control

Use a *simple control* to wrap an element for inclusion in a group control, radio group, check box group, or item. For example, a radio group with **Yes** and **No** selections can consist of two simple controls, one wrapping a **Yes** element and the other wrapping a **No** element.

Creating a new simple control

To create a new simple control, drag and drop the following icon onto an item or onto another control: 

Defining simple control properties

To define the properties of a simple control, select the simple control icon in the Form definition window and edit the simple control's properties in the Properties window:

Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)

Property	Description
UUID	<p>String that uniquely identifies the component across all databases, trials, and machines.</p> <p>Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters.</p> <p>OPTIONAL, not necessary for simple controls.</p>
Element	<p>Specifies the element to include in the simple control. If the element you want to include has not yet been defined, you must first define it before you can include it in the simple control. For more information, see <i>Defining element components</i> (on page 134).</p> <p>REQUIRED..</p>
Units	<p>Specifies the unit type. Select the unit from the pull-down list.</p> <p>For information on how to create a new unit type, see <i>Defining units</i> (on page 136).</p> <p>OPTIONAL.</p>
Unit Display Type	<p>Element, indicating that the selected unit will be displayed as a label.</p> <p>OPTIONAL.</p>
Design Note	<p>Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only.</p> <p>OPTIONAL.</p>

Defining element components

Elements are the lowest-level form components. You use them to define specific responses that a user selects in the context of a group or list type of control.

- In a pulldown list control, elements represent the choices in the list, and you select the elements to include when you define the pulldown list.
- In a group control, radio group, or check box group, elements can represent individual radio buttons or check boxes.

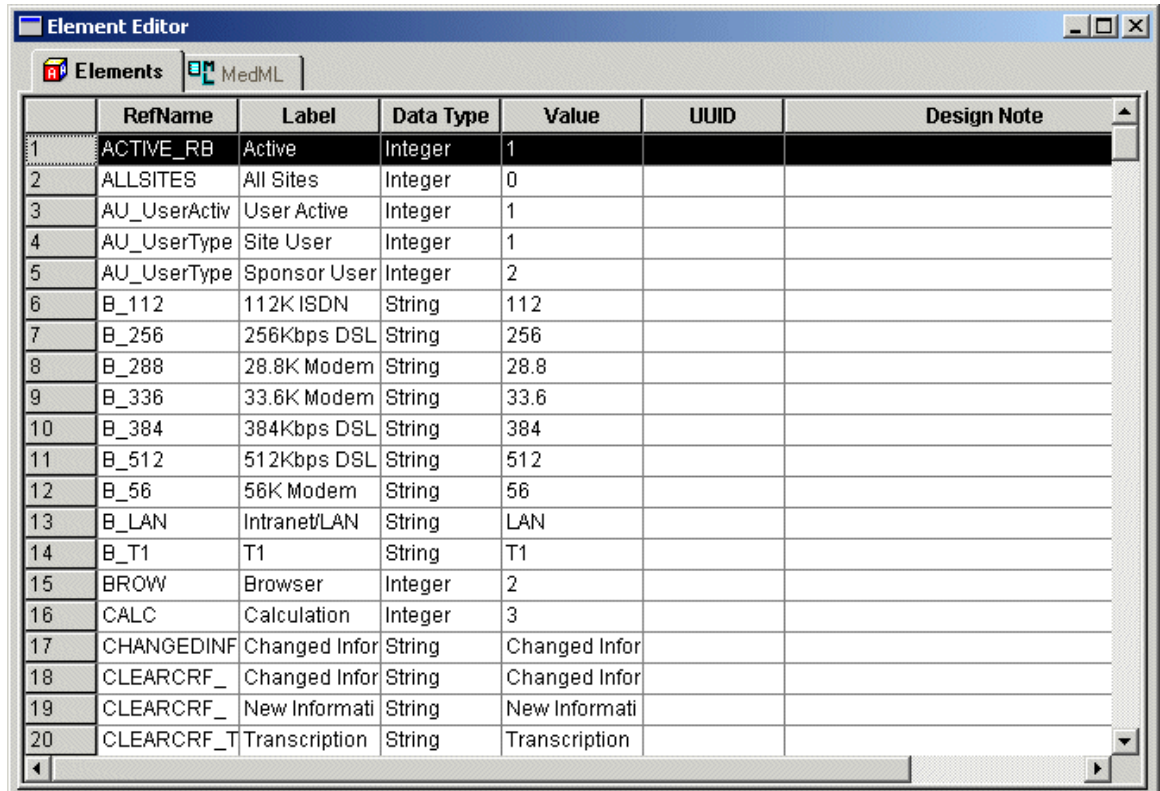
In these controls, an element must be wrapped in a simple control definition before inclusion in the group control. When you define a simple control to include in a group, you select the element for which the simple control is the wrapper.

Creating an element definition

To create an element definition:

- 1 Do one of the following:
 - Select **Trial > Element Editor**.
 - Click the **Element Editor** button in the **Trial** toolbar.
 - Select **File > New > Element**. If you use this method, skip the next step and go to step 3.

The **Element Editor** window opens.



The screenshot shows the 'Element Editor' window with a table of element definitions. The table has columns for RefName, Label, Data Type, Value, UUID, and Design Note. The table contains 20 rows of data.

	RefName	Label	Data Type	Value	UUID	Design Note
1	ACTIVE_RB	Active	Integer	1		
2	ALLSITES	All Sites	Integer	0		
3	AU_UserActiv	User Active	Integer	1		
4	AU_UserType	Site User	Integer	1		
5	AU_UserType	Sponsor User	Integer	2		
6	B_112	112K ISDN	String	112		
7	B_256	256Kbps DSL	String	256		
8	B_288	28.8K Modem	String	28.8		
9	B_336	33.6K Modem	String	33.6		
10	B_384	384Kbps DSL	String	384		
11	B_512	512Kbps DSL	String	512		
12	B_56	56K Modem	String	56		
13	B_LAN	Intranet/LAN	String	LAN		
14	B_T1	T1	String	T1		
15	BROW	Browser	Integer	2		
16	CALC	Calculation	Integer	3		
17	CHANGEDINF	Changed Infor	String	Changed Infor		
18	CLEARCRF_	Changed Infor	String	Changed Infor		
19	CLEARCRF_	New Informati	String	New Informati		
20	CLEARCRF_T	Transcription	String	Transcription		

- 2 Do one of the following:
 - Select **File > New > Element**.
 - Right-click in the **Element Editor** window, and select **Add Element** from the pop-up menu.

A template row appears in the window table with the default RefName of **ELEMENT1**.

- 3 Enter the following element properties in the new row:
 - In the **RefName** column, enter the **RefName** of the element. A RefName can have a maximum of 63 characters.
 - Optionally, in the **UUID** column, enter a string that uniquely identifies the unit across all databases, trials, and machines.
 - In the **Label** column, enter the text with which the element will be displayed online. A Label can have a maximum of 255 characters.
 - In the **Data Type** column, select the data type of the element: Integer, Float, or String.
 - In the **Value** column, enter the value with which the data is stored in the database when a user selects the element and submits the form. When a rule tests the value entered in a control with element components, this is the value it checks.

Note: The value must be compatible with the data type of the element.

- Optionally, in the **Design Note** column, enter free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the element. This information is for documentation only and is not displayed.

Note: Once created, elements cannot be deleted.

Angle brackets in element values

If you define an element with a String type, you can include special characters in the value. If you use angle brackets (< or >) in the definition of an element value, you must use their HTML representations:

- Less than symbol (<)—use <
- Greater than symbol (>)—use >

Navigating in the Element Editor window

To navigate to a specific element, type the first few letters of the element. The elements list scrolls to the first element whose name begins with that string. After a short pause (less than two seconds), you can type another search string.

Note: This feature is available only when the cursor is not focused on a cell in an element definition row. If you attempt to navigate by typing when you are editing a cell in an element definition, the typing overwrites the contents of the cell. To prevent this, press Esc before typing to navigate.

Defining units

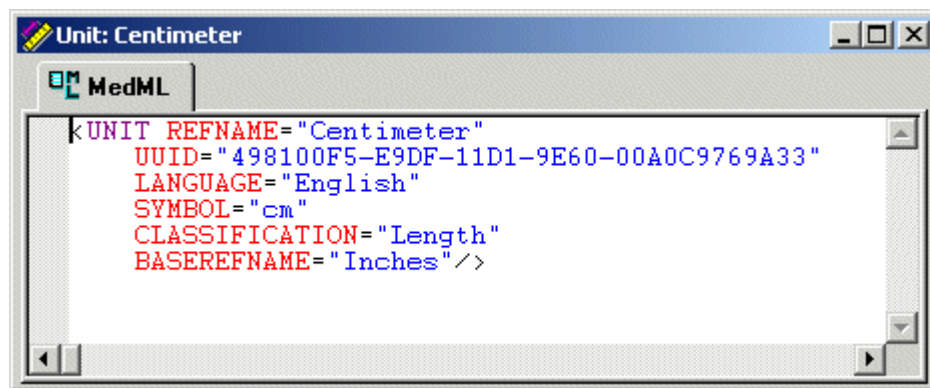
The definitions of a simple control, pulldown control, and text box control enable you to specify the units in which a user can enter data. When you define any of these types of controls, you can select the units to be associated with the control. This section describes how to define the units that are part of the Unit property selection list for these types of controls.

Creating a unit definition

To create a unit definition:

- 1 Do either of the following:
 - Select **File > New > Unit**.
 - In the **Trial Objects** window, right-click the **Units** node. From the pop-up menu, choose **Add Unit**.

The **Unit** window opens in the **Trial Design** workspace. This window consists of the **MedML tab**, which displays the MedML tags that define the unit.



- 2 Create a conversion rule to convert the entered value to the base value. For examples, see *Example of conversion rule for Inches unit* (on page 138) and *Example of conversion rule for Centimeter unit* (on page 138).
- 3 In the **Properties** window, enter values for the properties described in the following table:

Property	Description
RefName	RefName of the component. When you specify the RefName, the hierarchical representation of the component in the Trial Objects window is updated. REQUIRED..

Property	Description
UUID	<p>String that uniquely identifies the component across all databases, trials, and machines.</p> <p>Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters.</p> <p>OPTIONAL. Note that the unit definitions that are installed with the Base trial include predefined UUID properties. These UUIDs are not currently used. They are provided for possible future use and are listed for reference in <i>UUIDs</i> (on page 402).</p>
Symbol	<p>Label used to represent the unit on the screen.</p> <p>OPTIONAL.</p>
Classification	<p>Type of measurement represented by the unit.</p> <p>REQUIRED.. The following classifications have been predefined:</p> <ul style="list-style-type: none"> • Length • Weight • Temperature • Pressure • Volume • Area • Time • Frequency
Base Unit Name	<p>RefName of the unit definition used as the basis for conversion.</p> <p>REQUIRED..</p>
Convert To Base Rule	<p>RefName of a VBScript conversion rule used to convert the entered value to the base value.</p> <p>OPTIONAL.</p>
Convert From Base Rule	<p>RefName of a VBScript conversion rule used to convert to this unit from the unit specified in the Convert To Base Rule.</p> <p>Not currently used.</p>
Design Note	<p>Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only.</p> <p>OPTIONAL.</p>

Note: Once created, units cannot be deleted.

Example of conversion rule for Inches unit

The properties in this example define a unit called **Inches**. In this example, inches are both the unit being defined and the unit specified as the basis for conversion, so the conversion rule specified in the Conversion to Base Rule property simply multiplies the entered value by 1:

Property	Value
RefName	INCHES
Symbol	IN
Classification	Length
Base RefName	Inches
Conversion To Base Rule	UnitIsBase Rule script text: Data.Result=Data.BaseValue*1

Example of conversion rule for Centimeter unit

This example defines a unit called **Centimeters**. In this example, the Unit specified as the basis for conversion is **Inches**. Therefore, the conversion rule referenced in the Conversion to Base Rule property reflects the factor used to convert to inches from centimeters.

Property	Value
RefName	CENTIMETERS
Symbol	CM
Classification	Length
Base RefName	Inches
Conversion To Base Rule	CmToInches Rule script text: Data.Result=Data.Basevalue*.3937

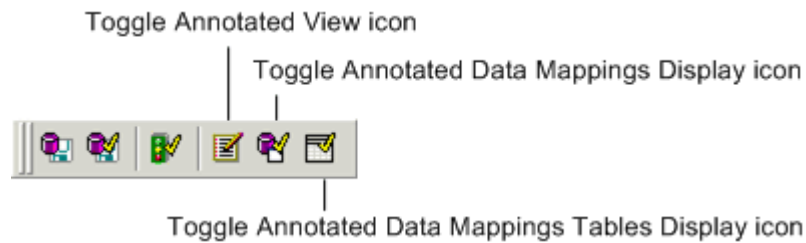
Annotating forms

The InForm Architect application enables you to annotate the forms you create with design information about the controls, elements, dates and external mappings. This allows you to print out design copies of the forms for design review and approval.

Once you have the form and all its controls and properties created, you can choose the types of design information with which you want to annotate the form. The following options are available for producing annotated forms:

- Annotated View
- Annotated External Mappings Display
- Annotated Data Mappings Tables Display

The Settings menu bar icons that enable you to use these features are:



Annotated View

When Toggle Annotated View is selected, the Form window displays annotations describing the form components, along with tables that provide further details about the component definitions. You can specify which types of annotations you want to see. For more information, see *Selecting meta data for display* (on page 142). The following figure illustrates how a form appears when Toggle Annotated View is selected.

PFST45 : DEMOGRAPHICS (DEM)	
Demographics	
1. Gender	[1] <input type="radio"/> Male [2] <input type="radio"/> Female
2. Date of Birth	Req ▾ / Req ▾ / Req ▾ (1932-1987)
3. Race	[4] <input type="radio"/> White, not of Hispanic origin [2] <input type="radio"/> Black, not of Hispanic origin [3] <input type="radio"/> Hispanic [1] <input type="radio"/> Asian or Pacific Islander [5] <input type="radio"/> American Indian or Alaskan Native [6] <input type="radio"/> Other or Unknown
4. Screening Date	Req ▾ / Req ▾ / Req ▾ (2004-2010)
5. Height	xxxx. <input type="radio"/> cm <input type="radio"/> in
Family History	
6. Marital Status	Pulldown List 1 ▾

Annotated Data Mappings Display

For the Annotated Data Mappings Display feature to take effect, you have to have the Annotated View on. Once you have created a data mapping and have used the Toggle Annotated View icon to turn on annotations, you can click the Toggle Annotated Data Mappings Display icon to view the data mappings as well. The following figure illustrates the annotations displayed when both Toggle Annotated View and Toggle Annotated Data Mappings Display are selected. Mapping annotations appear for a CDD and a Clintrial software mapping definition.

PFST45 : DEMOGRAPHICS (DEM)	
Demographics	
1. Gender	(MAPPINGS2:p_frmDem.sc_itmDe_rdcDemGendr) [1] <input type="radio"/> Male [2] <input type="radio"/> Female
2. Date of Birth	<input type="text" value="Req"/> / <input type="text" value="Req"/> / <input type="text" value="Req"/> (1932-1987) (MAPPINGS2:p_frmDem.sc_itmDe_dtmDemDO_DY) (MAPPINGS2:p_frmDem.sc_itmDe_dtmDemDO_MO) (MAPPINGS2:p_frmDem.sc_itmDe_dtmDemDO_YR)
3. Race	(MAPPINGS2:p_frmDem.sc_mitmR_mrdcRace) [4] <input type="radio"/> White, not of Hispanic origin [2] <input type="radio"/> Black, not of Hispanic origin [3] <input type="radio"/> Hispanic [1] <input type="radio"/> Asian or Pacific Islander [5] <input type="radio"/> American Indian or Alaskan Native [6] <input type="radio"/> Other or Unknown
4. Screening Date	<input type="text" value="Req"/> / <input type="text" value="Req"/> / <input type="text" value="Req"/> (2004-2010) (MAPPINGS2:p_frmDem.sc_itmDe_mGlobalD_DY) (MAPPINGS2:p_frmDem.sc_itmDe_mGlobalD_MO) (MAPPINGS2:p_frmDem.sc_itmDe_mGlobalD_YR)

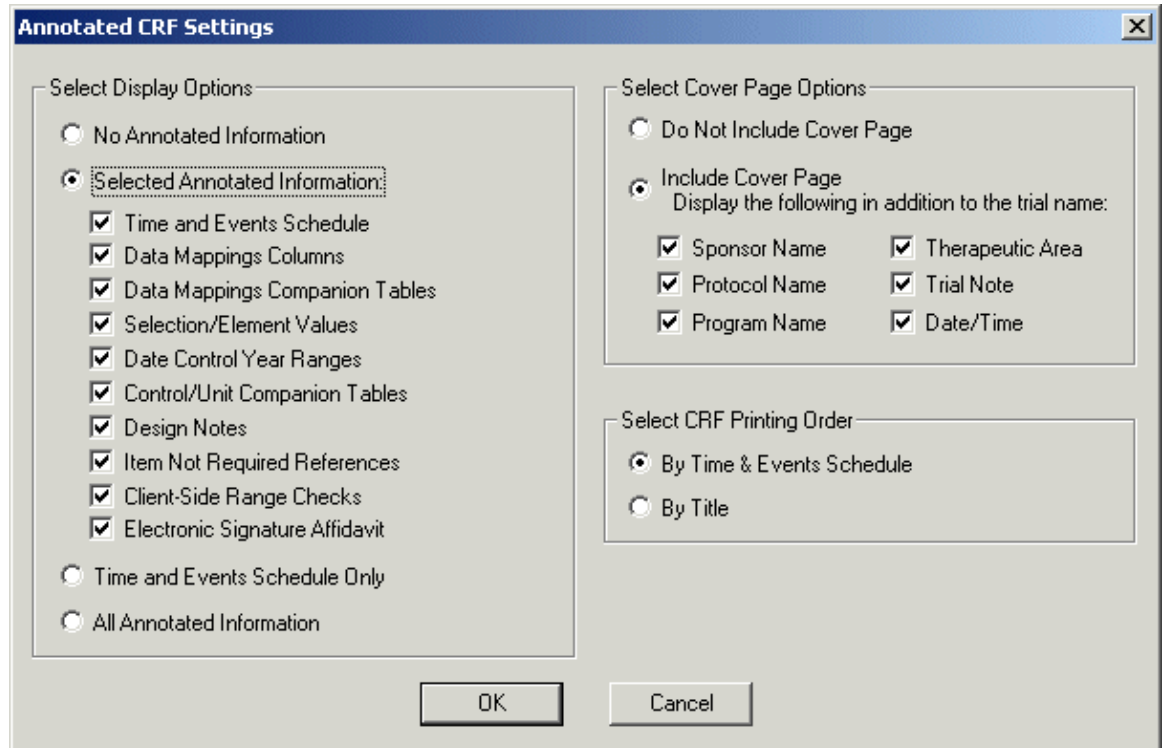
Annotated Data Mappings Tables Display

The Toggle Annotated Data Mappings Tables Display icon enables you to specify whether the annotated CRF display includes the tables defined in any CDD or Clintrial software data mappings that have been created for the CRF. When you select the Toggle Annotated Data Mappings Tables Display icon, listings of data mapping tables follow the display of the annotated CRF.

CDD: CDD_MAP Table: t_frmDem Key Type: PATIENTVISIT		
Column Name	Column Data Type	Design Note
sc_itmDemGe_rdcDemGendr	NUMERIC	
sc_itmDemDO_dtmDemDOB	DATE - DDMONYYYY	
sc_mitmRace_mrdcRace	NUMERIC	
sc_itmDemSc_dtmGlobalDate	DATE - DDMONYYYY	
sc_itmDemHg_txtDemHgt	FLOAT - F5.0	
sc_itmFHMSt_pdcFHMarriage	STRING(255) - NeverMarried, Married, Separated, Divorced, Widowed	
sc_itmFHChl_mrdcYesNo	NUMERIC	
sc_itmFHDep_rdcFHDepYUnk	NUMERIC	
sc_itmFHDep_FHGrandParent	STRING(255)	
sc_itmFHDep_HxchkFHParent	STRING(255)	
sc_itmFHDep_ParentSibling	STRING(255)	
sc_itmFHDep_xchkFHSibling	STRING(255)	
sc_itmFHDep_chkFHOtherFam	STRING(255)	
sc_itmFHDep_mtFHDDepOthr	STRING(60) - A60	

Selecting metadata for display

To select which metadata you want to appear on the form when it is in Annotated View and/or Annotated External Mappings Display, select View, Settings/Annotated CRF Settings. When you select the Annotated CRF Settings command, the following dialog box appears:



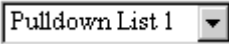
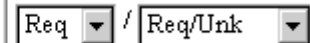
You can choose to display any of the following types of information about the form:

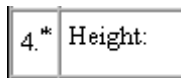
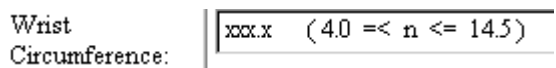
- **No Annotated Information**—When you choose this option, the Form window shows only the blank controls with no metadata.
- **Selected Annotated Information**—When you choose this option, the Form window shows the annotations you select.
- **Time and Events Schedule Only**—When you choose this option, a representation of the Time and Events Schedule appears in the Trial Design workspace when you select File, Annotated Trial Design. In the Annotated CRF Settings dialog box, all Selected Annotated Information check boxes are disabled. However, the Form window displays all of the selected annotations.
- **All Annotated Information**—When you choose this option, the Form window and the Annotated Trial Design window display every available annotation.

You can also choose whether or not to display a cover page and what it should include, and in what order to display the forms (alphabetically by title or as ordered in the Time & Events schedule). If included, the cover page appears in the Trial Design workspace when you select File > Annotated Trial Design.

Metadata in annotated CRFs

The following table presents how the meta data is displayed in the annotated CRF and in the tables that follow.

Meta Data	Displayed
External Column Mappings	<p>Appear in parenthesis above or beside their corresponding control:</p> <ul style="list-style-type: none"> The data mapping table name is concatenated to the column names. If a control is mapped to multiple data mappings, the path includes the data mapping name. If a control mapping includes a data label, the label follows the path after a forward slash. If a control mapping has a design note, a superscript follows the path, and the companion table for the external mapping includes the design note, identified by the same superscript <p>Example:</p> <pre>(CDD2.t.DEM.SH_HOVMUCHS_RADIO2NUMTEXT / HowMuchSmoked) (DEM_Only.DEM_e1.SH_HOVMUCHS_SMOKERADIO2) ¹</pre>
External Mapping Companion Table	<p>Appears labeled as such below the form. A companion table appears for each external mapping table that has form controls mapped to it. For an example, see <i>Data mapping tables</i> (on page 146).</p>
Selection/Element Values	<p>Selection values for check boxes and radio controls in a group control appear in square braces in front of their corresponding controls. If the value is derived from the control definition (for example, the element value of a simple control), the value appears in italic font.</p> <p>Selection value example:</p> <pre>{NDElement}</pre> <p>Element values for pulldown controls appear in companion tables below the form. The pulldown control shows the title of the appropriate companion table.</p> <p>Element value example:</p> 
Date Controls	<p>Year ranges appear in parenthesis beside their corresponding control:</p> <pre>{1932-1998}</pre> <p>If a date control component allows an “unknown” value, “Unk” appears in the control:</p> <p>Date of Birth: </p>

Meta Data	Displayed
Control/Unit Companion Tables	Appear below the form. For an example, see <i>Unit lists</i> (on page 148).
Design Notes	<ul style="list-style-type: none"> Form, section, itemset, and item design notes appear in tables following the form. All item notes appear in the same table (for example, see below). Design notes on a data mapping control definition appear in a Design Note column in the data mapping column table, identified by a footnote.
Item Not Required References	<p>Denoted by an asterisk in the first column of the form (where the items/itemsets are numbered).</p> 
Client-Side Range Checks	<p>The range check notation is displayed in the control.</p> 
Electronic Signature Affidavit	Appears below the form in the Annotated Trial Design view. For an example, see <i>Electronic signature affidavits</i> (on page 148).
Pulldown List References	The control contains a label that corresponds to a list in the companion tables following the form.
Text Box Specifications	<p>Appear in the control:</p> <ul style="list-style-type: none"> Data type and maximum character length for string data (text) numeric formatting information, including decimal places and significant digits if applicable, for numeric data (float)
Cover page	Appears when you select File, Annotated Trial Design and you have chosen to display it from the Annotated CRF Settings dialog box.
Sponsor Name Protocol Name Program Name Therapeutic Area Trial Note Date/Time	Part of the cover page displayed when selected on the Annotated CRF Settings dialog box. A suggested use of the Trial Note is to provide a revision number on the cover page.

Companion tables

In addition to the annotations that appear on the form, the following summary information is provided in tables that follow the form display:

- Element definition lists for pulldown or listbox controls.
- Data mapping tables.
- Design notes for form, section, itemsets, and item.
- Lists of units that appear as pulldown controls in the form.
- Electronic signature affidavits.

Element definition lists

Element definition lists are provided for pulldown or listbox controls (including RefName, internal value, display value, and design note information for each element).

To include element definition lists in the annotations, select **Control/Unit Comparison Tables** in the **Annotated CRF Settings** dialog box. To include the Value column in the lists, showing the internal value of each element as defined in the database, select **Selection/Element Values** in the **Annotated CRF Settings** dialog box.

The following figure illustrates an element definition list for a pulldown control annotated with the label Pulldown List 1.

Pulldown List 1:			
RefName	Display Text	Value	Design Note
WHITE	Caucasian	1	
BLACK	African American	2	BLACK element design note
ASIAN	Asian	3	ASIAN element design note
HISPANIC	Hispanic	4	

Data mapping tables

Data mapping tables show the following information for all mapped controls in CDD and Clintrial software mapping definitions:

- **CDD**—Data mapping name, table name, key type, column names, data types, possible values, and design notes.
- **Clintrial**—Data mapping name, panel name and properties, item names and properties of each item.

These tables appear when you select the **Toggle Annotated Data Mappings Tables Display** button or when you select the **Data Mappings Companion Tables** check box in the **Annotated CRF Settings** dialog box. The following figures illustrate a CDD mapping table and a Clintrial software mapping table:

CDD: CDD_MAP Table: t_frmDem Key Type: PATIENTVISIT		
Column Name	Column Data Type	Design Note
sc_itmDemGe_rdcDemGendr	NUMERIC	
sc_itmDemDO_dtmDemDOB	DATE - DDMONYYYY	
sc_mitmRace_mrdcRace	NUMERIC	
sc_itmDemSc_dtmGlobalDate	DATE - DDMONYYYY	
sc_itmDemHg_txtDemHgt	FLOAT - F5.0	
sc_itmFHMSt_pdcFHMarriage	STRING(255) - NeverMarried, Married, Separated, Divorced, Widowed	
sc_itmFHChl_mrdcYesNo	NUMERIC	
sc_itmFHDep_rdcFHDepYUnk	NUMERIC	
sc_itmFHDep_FHGrandParent	STRING(255)	
sc_itmFHDep_HxchkFHParent	STRING(255)	
sc_itmFHDep_ParentSibling	STRING(255)	
sc_itmFHDep_xchkFHSibling	STRING(255)	
sc_itmFHDep_chkFHOtherFam	STRING(255)	
sc_itmFHDep_mttxtFHDepOthr	STRING(60) - A60	

ClinTrial: CT_MAP Panel: p_frmDem Panel Type: 1 (1 Record Per Patient)													
Description: DEMOGRAPHICS													
Subset Item:			Detail: False			Detail CTItem:							
Protected: False			Verifiable: False										
Master Panel:			Master Item:										
SAS Name:			Lock Status: 0 (Modifiable)										
Item Name	Database Format	Date Part	SAS Name	Derived	Required	Repeat	Code List	Check List	Range	Key Order	Copy with Panel	Lock Status	Design Note
sc_itmDe_rdcDemGendr	NUMBER (1)											M	Gender
sc_itmDe_dtmDemDO_YR	NUMBER (4)	Year										M	Date of Birth
sc_itmDe_dtmDemDO_MO	NUMBER (4)	Month										M	Date of Birth
sc_itmDe_dtmDemDO_DY	NUMBER (4)	Day										M	Date of Birth
sc_mitmR_mrdcRace	NUMBER (1)											M	Race
sc_itmDe_mGlobalD_YR	NUMBER (4)	Year										M	Screening Date
sc_itmDe_mGlobalD_MO	NUMBER (4)	Month										M	Screening Date

Design notes

Design notes for forms, sections, and itemsets and items appear in individual design note tables, as illustrated in the following figure. These tables appear when you select **Design Notes** in the **Annotated CRF Settings** dialog box.

Form Design Note:
Added WRISTCIRC item 8/16

Section Design Notes:	
Title	Design Note
Smoking History	Add Section Note: See Alt form DEM02 if patient DOB is before 1942

Design notes for items are identified by the item number. If the item is in an itemset, the item number is followed by a letter to indicate the order of the item within the itemset.

Item Design Notes:	
Item No.	Design Note
3.	Add Hispanic to pulldown list
4.	in, cm as units in radio group display

Item Design Notes:	
Item No.	Design Note
1.a	Add bone density test
1.c	Needs range check rules
1.d	Discuss pulldown alternative

Unit lists

If a group of units is represented as a pulldown list in a form, a unit list table displays the units that are available in the pulldown list. To include unit list tables in the annotated trial design display, select **Control/Unit Comparison Tables** in the **Annotated CRF Settings** dialog box. The following figure illustrates a unit list table:

Unit List 1:			
RefName	Symbol	Base RefName	Classification
Inches	in	Inches	Length
Centimeter	cm	Inches	Length

Electronic signature affidavits

If a trial has forms that are associated with an electronic signature affidavit, the affidavit text appears in a companion table at the end of the complete listing of annotations that is generated when you select File > Annotated Trial Design. To include the affidavit text in the annotated trial design listing, select the **Electronic Signature Affidavit** check box in the **Annotated CRF Settings** dialog box.

The following figure illustrates signature affidavits for the Case Book and for an individual CRF. Note the space left in the text for the signer's name and title, which the InForm application fills in based on the user who initiates the signing activity.

CRB Electronic Signature Affidavit
By my dated signature below, I, , verify that all case report form pages accurately display the results of the examinations, tests, evaluations and treatments performed on this patient.
Pursuant to Section 11.100 of Title 21 of the Code of Federal Regulations, this is to certify that I intend that this electronic signature is to be the legally binding equivalent of my handwritten signature.
To this I do attest by supplying my password and clicking the button marked Submit below.

CRF Electronic Signature Affidavit
By my dated signature below, I, , verify that this case report form accurately displays the results of the examinations, tests, evaluations and treatments noted within.
Pursuant to Section 11.100 of Title 21 of the Code of Federal Regulations, this is to certify that I intend that this electronic signature is to be the legally binding equivalent of my handwritten signature.
To this I do attest by supplying my password and clicking the button marked Submit below.

Time and Events Schedule

Optionally, the annotated trial design listing includes a representation of the Time and Events Schedule for the trial. To include the Time and Events Schedule, select the **Time and Events Schedule** check box in the **Annotated CRF Settings** dialog box. To create a listing of only the Time and Events Schedule, select the **Time and Events Schedule Only** radio button.

The Time and Events Schedule listing features:

- An indication of the order in which each form occurs in each visit where it appears. For common forms, the order number is followed by the initial C.
- A code specifying whether each visit is Scheduled, Unscheduled, Optional, Repeating, or Dynamic.

Landscape print option

If there are a large number of visits in the Time and Events schedule, the printout will not show all visits. You must select the Landscape print option and the printout will then appear accordingly.

The following figure illustrates a Time and Events Schedule listing.

	Assessment	CRF	Baseline (Base) [S]	Core Week 1 (Wk 1) [S]	Core Week 2 (Wk 2) [S]	Core Week 4 (Wk 4) [S]	Core Week 8 (Wk 8) [S]	Study Termination (Final) [S]	Unscheduled Patient Visit (Unsch) [U/R]	VISIT1 (VISIT1) [S]	Extension Week 16 (Wk 16) [S/D]	Extension Week 32 (Wk 32) [S/D]	Conflict (Conflict) [U/R/D]
1	DATE OF VISIT	DOV	1	1	1	1	1	1	1		1	1	
2	PATIENT INFORMATION	PI	2										
3	DEMOGRAPHICS	DEM	3	9									
4	BASELINE PHYSICAL EXAMINATION	PE	4										
5	VITAL SIGNS	VS	5	2	2	2	3		2		2	3	
6	CLINICAL GLOBAL IMPRESSION	CGI	6	4	4	5	5				4	5	
7	HAMILTON RATING SCALE FOR DEPRESSION	HAMD	7	5	5	6	6				5	6	
8	BASELINE ELECTROCARDIOGRAM	ECG	8										
9	LABORATORY EVALUATION	LAB	9	8	7		8					8	
10	SERUM CHEMISTRY	CHEM	10		8		9					9	
11	HEMATOLOGY	HEMA	11		9		10					10	
12	RANDOMIZATION	RAND	12										
13	PRIOR and CONCOMITANT	CM	13-C-RF	6-C-RF	10-C-RF	7-C-RF	11-C-RF	2-C-RF	5-C-RF		8-C-RF	11-C-RF	

Cover page

Optionally, the annotated trial design listing includes a cover page containing any of the following information:

- Trial name
- Sponsor name
- Therapeutic area
- Program name
- Trial notes
- Date created

All of these properties can be found and edited in the Property sheet for the trial. To edit these properties:

- 1 Click trial name in the **Trial Objects** window.
- 2 When the property sheet is displayed in the **Properties** window, click each field to edit the property value.

To specify which properties you want to display on the cover page, and to specify whether or not to generate a cover page, select the desired values in the **Annotated CRF Settings** dialog box.

Viewing the annotated trial design

To see an annotated view of all of the forms in a trial, presented in the sequence specified in the Annotated CRF Setting dialog box, select **File > Annotated Trial Design**. The Annotated Design window opens in the Trial Design workspace and displays the forms, along with a cover page and a Time and Events Schedule, if you selected those options in the Annotated CRF Setting dialog box.

Note: Although you can view annotated versions of individual CRFs by opening their Form windows, you can only view the cover page, the Time and Events Schedule, and the electronic signature affidavits in the Annotated Trial Design window.

Printing the annotated trial design

There are two methods of printing annotated CRFs within a trial:

- Printing the form actively displayed in the Form Editor window.
- Printing all forms defined for the trial.

If your browser is Internet Explorer, you can use the following browser setting to print the annotated trial design with shaded heading bars, as it appears online:

- 1 Select **Tools > Internet Options** from the Browser menu bar.
- 2 In the **Internet Options** dialog box, click the **Advanced** tab.
- 3 In the **Printing** section, select **Print background colors and images**.
- 4 Click **OK**.

If you do not select this setting, the annotated trial design prints with no shading.

Setting up for printing

Before printing, you can specify page size, header and footer, page orientation, and margins by selecting **File > Page Setup** and setting the options in the **Page Setup** dialog box.

Note: Depending on the length of annotations, all annotated information may not fit on a page printed in Portrait mode. If this happens, switch to Landscape mode for a larger page width.

Printing the active form

To print the active form:

- 1 Display the form you wish to print displayed in the **Form** editor window, with **Toggle Annotated View** on.
- 2 Select **File > Print**.
- 3 Click **OK**.

Printing all forms in a trial

When you select File > Annotated Trial Design, the Annotated Design window opens in the Trial Design workspace, listing the full annotated trial design. This listing is generated with the features selected in the Annotated CRF Settings dialog box.

To print the entire file, or any number of virtual pages, select **File > Print**.

The order of the listing, if all features are selected, is as follows:

- 1 Cover page
- 2 Time and Events Schedule
- 3 CRFs. The order of CRFs is determined by the selections in the Annotated CRF Settings dialog box. You can order forms as they occur in the Time and Events Schedule or alphabetically by form title. The following listings and tables are displayed after each CRF:
 - a Design note lists
 - b Unit lists
 - c Element lists
 - d Data mapping tables
 - e Electronic signature affidavits

Each printed form starts a new page and its supporting information (pulldown list tables, data mapping tables, etc.) prints immediately afterwards. Aside from each form beginning on a new page, you cannot control or specify page breaks.

CHAPTER 6

Defining data mappings

In this chapter

- Overview154
- About data mappings155
- Mapping data to a Customer-Defined Database156
- Creating manual CDD mapping definitions.....170
- Mapping associations to a CDD.....186
- Mapping data to the Clintrial software189
- Autogenerating Clintrial mappings199
- Creating manual Clintrial mapping definitions.....204
- Customizing Clintrial mappings223
- Mapping autocoded data.....233
- Mapping data for coding with Central Coding239
- Mapping data to Oracle Clinical260
- Maintaining mappings274

Overview

This chapter describes how to use the InForm Architect application to create the definitions that specify how data is transferred from a trial database to one of the following:

- An external data management system.
- A calculated control targeted for a code supplied by an external autocoding tool.

About data mappings

To specify how to transfer data from a trial database to an external system for analysis, you create a set of data mapping definitions. The InForm application supports the following data mapping types:

- Customer-Defined Database (CDD)
- Clintrial software
- Central Coding
- Oracle Clinical
- Autocode

The definitions of data mappings specify:

- Where each mapped data point comes from in the source trial.
- Where each mapped data point goes in the coded control or the external system.

Two methods for creating data mappings

For a Customer-Defined Database (CDD) or Clintrial software, the InForm Architect application provides two methods for creating data mappings:

- Auto-generated data mappings.
- Manual data mappings.

Mapping data to a Customer-Defined Database

A *Customer-Defined Database* (CDD) is a database that gives sponsor personnel access to the data being collected in a trial while the trial is ongoing and when it is completed. Sponsors can review the data as needed to spot results that might require protocol adjustments or warrant early statistical analysis. They do not even need to shut down the trial database.

About CDD mappings

To specify how to transfer data from a trial database to a CDD for analysis, you create a set of CDD mapping definitions that specify:

- Where each mapped data point comes from in the source trial.
- Where each mapped data point goes in the CDD.
- How the data is organized in each CDD table.
- Optional, supplemental text about the design of the components of the CDD definition.
- Optional label text that is transferred to the CDD along with data values.

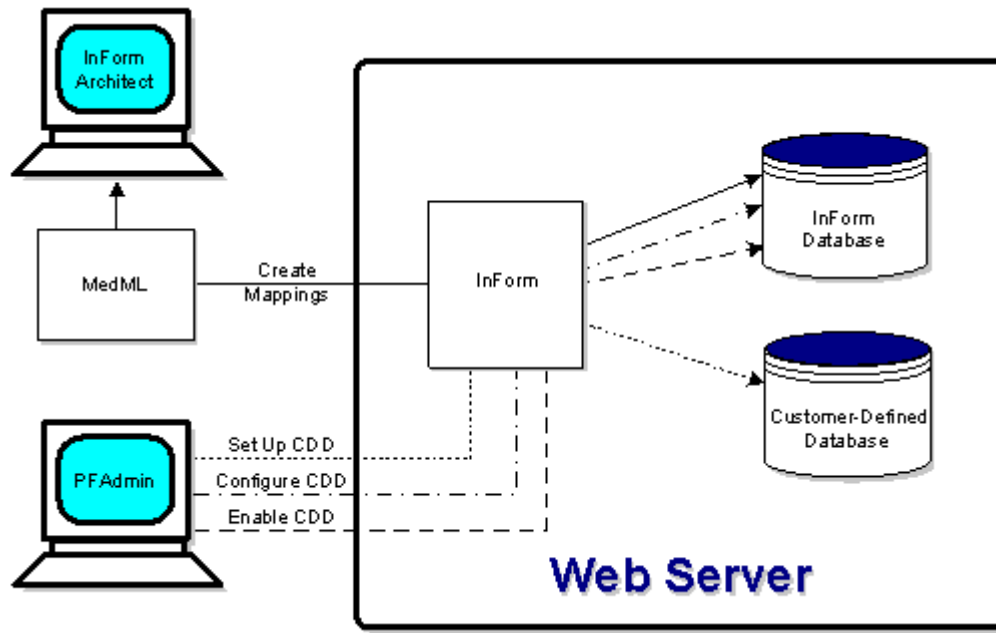
Setting up a CDD

Setting up a CDD involves the following steps:

- 1 Design CDD data mappings and define them through the InForm Architect application.

Note: After defining CDD data mappings you must install them in the InForm application database by saving them in the InForm Architect application with the Install MedML When Saving option enabled or by using the MedML Installer utility.

- 2 Create a CDD using the InForm application PFAdmin command.
- 3 Enable CDD processing using the PFAdmin command in the InForm application.



Note: This section describes how to define and maintain CDD mapping definitions in the InForm Architect application. For information on creating the CDD database and Oracle 9i user, creating the ODBC connection for a CDD, and enabling CDD processing in the InForm application, see *Installation and Configuration Guide*.

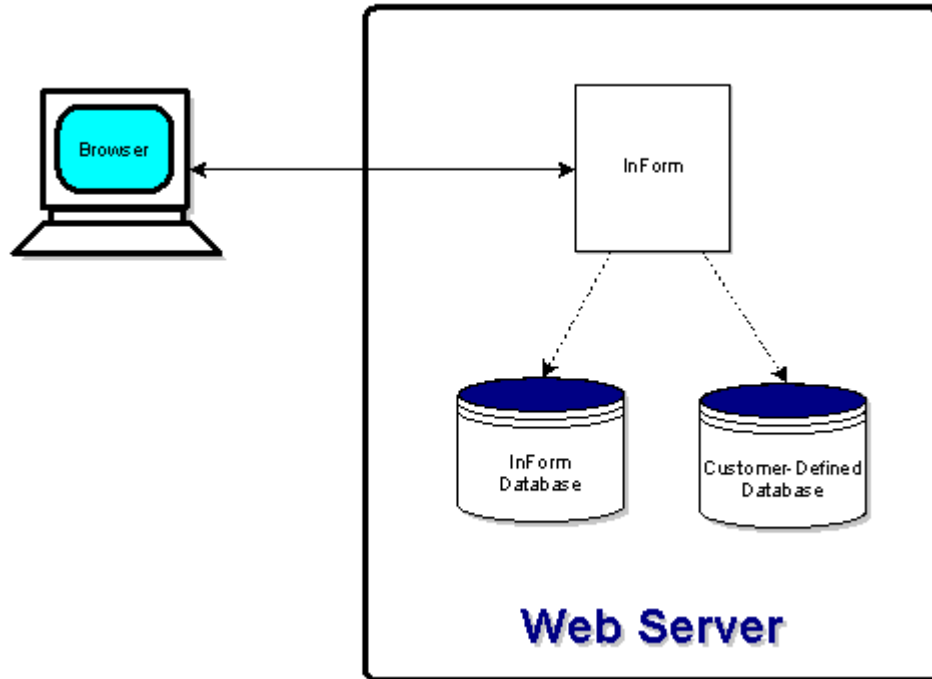
Types of CDD data transfer

The InForm software provides two ways to transfer data to a CDD:

- Transaction-based transfer
- Bulk transfer

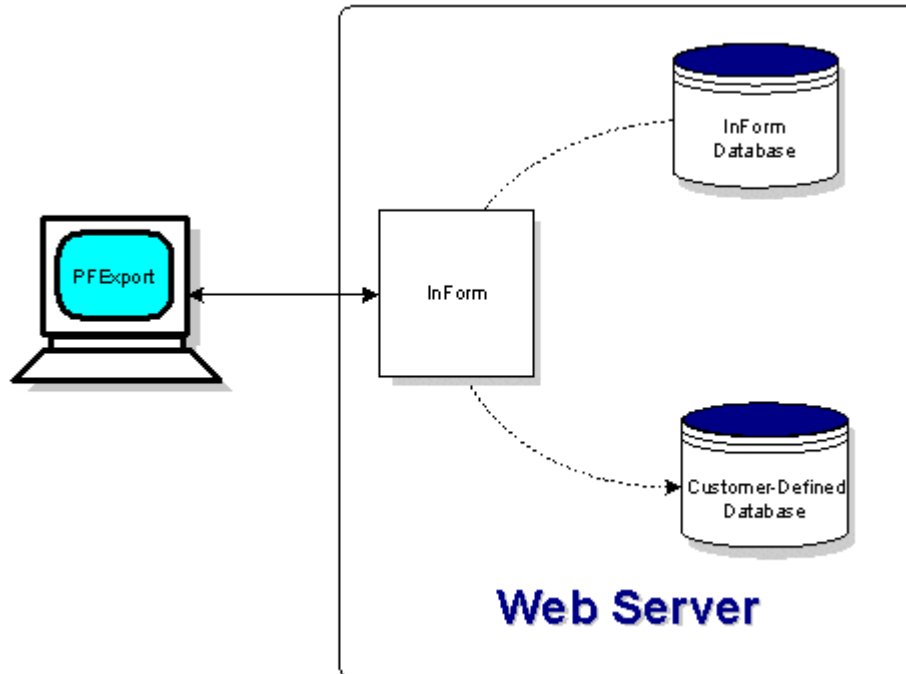
Transaction-based transfer

With transaction-based transfer, each time data is submitted or updated, the InForm application transfers the data to the CDD. This method enables you to have real-time access to data as the trial progresses.



Bulk transfer

With bulk transfer, the CDD Export component of the PF Export data transfer utility provided with the InForm application transfers data to the CDD on demand in a bulk export operation. This method enables you to transfer all of the data at once, for example, if you set up another analysis database after a trial has been running for a while. For information on using the CDD option of the InForm Export utility, see *InForm Utilities Guide*.



Designing CDD mappings

When designing CDD mappings, consider how you want to be able to retrieve the data from the CDD. Do you want to map multiple data points to the same CDD table column? Do you want to repeat a particular data point in each row of the table? Do you want to combine data from different visits, forms, or sections in the same CDD table?

Definitions of terms in CDD mappings

To understand how the InForm application handles the transfer of data to a CDD, you should familiarize yourself with the terms in the following table:

Term	Definition
Control path	A sequence of IDs referring to the InForm software components that define the physical location of a data point in a trial and in the InForm software database. The control path of a data point consists of its patient, visit, visit index, form, section, itemset, itemset index, item, and controls.

Term	Definition
DBUID	An ID that uniquely identifies a trial component (for example, an item) in the trial database. The primary key of a target table consists of a sequence of DBUIDs and indexes that correspond to the components of a control path.
Itemset index	A number that indicates a specific instance of an itemset row. If a data point is not in an itemset, its itemset index is 0.
Key type	A target table property that determines how the InForm software loads data into the target table. Two groups of key types are available: one group defines pivot tables, and the other defines non-pivot tables. For more information, see <i>Key types</i> (on page 161).
Pivot column	A column in a pivot table used to determine where to create duplicate entries of specified data points. When InForm software inserts data into a pivot table, it processes each pivot set in succession. In each pivot set, the InForm software creates a new row for each data point that is mapped to a pivot column. Then, it inserts the value of each data point that is mapped to a non-pivot column into each row of the pivot set. A pivot table can have only one pivot column. For more information and for examples, see <i>Pivot table key types</i> (on page 161).
Pivot set	A set of rows in a pivot table that are grouped by a portion of their composite key. The key type of the table specifies the parts of the table's primary key used for the grouping. For example, if the table has a key type of Pivot Form, a pivot set consists of the rows inserted for all data points that have the same patient, visit, form, and visit index values in their control paths.
Pivot table	A type of target table in which the key type is Pivot Patient, Pivot Visit, Pivot Form, or Pivot Section, and in which one of the columns is defined as a pivot column.
Repeating form index	A number that indicates a specific instance of a repeating form row. If a data point is not in a repeating form, its repeating form index is 0.
Target column	A target table column that receives data from one or more data points in a trial database. Each data point from which data is transferred to a CDD is mapped to a specific target column. You can map multiple data points to the same target column.
Target table	A table in a CDD. Target tables receive the data that is transferred from a trial database. The target table parameter must be limited to 27 characters.
Visit index	A number that indicates a specific instance of a repeating visit. If a visit is not repeating, its visit index is 1.

Key types

In each mapping definition, you specify the target table and column where each value of the data point goes. Additionally, the InForm application and the InForm Architect application provide a variety of options for grouping transferred data. These options fall into the following categories, and are identified in a CDD table definition by the Key Type property:

- **Pivot table key types**—These key types enable you to define CDD tables in which you can:
 - Map multiple data points to the same CDD table column.
 - Associate a particular data point with each row in the table.

In pivot tables, insertion of data is organized around a set of rows called a pivot set. The data points that make up a pivot set share the same values in specified parts of their control paths. The key type of a pivot table determines where each pivot set breaks by defining which parts of the control path are the same within a pivot set. For example, in a table with a key type of Pivot Section, a pivot set is made up of all data points in which the patient, visit, form, and section components of the control path match.

One of the columns in a pivot table is identified as the pivot column. Each time the InForm application loads one pivot set of data into a pivot table, it creates a new row for each data point mapped to the pivot column, and it duplicates data points mapped to non-pivot columns in each of those rows.

For more information and for examples, see *Pivot table key types* (on page 161).

- **Non-pivot table key types**—These key types enable you to define CDD tables in which data is grouped by a portion of the source data point's control path. For example, to capture all of the data for a single patient in one row, you might define a target table with the Patient Only key type. To capture the data for each form in a single row, with a new row for each patient and visit, you might define a target table with the Patient to Form key type.

Insertion into CDD target tables is determined by the primary key of the target table. When you create a CDD target table definition, the primary key of the table is a set of database IDs (DBUIDs) and indexes that correspond to the components of a data point's control path or a subset of the control path. When the InForm application loads data into a CDD table, it creates a new row each time the value of the primary key changes.

For more information and for examples, see *Pivot table key types* (on page 161).

Pivot table key types

In tables defined with pivot table key types, each key type specifies how much of the control path of a mapped control is used to compose the matching key components of a pivot set in the target table. For example, a pivot set in a table with the Pivot Visit key type consists of rows in which the PatientID, VisitID, and VisitIndex of each data point match. Tables with the following key types operate in this way:

- Pivot Patient
- Pivot Visit
- Pivot Form
- Pivot Section

You might choose one of these key types in order to be able to select all data values that have some related data value in common; for example, to select all lab values for a specific test that were collected on the same date.

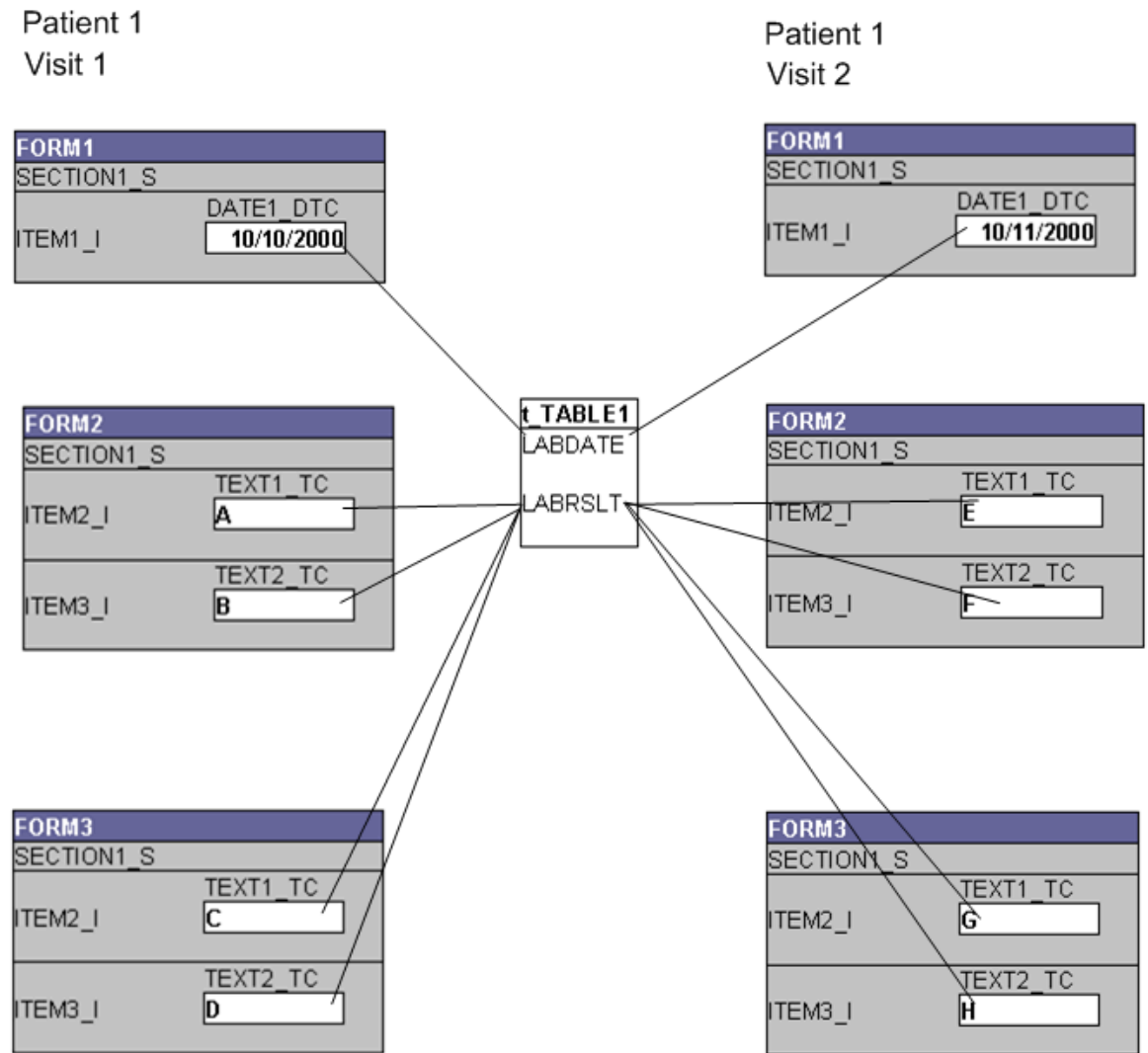
When you create a control path definition, you can specify a Data Label as a property of the control path. When the target table has a pivot key type, any Data Label defined for a data point mapped to a pivot column becomes a column in the target pivot table. This enables you to select data without knowing the sequence of IDs that makes up the target table's key columns.

Example - using pivot table keys

The following example illustrates a mapping scheme that uses the Pivot Visit key type to group lab results by visit date. In this scheme, data from three forms in two visits is mapped to the t_TABLE1 target table:

- The dates in FORM1 are both mapped to the LABDATE column.
- The text box controls in FORM2 and FORM3 are all mapped to the LABRSLT column.

The following figure illustrates the mapping of controls to columns in the t_TABLE1 target table.



T_TABLE1 is defined with a key type of Pivot Visit, and the LABRSLT column is defined as the pivot column. Because the key type is Pivot Visit, the components of the control path used to determine the composition of the pivot set are PatientID and VisitID. The date control and the four text box controls from the forms in Visit 1 have these IDs (P1 and V1) in common and thus make up one pivot set. The date and text boxes from the forms in Visit 2 also have common PatientIDs and VisitIDs (P1 and V2), and thus make up the second pivot set.

The following figure summarizes the characteristics of the table, column, and control path mapping definitions. In fact, this figure illustrates the layout of the InForm software database table PF_CDDCOLUMNMAPPING, which holds CDD mapping definitions.

CONTROLPATH	TOTABLE	TOCOLUMN	KEYTYPE	PIVOT	DATATYPE	LABEL	
P1.V1.FORM1.SECTION1_S.0.ITEM1_I.DATE1_DTC	t_TABLE1	LABDATE	PivotVisit	FALSE	DATE	L1A	PivotSet1
P1.V1.FORM2.SECTION1_S.0.ITEM2_I.TEXT1_TC	t_TABLE1	LABRSLT	PivotVisit	TRUE	TXT	L2A	
P1.V1.FORM2.SECTION1_S.0.ITEM3_I.TEXT2_TC	t_TABLE1	LABRSLT	PivotVisit	TRUE	TXT	L3A	
P1.V1.FORM3.SECTION1_S.0.ITEM2_I.TEXT1_TC	t_TABLE1	LABRSLT	PivotVisit	TRUE	TXT	L4A	
P1.V1.FORM3.SECTION1_S.0.ITEM3_I.TEXT2_TC	t_TABLE1	LABRSLT	PivotVisit	TRUE	TXT	L5A	
P1.V2.FORM1.SECTION1_S.0.ITEM1_I.DATE1_DTC	t_TABLE1	LABDATE	PivotVisit	FALSE	DATE	L1B	PivotSet2
P1.V2.FORM2.SECTION1_S.0.ITEM2_I.TEXT1_TC	t_TABLE1	LABRSLT	PivotVisit	TRUE	TXT	L2B	
P1.V2.FORM2.SECTION1_S.0.ITEM3_I.TEXT2_TC	t_TABLE1	LABRSLT	PivotVisit	TRUE	TXT	L3B	
P1.V2.FORM3.SECTION1_S.0.ITEM2_I.TEXT1_TC	t_TABLE1	LABRSLT	PivotVisit	TRUE	TXT	L4B	
P1.V2.FORM3.SECTION1_S.0.ITEM3_I.TEXT2_TC	t_TABLE1	LABRSLT	PivotVisit	TRUE	TXT	L5B	

When the InForm application inserts data from the first pivot set into the t_TABLE1 target table, it creates a row for each data value mapped to the pivot column, LABRSLT. Then, it populates all non-pivot columns with the same date value. Note that in CDD target tables, date and time control values are stored in several different columns, according to the composition of the control and its completeness. In this example, mapping the date to the LABDATE column results in entries in the LABDATE_DT and LABDATE_STR columns. When the pivot set key changes on the data from the second visit, it creates a row for each Visit 2 LABRSLT value and fills in the LABDATE_DT and LABDATE_STR columns with the date of the second visit. The Data Label property specified for each control path mapping definition appears in the LABEL column. The following figure illustrates the results of the data insert into t_TABLE1.

T_TABLE1																	
P	V	VI	ISI	F	S	IS	I	C1	C2	C3	C4	C5	LABEL	DATE	DATE_STR	DATE_TIME	LABRSLT
P1	V1	1	0		FORM2 SECTION1_S	0	ITEM2_I	TEXT1_TC					L2A	10/10/00			A
P1	V1	1	0		FORM2 SECTION1_S	0	ITEM3_I	TEXT2_TC					L3A	10/10/00			B
P1	V1	1	0		FORM3 SECTION1_S	0	ITEM2_I	TEXT1_TC					L4A	10/10/00			C
P1	V1	1	0		FORM3 SECTION1_S	0	ITEM3_I	TEXT2_TC					L5A	10/10/00			D
P1	V2	1	0		FORM2 SECTION1_S	0	ITEM2_I	TEXT1_TC					L2B	10/11/00			E
P1	V2	1	0		FORM2 SECTION1_S	0	ITEM3_I	TEXT2_TC					L3B	10/11/00			F
P1	V2	1	0		FORM3 SECTION1_S	0	ITEM2_I	TEXT1_TC					L4B	10/11/00			G
P1	V2	1	0		FORM3 SECTION1_S	0	ITEM3_I	TEXT2_TC					L5B	10/11/00			H

Note: This type of CDD mapping does not support itemset data. To create CDD mappings for controls that occur in itemsets, use one of the non-pivot table key types.

Non-pivot table key types

In tables with *non-pivot table key types*, each key type specifies how much of the control path of a mapped control is used to make up the primary key of the target table. For example, the primary key of a table with the *Patient to Form key type* consists of columns containing the *PatientID*, *VisitID*, *ItemsetIndex*, *VisitIndex*, and *FormID*. Tables with the following key types operate in this way:

Key type	You might choose this key type if you want...
Patient Only	All data for a patient in one row
Patient Visit (default)	All data for a visit in one row, with a new row for each patient or visit
Patient to Form	All data for a form in one row, with a new row for each patient, visit, or form
Patient to Section	All data for a section in one row, with a new row for each patient, visit, form, or section
Patient to Itemset	All data for an itemset instance in one row, with a new row for each patient, visit, form, section, or itemset row
Patient to Item	All data for an item in one row, with a new row for each patient, visit, form, section, itemset, or item
Patient to Control	Each control on a separate row

Because the *pivot table key types* **do not** support transfer of itemset data, you **must** choose one of the non-pivot table key types in order to create CDD mappings for data in an itemset.

When you create a control path definition, you can specify a Data Label as a property of the control path. When the target table has a key type of Patient to Control, the Data Label becomes a column in the target table. This enables you to select data without knowing the sequence of IDs that makes up the target table's key columns. For other key types, the Data Label text has a documentation function and is not transferred to the CDD target table.

Example - using non-pivot table keys

The following example illustrates a mapping scheme that uses the Patient to Section key type to insert lab results by section in the same row as the date on which they were collected. In this scheme, data from two sections of the same form is mapped to the t_TABLE1 target table:

- The dates in SECTION1_S and SECTION2_S are both mapped to the LABDATE column.
- The text box controls in ITEM2_1 in both sections are mapped to the LABRSLT column.
- The text box controls in ITEM3_1 in both sections are mapped to the LABRSLT2 column.

The following figure illustrates the mapping of controls to columns in the t_TABLE1 target table.

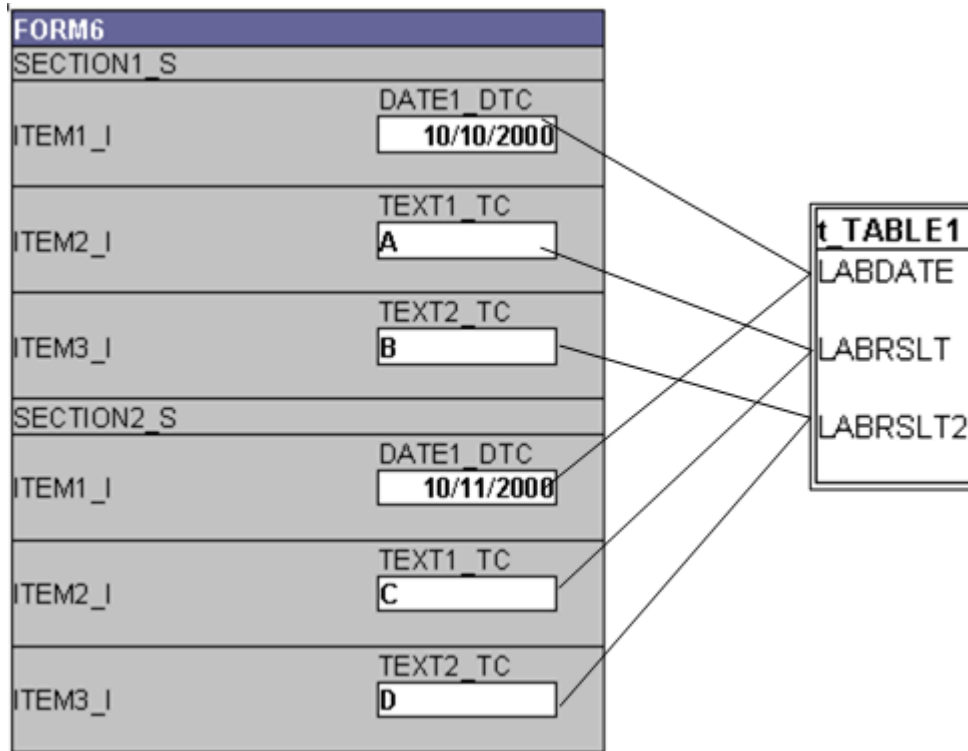


Table t_TABLE1 is defined with the Patient to Section key type. Its primary key consists of columns containing the PatientID, VisitID, VisitIndex, ItemsetIndex, FormID, and SectionID of each data point. It also contains target data columns for the LABDATE and for the two lab result items, LABRSLT and LABRSLT2. Because t_TABLE1 has a non-pivot table key type, no pivot column is defined. Any label data associated with the control path mappings is saved with the mapping definitions but will not be transferred to t_TABLE1.

The following figure summarizes the characteristics of the table, column, and control path mapping definitions. In fact, this figure illustrates the layout of the InForm application database table PF_CDDCOLUMNMAPPING, which holds CDD mapping definitions.

CONTROLPATH	TOTABLE	TOCOLUMN	KEYTYPE	PIVOT	DATATYPE	LABEL
P1.V1.FORM6.SECTION1_S.ITEM1_I.DATE1_DTC	t_TABLE1	DATE	PatientToSection	FALSE	DATE	
P1.V1.FORM6.SECTION1_S.ITEM2_I.TEXT1_TC	t_TABLE1	LABRSLT	PatientToSection	FALSE	TXT	
P1.V1.FORM6.SECTION1_S.ITEM3_I.TEXT2_TC	t_TABLE1	LABRSLT2	PatientToSection	FALSE	TXT	
P1.V1.FORM6.SECTION2_S.ITEM1_I.DATE1_DTC	t_TABLE1	DATE	PatientToSection	FALSE	DATE	
P1.V1.FORM6.SECTION2_S.ITEM2_I.TEXT1_TC	t_TABLE1	LABRSLT	PatientToSection	FALSE	TXT	
P1.V1.FORM6.SECTION2_S.ITEM3_I.TEXT2_TC	t_TABLE1	LABRSLT2	PatientToSection	FALSE	TXT	

When the InForm application inserts data from the form into the t_TABLE1 target table, it creates a row for each control path in which the PatientID, VisitID, VisitIndex, ItemsetIndex, FormID, and SectionID differs from the previous one. The data transferred from the three items in SECTION1_S all have the same IDs and indexes. Their control paths do not differ until the Item component. Therefore, the InForm application inserts those items into a single row in the target table. The data transferred from the three items in SECTION2_S have the same IDs and indexes as each other but a different SectionID from the previous group of data points. Therefore, the InForm application creates a new row and inserts those items into the new row.

The following figure illustrates the results of the data insert into t_TABLE1.

T_TABLE1																				
P	V	VI	ISI	F	S	IS	I	C1	C2	C3	C4	C5	LABEL	DATE	DATE_STR	DATE	TIME	TIME	LABRSLT	
P1	V1	1	0	FORM6	SECTION1_S	0	ITEM2_I	TEXT1_TC					L2A	10/10/00						A
P1	V1	1	0	FORM6	SECTION1_S	0	ITEM3_I	TEXT2_TC					L3A	10/10/00						B
P1	V1	1	0	FORM6	SECTION2_S	0	ITEM2_I	TEXT1_TC					L2B	10/11/00						C
P1	V1	1	0	FORM6	SECTION2_S	0	ITEM3_I	TEXT2_TC					L3B	10/11/00						D

When you create a CDD table definition, make sure that you choose a key type that includes all of the IDs required to uniquely identify each mapped data point. For example, if the key type is Patient to Form, and you map the same control occurring in two sections of the form to the same CDD column, the data value for the control in the second section will overwrite the first. To capture the values of both controls, you would need to choose at key type with a primary key that includes the SectionID or map the two controls to two different CDD columns.

CDD mapping and generic visits

CDD mapping definitions include the control path for each data source. If a form occurs in multiple visits, a CDD mapping definition is necessary for each control in each visit where it occurs, each mapping definition differing from the others only in the visit name. If you generate mappings for all of the data points in a trial, a large number of almost identical mappings can result, creating a heavy and error-prone maintenance burden.

To reduce the number of CDD mapping definitions required, the InForm and InForm Architect application support the concept of *generic visits*. A generic visit is a reserved MedML visit RefName (“PF_ALL_VISITS”) used in a CDD mapping definition. The generic visit RefName indicates that the control is a source data point for a CDD table in every visit in which it occurs. In the following example, as in the CDD window, a CDD mapping definition is represented by a string of RefNames connected by periods.

The VITAL SIGNS (VS) form occurs in almost every visit in the PFST sample trial—that is, in vstBASE, vstCORE1, vstCORE2, vstCORE4, vstCORE8, and vstUnschVisit. To generate CDD mapping definitions for the txtVSPulseRte control by using explicit visits, you would need to create six definitions—one for each visit in which the form appears. Using the concept of the generic visit, only a single definition is necessary, as shown in the following figure.

CDD mapping definitions with explicit visits

Patient (always 0)

```

0.vstBASE.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
  |           |           |           |           |
  Visit      Section     Item        Control
  
```

```

0.vstBASE.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
0.vstCORE1.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
0.vstCORE2.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
0.vstCORE4.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
0.vstCORE8.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
0.vstUnschVisit.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
  
```

CDD mapping definition with generic visit

Generic visit (always 0)

```

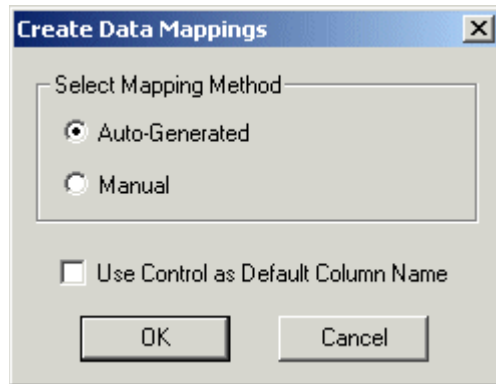
0.0.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
  
```

When you create a CDD with automatically generated mapping, the InForm Architect application creates mappings that specify the generic visit by default. When you create a CDD with manual mapping, you can specify the generic visit when you create a mapping definition for a form control.

Auto-generating CDD mappings

The preferred and easiest method for creating CDD mappings is to use the InForm Architect application auto-generate feature. To auto-generate CDD data mappings for a complete trial:

- 1 Start the InForm Architect application.
- 2 Open the trial for which you will create CDD data mappings.
- 3 In the **Trial Objects** tree, right click **Data Mappings** and select **Create CDD** from the pop-up menu to display the **Create Data Mappings** dialog.



- 4 In the **Create Data Mappings** dialog box, select **Auto-Generated**. To generate column names consisting of the RefNames of controls, select **Use Control as Default Column Name**.
- 5 Click **OK**. The InForm Architect application produces data mappings for each object in the trial.

Controlling column length for radio group mappings

By default the InForm Architect application generates mapping definitions for radio group controls using the maximum length of all possible values in the control. For example, if the longest string control in a radio group is a text control with a maximum length of 45, the InForm Architect application generates a mapping definition with a column length of VARCHAR(45).

By using a registry setting, you can specify that the InForm Architect application assigns a length of VARCHAR(255) instead of computing the column length of the mapping by evaluating the controls that make up the radio group. If this is the behavior you prefer:

- 1 Create a DWORD key called **FixedSizeRadioTDE** in the **HKEY_LOCAL_MACHINE/Software/Oracle/InForm Architect** folder of the registry on the server where the InForm Architect application is installed.
- 2 Assign a value of 1.

If the **FixedSizeRadioTDE** key is present and has a value of 1, the InForm Architect application assigns a length of VARCHAR(255) to radio group mappings with a type of STRING. If the key has a different value or is not present, the InForm Architect application assigns the maximum length defined for the components of the radio group.

Mapping new forms to an auto-generated CDD

When you create a new form in the InForm Architect application or load an XML file containing a form definition, the form is added to the list of forms in the Trial Objects window so that you can add to or update its design. To map form controls to a manual CDD, follow the instructions in *Creating a CDD mapping definition table* (on page 172), *Creating a CDD mapping definition table column* (on page 174), and *Mapping a form control* (on page 181). To map the form and its controls to an automatically generated CDD:

- 1 Open the **Form** window in the Design Workspace.
- 2 Right-click the **Form RefName**, and select **Add to Auto-Generated Data Mappings**.

The InForm Architect application generates table, table column, and control path mappings for the components of the form definition in each automatically-generated CDD and Clintrial software database in the trial.

Creating manual CDD mapping definitions

To set up data mapping definitions for a CDD, follow the procedure outlined here:


- 1 Create a CDD mapping definition.
- 2 Create CDD table definitions.
- 3 Create CDD table column definitions.
- 4 Create control data mapping definitions.

Creating a CDD mapping definition object

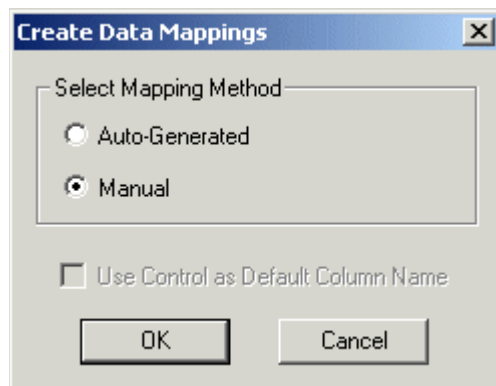
In the InForm Architect application, you can create a data mapping definition using the data mappings menu or toolbar commands or by loading an XML file containing MedML tags that specify a data mapping definition.

Note: Oracle recommends that you create data mapping definitions only after installing your completed trial definition in the strict mode of the MedML Installer tool. This insures that your component definitions are complete and all dependencies among definitions are satisfied. If you do not use strict mode (the default MedML Installer mode), any incomplete component definitions that non-strict mode allows to process are carried over into the data mapping definition.

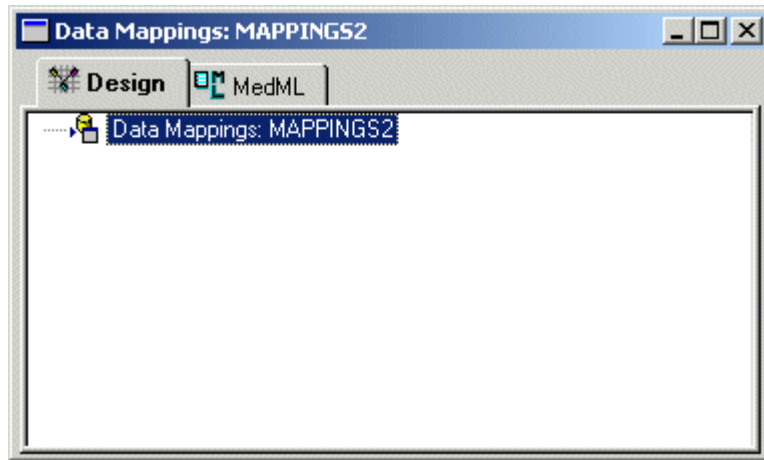
To create a CDD mapping definition object:

- 1 Do one of the following:
 - Select **Trial > Create Data Mappings > CDD**.
 - In the **Trial** toolbar, click the **Data Mappings** button (). In the **Data Mappings Type** dialog box, select **CDD** and click **OK**.
 - In the **Trial Objects** window, right-click the **Data Mappings** node and select **Create CDD** from the pop-up menu.

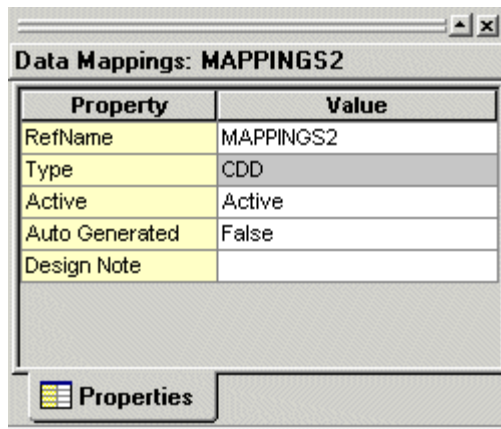
The **Create Data Mappings** dialog box appears.



- 2 Accept the default of **Manual**, and click **OK**. The **Data Mappings** window opens in the **Trial Design** workspace.



- 3 Edit the CDD mapping property values in the **Properties** window.



The following table describes the properties of a CDD mapping definition.

Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
Type	CDD. READ ONLY.
Active	Active or Inactive, indicating whether the CDD definition is accessible with InForm Architect application. The default is Active. To disable a CDD definition, set the value to Inactive and install the definition by saving it with the Install MedML When Saving option enabled or by using the MedML Installer utility. The CDD definition becomes inaccessible with the InForm Architect application. OPTIONAL.

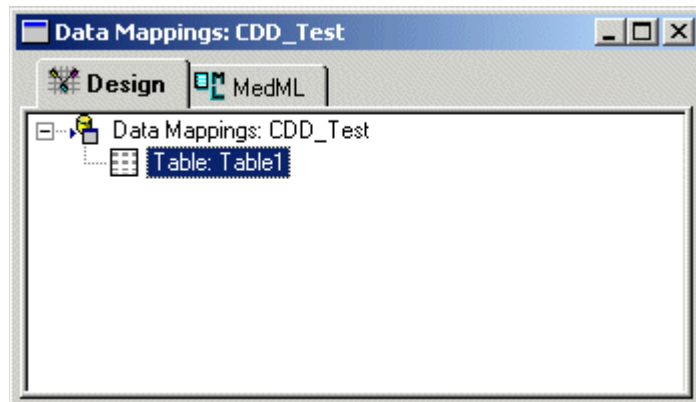
Property	Description
Auto Generated	True or False, indicating whether the CDD is in Auto-Generate mode. In this mode, when you add a new control to a form in the Form window, the InForm Architect application automatically generates a CDD mapping definition for it at the end of the list of column definitions in the target table. REQUIRED..
Design Note	Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only. OPTIONAL. Design Note text is not transferred to the target tables.

Creating a CDD mapping definition table

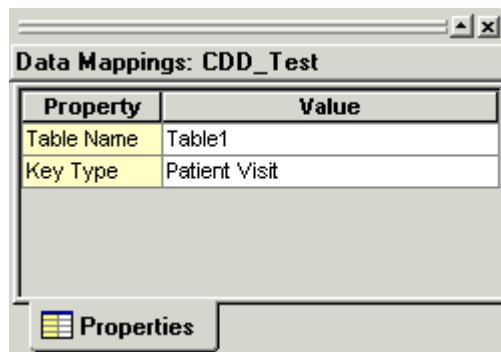
To create a CDD mapping table definition:

- 1 In the **Data Mappings** window, right-click the **CDD mapping RefName**, and select **New Table**.

A new table line is added to the **Data Mappings** node in the **Data Mappings** window.



- 2 In the **Data Mappings** window, select the table name.
- 3 Edit the CDD mapping table property values in the **Properties** window.



The following table describes the properties of a CDD mapping table definition:

Property	Description
Name	<p>Name of the data mapping target table. When you update this name, the new name is reflected in the Data Mappings window. This name can have a maximum of 30 characters.</p> <p>REQUIRED..</p>
Key Type	<p>For the following key types, this specifies the composition of the primary key columns of the target table. Each time a component of the primary key changes from the previous primary key submitted, the InForm application inserts a new row in the target table. Primary keys consist of the following DBUIDs and indexes:</p> <ul style="list-style-type: none"> • Patient Only—PatientID, FormIndex, ItemsetIndex. • Patient Visit (default)—PatientID, VisitID, FormIndex, ItemsetIndex, and VisitIndex. • Patient to Form—PatientID, VisitID, FormIndex, ItemsetIndex, VisitIndex, and FormID. • Patient to Section—PatientID, VisitID, FormIndex, ItemsetIndex, VisitIndex, FormID, and SectionID. • Patient to Itemset—PatientID, VisitID, FormIndex, ItemsetIndex, VisitIndex, FormID, SectionID, and ItemsetID. • Patient to Item—PatientID, VisitID, FormIndex, ItemsetIndex, VisitIndex, FormID, SectionID, ItemsetID, and ItemID. • Patient to Control—PatientID, VisitID, FormIndex, ItemsetIndex, VisitIndex, FormID, SectionID, ItemsetID, ItemID and five ControlIDs. A target table with this key type also contains a data label that can be used for data selection. <p>For the following key types, the primary key columns are PatientID, VisitID, ItemsetIndex, VisitIndex, FormID, SectionID, ItemsetID, ItemID and five ControlIDs. The key type selection determines the composition of a pivot set. Within a pivot set, data points mapped to non-pivot columns are repeated in each row. Target tables with these key types also contain a data label that can be used for data selection. Pivot set keys consist of the following DBUIDs and indexes:</p> <ul style="list-style-type: none"> • Pivot Patient—PatientID and VisitIndex • Pivot Visit—PatientID, VisitID, and VisitIndex • Pivot Form—PatientID, VisitID, FormID, and VisitIndex <p>Pivot Section—PatientID, VisitID, FormID, SectionID, and VisitIndexOPTIONAL.</p>
Pivot Column	<p>Identifies the column to use as the pivot column, if the key type is Pivot Form, Pivot Patient, Pivot Section, or Pivot Visit.</p>

Note: A table that has mappings for controls within an itemset cannot have any of the pivot key types. Additionally, a pivot column must be the first column in a data mapping table. When you select a pivot column, if it is not the first column, the InForm Architect application prompts you to reorder the column. If you do not choose to reorder the column to the first position in the table, the InForm Architect application does not make the column the pivot column.

Creating a CDD mapping definition table column

Creating a CDD mapping definition table column with the InForm Architect application has two parts:

- 1 Defining a column in a CDD mapping table.
- 2 Identifying the form controls that are mapped to the column.

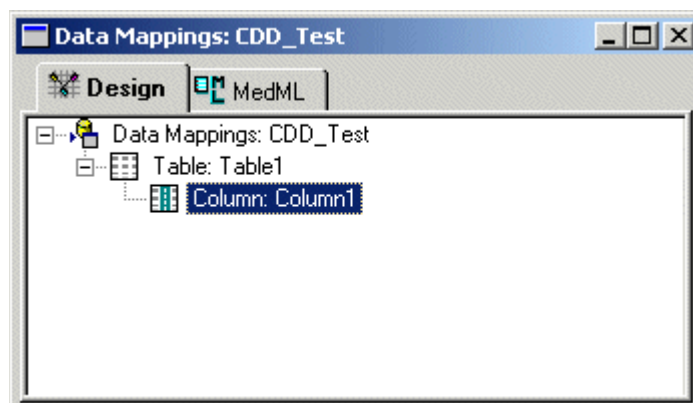
You can do these activities as separate steps, or you can create both the column and the control mapping in a single step.

Creating only a column definition

To create a column definition that does not include a control mapping:

- 1 In the **Data Mappings** window, right-click the name of the table in which you want to create a new column.
- 2 From the pop-up menu, choose **New Column Definition**.

The InForm Architect application adds a new column line to the table node in the **Data Mappings** window, and you can define its properties as described in *Defining CDD mapping table column properties* (on page 178).



Creating a column definition and mapping a control

To create a table column definition and control mapping in a single step, you can do either of the following:

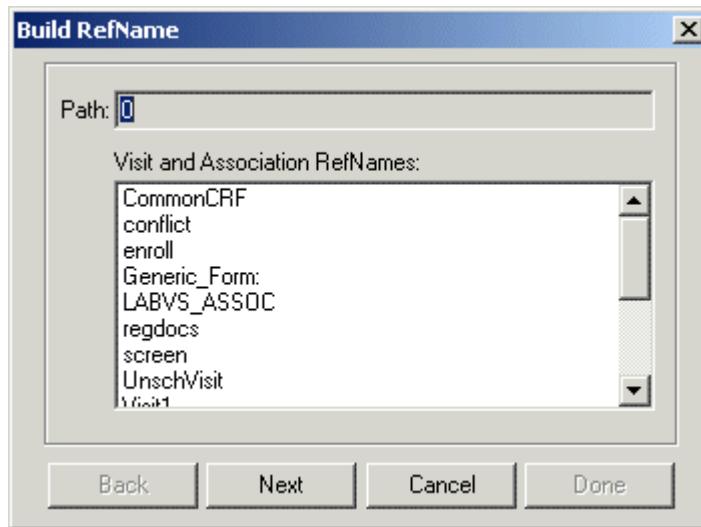
- **In the CDD Mappings window** (on page 175), use the New Column pop-up menu command for the table in which you want to create the column and control definition.
- **In the Form window** (on page 176), you can either use the Add Data Mapping pop-up menu command to map a control, or you can drag the control from the Form window onto the Data Mappings window.

Using the Data Mappings window to create a column definition and control mapping

To create the table column definition and control mapping definition in the Data Mappings window:

- 1 Right-click the name of the table in which you want to create a new column.
- 2 From the pop-up menu, choose **New Column**.

The Build RefName dialog box appears.



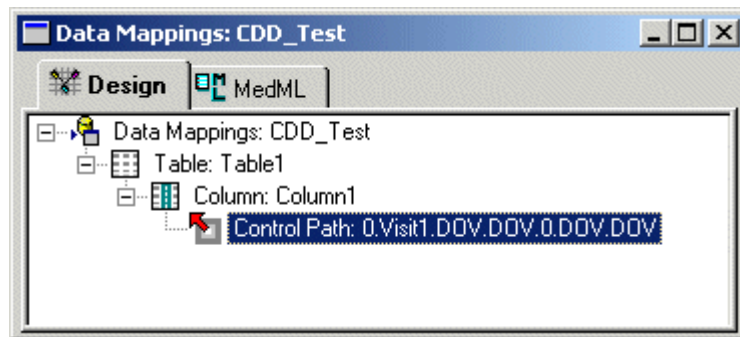
- 3 For each component of the RefName path, select the component from the list in the dialog box. To specify that you want the control to be mapped from all visits in which it occurs, select **Generic_Form** as the visit RefName.
- 4 Click **Next**, or, to backtrack to a previous component, click **Back**.

The InForm Architect application builds the path from your selections and displays it in the **Path** field at the top of the dialog box.

- 5 When the RefName path is complete, click **Done**.

The InForm Architect application inserts the path into the component definition you are creating.

- 6 Repeat this procedure as often as necessary to create additional column mappings for the child controls of group controls such as radio buttons. To view control path properties for any path, click the path in the **Data Mappings** window.



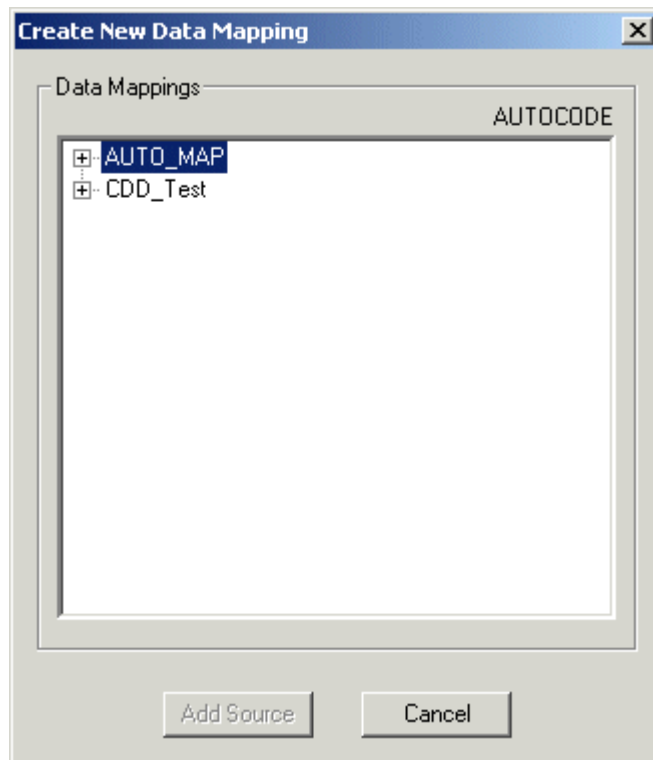
Using the Form window to create a column definition and control mapping

You can create a column and control mapping definition from the Form window. However, before you create a definition, the table in which you want to create the column definition must exist in the CDD mapping.

To create a definition from the Form window:

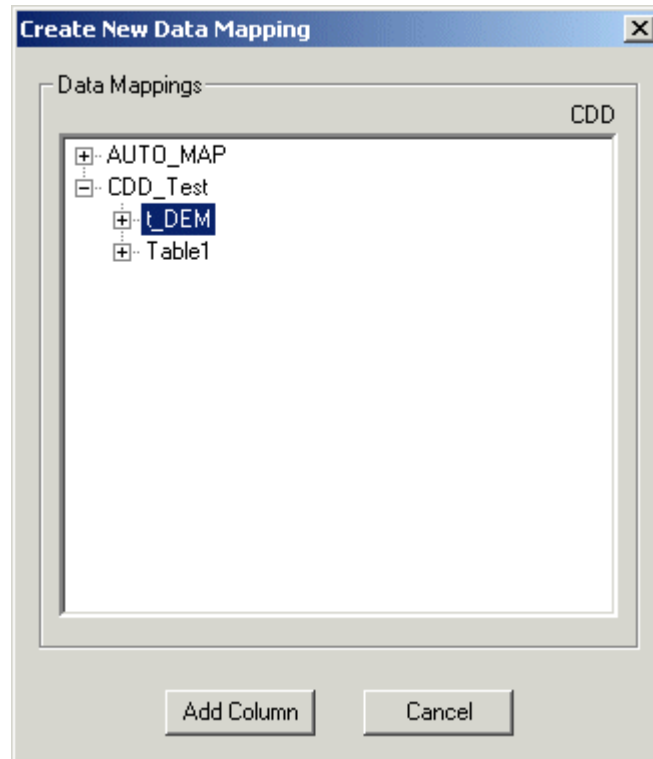
- 1 Open the **Form** window containing the control you want to map, and expand the tree so that the control is visible.
- 2 Right-click the control, and choose **Add Data Mapping** from the pop-up menu.

The **Create New Data Mapping** dialog box appears.



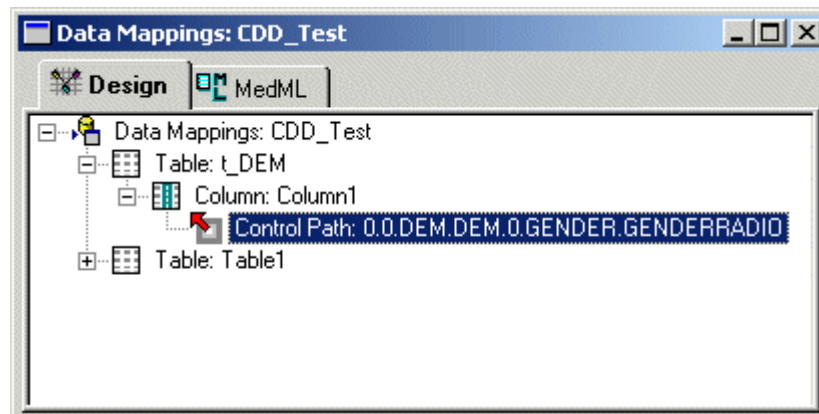
- 3 Expand the tree so that the table in which you want to create the column definition is visible.
- 4 Select the table.

The **Add Column** button becomes available.



- 5 Click **Add Column**.

The InForm Architect application generates a definition for the column mapping and the control path.



You can also quickly generate a definition by dragging the control from the Form window onto the Data Mappings window.

Note: The InForm Architect application only allows you to drag and drop controls for which you can legitimately create mappings. You cannot drag and drop a simple control or a group control.

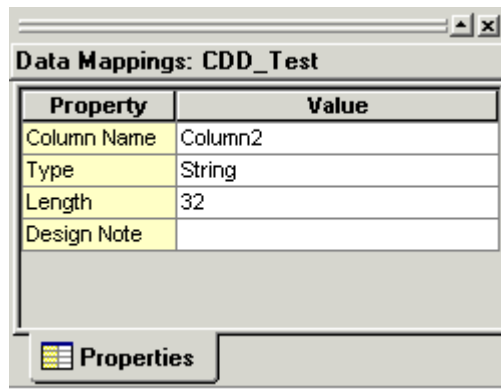
To drag a control from the Form window to the Data Mappings window:

- 1 Open the **Form** window containing the control you want to map and expand the tree so that the control is visible.
- 2 Open the **Data Mappings** window and expand the definition so that the table in which you want to create the column definition is visible.
- 3 Drag the control from the **Form** window and drop it onto the table in the **Data Mappings** window.

Defining CDD mapping table column properties

After creating a CDD mapping definition table column definition, specify its properties:

- 1 Select the table column name in the **Data Mappings** editor window.
- 2 Edit the CDD table column property values in the **Properties** window.



The following table describes the properties of a CDD mapping table column definition.

Property	Description
Name	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)

Property	Description
Type	<p>Specifies the column type in the target table. Note that when you create a control mapping, the InForm Architect application attempts to specify an appropriate column type based on the data type of the control. If the control type is not known, the default Type is Text.</p> <ul style="list-style-type: none"> • Date—Data for a complete date and time field. Note that when you map a date and time field, the InForm application actually creates four columns internally: <ul style="list-style-type: none"> ▪ DATE—For a complete date and time ▪ _DT suffix—For a date and time field with only date components ▪ _TM suffix—For a date and time field with only time components ▪ _STR suffix—For an incomplete date and time field • Float—Float data. When you map a float field, the InForm application creates three columns internally: one for the entered value, one for the normalized value, and one for the units associated with the value, if any. • Numeric—Numeric data. When you map a numeric field, the InForm application creates three columns internally: one for the entered value, one for the normalized value, and one for the units associated with the value, if any. • Split Date—Date and time data mapped to six columns, each containing one of the date or time components of the control. Each column has the generated or specified column name and the appropriate one of the following suffixes: _Day, _Mon, _Year, _Hour, _Min, or _Sec. • String—String data that is 255 characters long or less. • Text—Long varchar data. <p>OPTIONAL.</p>
Length	<p>Maximum length of a column, in the range 1 - 255. This property is visible only when the column type is STRING.</p> <p>REQUIRED. if the column type is STRING.</p>
Design Note	<p>Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only.</p> <p>OPTIONAL.</p> <p>Design Note text is not transferred to the target tables.</p>

Specifying properties for a control path

Along with its RefName path specification, a control path has an editable Data Label property, which is described in the following table:

Property	Description
Path	RefName path of the mapped control. Read only. REQUIRED..
Data Label	User-defined string to use for searching on data in the column. Maximum length is 30 characters. This label is included in the CDD mapping control path definition in the InForm database. For CDD mappings, it also appears in target CDD mapping tables that have any of the following key types: Patient to Control, Pivot Patient, Pivot Visit, Pivot Form, and Pivot Section. OPTIONAL.
Design Note	Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only. OPTIONAL. Design Note text is not transferred to the target tables.

Ordering columns in a table

When you add column definitions to a CDD mapping table, the InForm Architect application adds each new column to the end of the list. To rearrange the order of a column, drag and drop it on the name of the column you want it to follow. To insert it first in the list, drop it on the table name.

The InForm Architect application rearranges the MedML definition, and when you save and then use the MedML Installer to load the CDD mapping definition, the order you create is loaded into the InForm database and the CDD. Additionally, the CDD companion tables in the annotated CRF show the new order.

Mapping a form control

If you want to create new or additional mappings to an existing column definition, you must create a control mapping definition. This definition consists of the RefName path of the control, which is displayed but cannot be edited in the Properties window.

To create a control mapping, you can do either of the following:

- *In the Data Mappings window* (on page 181), use the Add Control pop-up menu command for the column in which you want to create the control path.
- *In the Form window*, (on page 182) you can either use the Add Data Mapping pop-up menu command for the control you want to map, or you can drag the control from the Form window onto the Data Mappings editor window.

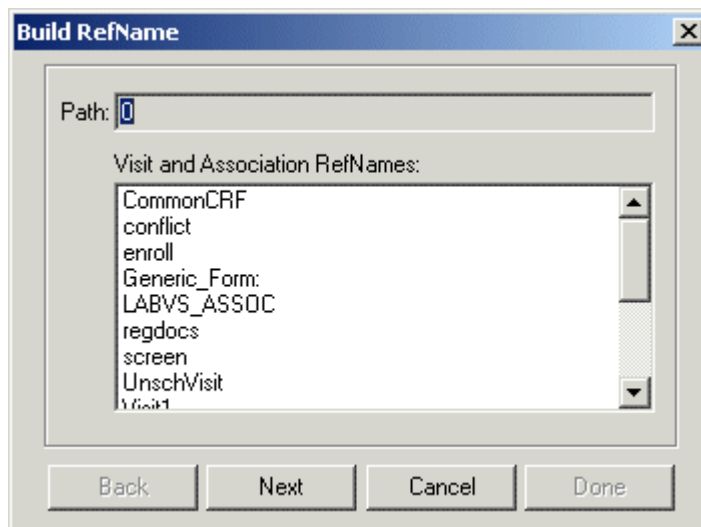
When you create a control mapping, the InForm Architect application generates control paths that use the Generic_Form visit type.

Using the Data Mappings window to create a control mapping

To create a control mapping definition In the Data Mappings window:

- 1 Right-click the name of the column in which you want to create a new control mapping.
- 2 From the right-click menu, select **Add Control**.

The Build RefName dialog box appears.

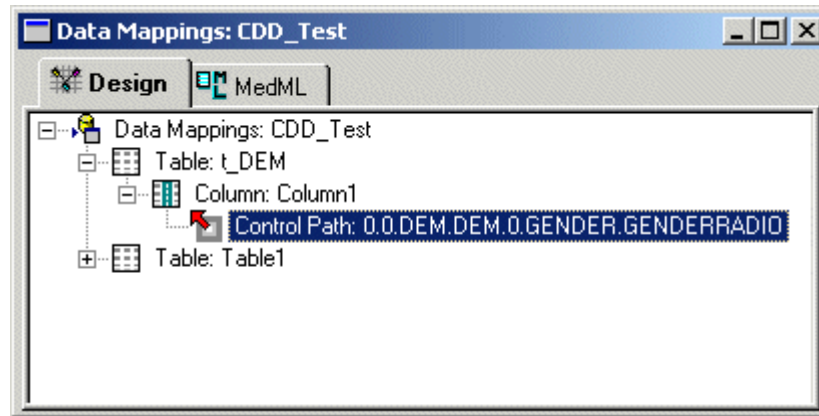


- 3 For each component of the RefName path, select the component from the list in the dialog box. To specify that you want the control to be mapped from all visits in which it occurs, select **Generic_Form** as the visit RefName.
- 4 Click **Next**, or, to backtrack to a previous component, click **Back**.

The InForm Architect application builds the path from your selections and displays it in the **Path** field at the top of the dialog box.

- 5 When the RefName path is complete, click **Done**.

The InForm Architect application inserts the path into the component definition you are creating.



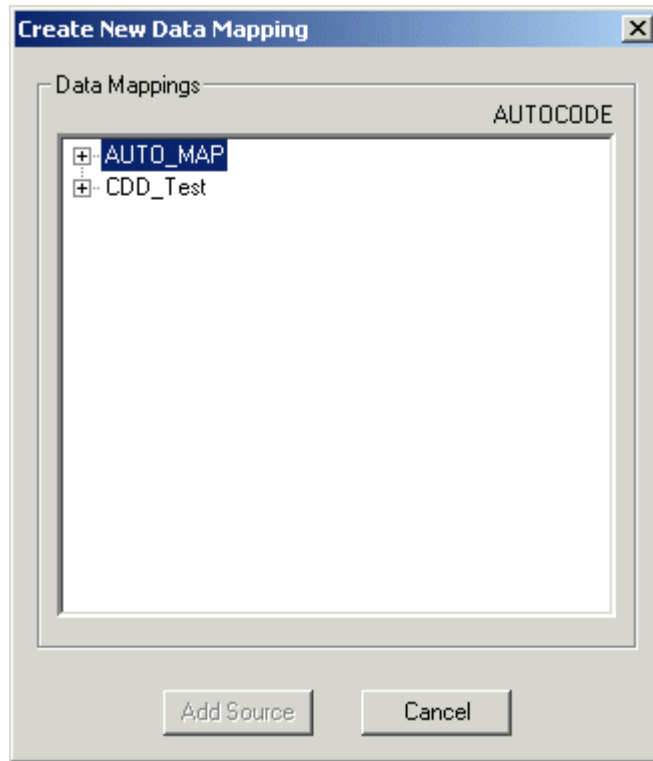
- 6 Edit the control mapping properties in the **Properties** window. For more information, see *Control path mapping properties* (on page 222).

Using the Form window to create a control mapping

To create a control mapping definition from the Form window:

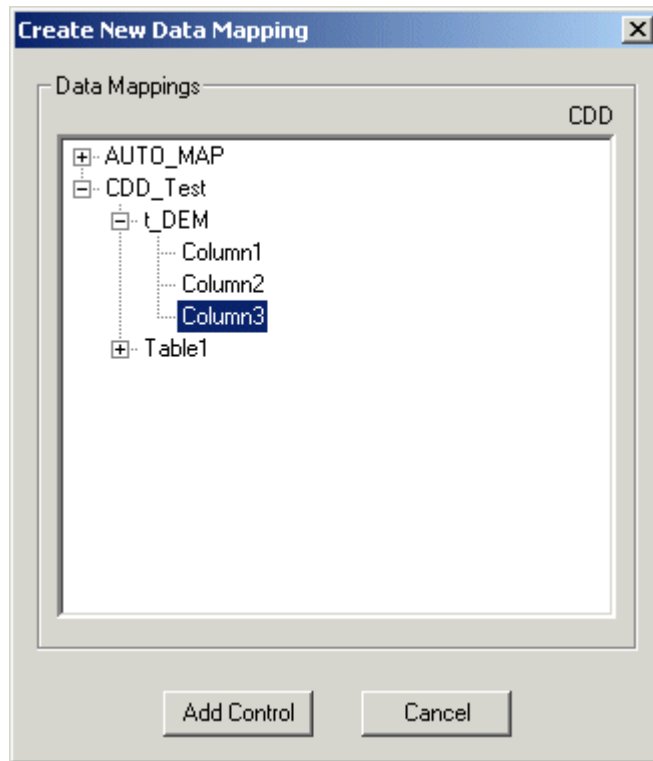
- 1 Open the **Form** window containing the control you want to map and expand the tree so that the control is visible.
- 2 Right-click the control, and select **Add Data Mapping** from the right-click menu.

The Create New Data Mapping dialog box appears.



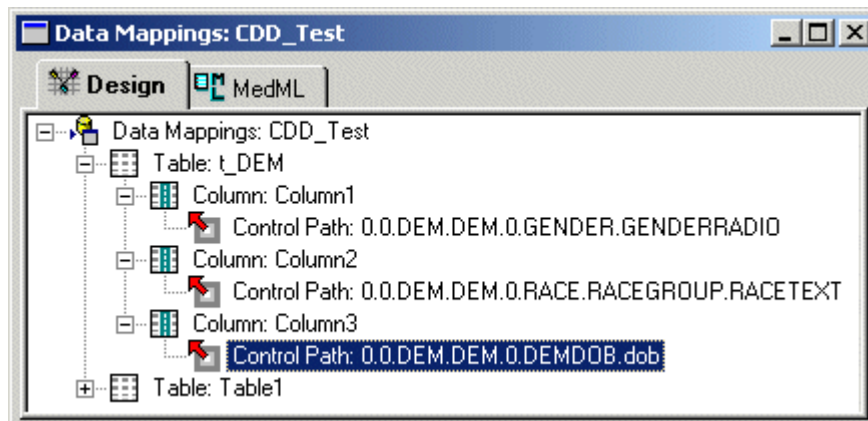
- 3 Expand the tree so that the column in which you want to create the control mapping definition is visible.
- 4 Select the column.

The Add Control button becomes available.



- 5 Click **Add Control**.

The InForm Architect application generates a definition for the control path.



- 6 Edit the control mapping properties in the **Properties** window. For more information, see *Control path mapping properties* (on page 222).

You can also quickly generate a definition by dragging the control from the Form window onto the Data Mappings window.

Note: The InForm Architect application only allows you to drag and drop controls for which you can legitimately create mappings. You cannot drag and drop a simple control or a group control.

To drag a control from the Form window to the Data Mappings window:

- 1 Open the **Form** window containing the control you want to map and expand the tree so that the control is visible.
- 2 Open the **Data Mappings** window and expand the definition so that the table in which you want to create the column definition is visible.
- 3 Drag the control from the **Form** window and drop it onto the table in the **Data Mappings** window.

Mapping associations to a CDD

When a trial includes repeating forms that are linked in an association, you create mapping definitions for the association separately from other trial objects. An association is a pair of repeating forms that are linked to allow the online display of cross-references between forms and the enhanced navigation between instances of the forms. For more information about associations and for instructions for defining an association, see *Associations* (on page 343).

When you auto-generate a CDD mapping, the InForm Architect application creates the association mappings along with the mappings of other trial objects. When you generate a CDD manually, you create an association mapping as described in *Creating an association mapping manually* (on page 187).

Association table schema definition

The principal difference between mappings for an association and mappings for other trial objects is that the schema definition of the CDD table for an association is fixed; you do not create table column mapping definitions. When you create an association mapping definition, the InForm Architect application generates a table with the following columns:

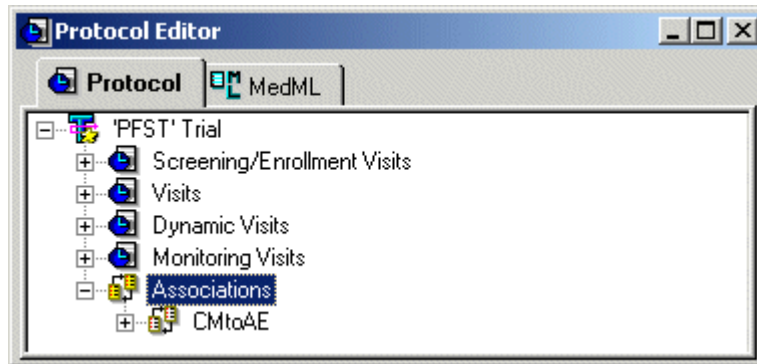
Column	Definition
PageAssocID: numeric	Primary key of the association mapping table
PageID_1: numeric	ID of the first CRF in the association.
PageIndex_1: float	Index of the instance of the first CRF.
VisitID_1: numeric	ID of the visit in which the first CRF occurs.
VisitIndex_1: numeric	Visit index of the visit in which the first CRF occurs
PageID_2: numeric	ID of the second CRF in the association.
PageIndex_2: float	Index of the instance of the second CRF.
VisitID_2: numeric	ID of the visit in which the second CRF occurs.
VisitIndex_2: numeric	Visit index of the visit in which the second CRF occurs
State: numeric	Status of the association: <ul style="list-style-type: none"> • 0 or null—Associated • 1—Not associated
PatientID: numeric	ID of the patient for whom data is being transferred

You can create an association mapping table for each association, or you can map multiple associations to the same table. Each table you create has the format described above.

Creating an association mapping manually

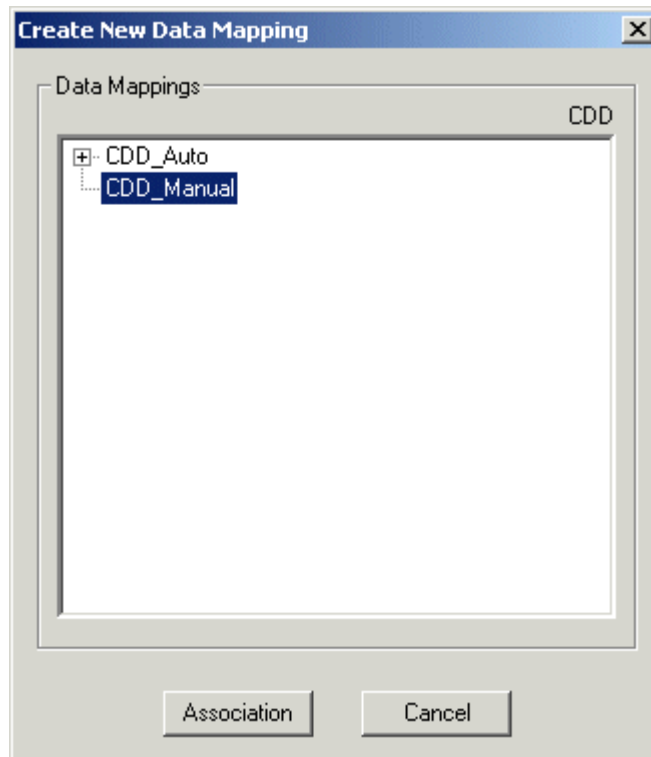
To create a mapping for an association:

- 1 In the **Trial Objects** window, double-click the **Associations** node.
The **Protocol Editor** window opens.
- 2 Expand the Associations node to expose the association you want to map.



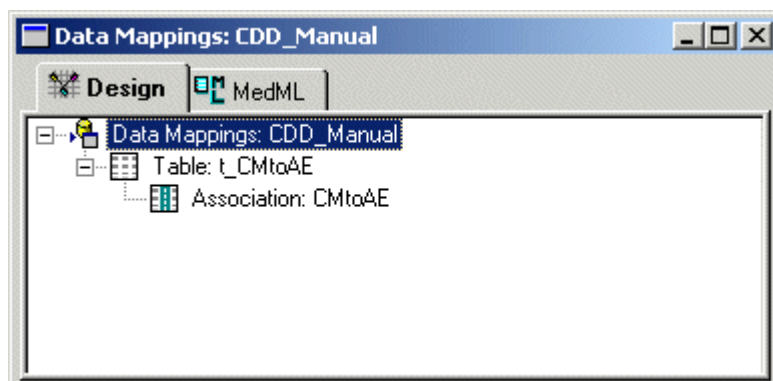
- 3 Right-click the association, and select **Add Data Mapping**.

The **Create New Data Mapping** dialog box opens.



- 4 Select the CDD definition in which you want to create the association mapping, and click **Association**.

The InForm Architect application generates the association mapping table definition, which is visible in the **Data Mappings** window for the CDD mapping after refreshing.



Defining association mapping properties

The only editable property for an association mapping definition is the name of the table. By default, the InForm Architect application assigns the prefix `t_` to the association name.

Mapping data to the Clintrial software

To pass data to a Clintrial database, you must create mapping definitions that show how InForm trial objects correspond to Clintrial objects. The InForm Architect application can generate mappings automatically for an entire trial, or you can create or modify them manually for individual objects.

Definitions of terms in Clintrial mappings

To understand how the InForm application handles the transfer of data to a Clintrial database, you should familiarize yourself with the terms in the following table.

Term	Definition
Context panel	A Clintrial panel containing a set of items common to all database tables in a protocol. When the InForm Architect application generates a Clintrial mapping, it supplies a context panel containing the items PATIENTID, VISITID, FORMID, VISITINDEX, and FORMINDEX.
Control path	A sequence of IDs referring to the InForm components that define the physical location of a data point in a trial and in the InForm database. The control path of a data point consists of its patient, visit, visit index, form, section, itemset, itemset index, item, and controls.
DBUID	An ID that uniquely identifies a trial component (for example, an item) in the trial database. The primary key of a target table consists of a sequence of DBUIDs and indexes that correspond to the components of a control path.
Itemset index	A number that indicates a specific instance of an itemset row. If a data point is not in an itemset, its itemset index is 0.
Master-detail relationship	A relationship between two page sections on a trial page, in which each record on one page section (the master page section) can have one or more associated records in the other page section (the detail page section). During data entry, the displayed records in the detail page section are associated with the selected record in the master page section.
Panel	A collection of logically related or clinically related items. A panel is an external representation of Clintrial database tables used to store clinical data.
Panel type	A specification of how the database tables associated with the panel are structured: <ul style="list-style-type: none"> • One record per patient. • One record per patient visit. • More than one record per patient. • More than one record per patient visit. • Enrollment panel.
Repeating form index	A number that indicates a specific instance of a repeating form row. If a data point is not in a repeating form, its repeating form index is 0.

Term	Definition
Subset item	Name of the item specified as the subset key for subset page sections based on the panel.
Visit index	A number that indicates a specific instance of a repeating visit. If a visit is not repeating, its visit index is 1.

Data mapping components

When you generate a set of data mapping components, either by registering a Clintrial protocol in a CIS library or by creating the mappings with the InForm Architect application, the following mapping components are created:

- CONTEXT panel mappings
- Data panel mappings
- Item mappings
- Control paths

CONTEXT panel mappings

Each Clintrial protocol has one CONTEXT panel. The items in the CONTEXT panel are attached as columns in a panel's clinical data tables when the panel is installed. When the InForm Architect application generates a Clintrial mapping, either through autogeneration or when you create a manual mapping definition, it supplies a CONTEXT panel whose items correspond to Clintrial CONTEXT items.

In Clintrial software, you assign CONTEXT item names when you create the context panel for a protocol. For example, the Medika Clinical sample trial has the following CONTEXT panel item names.

CONTEXT panel item name	CONTEXT item
SUBJECT	Subject Item
VISNO	Block Key Item
PAGENO	Page Key Item
VISRPT	Block Repeat Key Item
PAGERPT	Page Repeat Key Item

The CONTEXT panel items created from the mappings generated with the InForm Architect application have the following default names, established by the RefNames of the CONTEXT panel mapping definitions.

CONTEXT panel item name	Corresponding InForm trial object	CONTEXT item
PATNUM	Patient number	Subject Item
VISITID	Visit	Block Key Item
FORMID	Form	Page Key Item
VISITINDEX	Repeating visit index	Block Repeat Key Item
FORMINDEX	Repeating form index	Page Repeat Key Item

Note the following implications of assigning CONTEXT item names:

- When you design a trial with the InForm Architect application and synchronize to Clintrial software, the corresponding Clintrial protocol and panels are created automatically using the CONTEXT panel item RefNames assigned in the mappings generated by the InForm Architect application.
- When you design a trial in Clintrial software and import a Clintrial protocol to a CIS library, the autogenerated CONTEXT panel mappings preserve the CONTEXT item names assigned in the Design module.
- If you are running a hybrid trial or an EDC trial that includes components designed in both InForm Architect application and the Clintrial Design module, and the mappings do not match, data entered with InForm application and data entered with the Clintrial. Enter module is stored in data tables with different column heading names. You must ensure that data selection for your reports takes this into account.

If you want the mappings generated by InForm Architect application and CIS to match, do one of the following:

- Ensure that the CONTEXT item names assigned when you create a protocol with the Clintrial Design module match the CONTEXT item names generated by InForm Architect application, as shown above.
- In the mapping definitions generated by the InForm Architect application, change the RefNames of context panel items as needed before saving the trial MedML.

Working with additional CONTEXT panel items

The InForm Architect application generates mappings for the CONTEXT panel items.

The Clintrial Design module enables designers to define more CONTEXT panel items in addition to the ones handled by the InForm Architect application. For example, you could create a CONTEXT item called PROTOCOL, which would contain the protocol name in each record. Since the InForm Architect application does not automatically generate mappings and pass default values for additional CONTEXT panel items, users of the Enter module must manually enter the protocol name for each patient record.

If you desire additional CONTEXT items to be populated for EDC patients, you can create a derivation on the CONTEXT panel in Clintrial software to populate additional context panel items for EDC records.

PANEL mappings

In an automatically generated Clintrial mapping definition, each PANEL mapping definition corresponds to a panel in a Clintrial trial and to a form in an InForm trial. The RefName specified in a PANEL mapping definition is the panel name in the Clintrial protocol to which it is mapped.

In the InForm Architect application, you can also create PANEL mapping definitions that do not correspond directly to an InForm form. For example, you can create a mapping definition that generates a type 0 panel for non-patient data that will be populated from within the Clintrial software. For more information, see *Creating a type 0 panel* (on page 231).

ITEM mappings

Each ITEM definition in a set of automatically generated Clintrial mappings represents the correspondence between a panel item in the Clintrial software and a form item in the InForm application.

When you design a trial in the Clintrial software and import a Clintrial protocol to a CIS library, CIS creates an ITEM mapping definition for each panel item.

When you design a trial with the InForm Architect application and generate mappings, the InForm Architect application generates one ITEM mapping definition for each item in each form. Compound controls (radio groups and check box controls) in the InForm Architect application are mapped to a single column; you can manually add items defining mappings to columns that hold the data for each individual control.

Control paths

A control path mapping definition consists of a concatenated string of RefNames identifying the InForm components that define the physical location of a data point in a trial and in the InForm database. The control path of a data point consists of its patient, visit, visit index, form, section, itemset, itemset index, item, and controls. The CIS software and the InForm Architect application generate one control path per item.

Autogeneration and manual definition

The InForm Architect application provides the following options for creating mapping definitions:

- Autogenerated data mappings.
- Manual data mappings.

Autogenerated mappings

Autogeneration of mappings occurs when:

- You register a Clintrial protocol in a CIS library.
- You use the autogenerated mappings option to create mappings for a trial that you are designing with the InForm Architect application.

Autogenerated mappings and protocol imports

When you import a Clintrial protocol to a CIS library, CIS generates data mapping definitions for the following protocol components:

- The CONTEXT panel
- The following CONTEXT panel items:
 - PATNUM
 - VISITID
 - FORMID
 - VISITINDEX
 - FORMINDEX
- Each data panel
- Each panel item

CIS stores the data mapping definitions along with translated metadata definitions in the library protocol. When you import a protocol to the InForm Architect application, you specify whether to include the mapping definitions in the import. Once you have imported the mapping definitions to the InForm Architect application, you can customize them as needed, save them, and install them in an InForm trial.

Autogenerated mappings with the InForm Architect application

When you autogenerate mappings with the InForm Architect application, you specify:

- How you want column names to be generated.
- How you want date and time data to be mapped. Consider whether you want dates and times to be accessible:
 - As a single column or as separate columns for each part of the date or time.
 - As strings, integers, or complete DATE or DATETIME data types.

The automatic mappings feature maps all items, in all sections of an InForm form, to a single Clintrial panel. This feature automatically generates the panels, panel items and mapping details for a trial in the Data Mappings window of the InForm Architect application.

If multiple instances of a form are used within the InForm Architect application, you can see only one Clintrial panel for all instances of the form. The database tables (panels) in the Clintrial software include the appropriate references.

Manual mappings

If you choose, you can create manual data mappings for a trial. (This is not recommended unless you plan to store only a few data values in the Clintrial database, for example, during the testing of a specific portion of a trial design.) A more common strategy is to autogenerate mappings initially and then change them as necessary or create additional manual mappings to accommodate any data access requirements you have that are not met by the autogenerated mapping definitions.

After initially autogenerating mappings for a trial, you must manually update the mappings when you make changes to the trial if you want to capture the new or changed data in the Clintrial database. For example, you must update mappings manually when:

- Refreshing an existing protocol in the CIS library with changes introduced with the InForm Architect application.
- Adding a visit, form, or item to a trial.
- Changing the size of a data item, if you make it larger than originally designed.
- Applying any custom mappings that are required.

Manually mapping items in the InForm Architect application to the corresponding Clintrial item gives you flexibility. For example, in some instances it may be desirable to map two items in an InForm Architect form to the same item in the Clintrial software (mutually exclusive items in a radio group should only require one item). Alternatively, you may want to group clinical data items from one or more forms in an InForm trial into a single Clintrial panel.

Clintrial mapping and generic visits

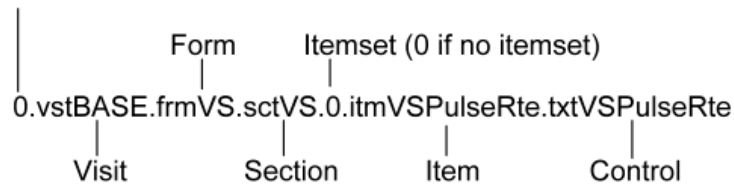
Clintrial mapping definitions include the control path for each data source. If a form occurs in multiple visits, a Clintrial mapping definition is necessary for each control in each visit where it occurs, each mapping definition differing from the others only in the visit name. If you generate mappings for all of the data points in a trial, a large number of almost identical mappings can result, creating a heavy and error-prone maintenance burden.

To reduce the number of Clintrial mapping definitions required, the InForm application and the InForm Architect application support the concept of *generic visits*. A generic visit is a reserved MedML visit RefName (“PF_ALL_VISITS”) used in a Clintrial mapping definition. The generic visit RefName indicates that the control is a source data point for a Clintrial table in every visit in which it occurs. In the following example, as in the Data Mappings window, a Clintrial mapping definition is represented by a string of RefNames connected by periods.

The VITAL SIGNS (VS) form occurs in almost every visit in the PFST sample trial—that is, in vstBASE, vstCORE1, vstCORE2, vstCORE4, vstCORE8, and vstUnschVisit. To generate Clintrial mapping definitions for the txtVSPulseRte control by using explicit visits, you would need to create six definitions—one for each visit in which the form appears. Using the concept of the generic visit, only a single definition is necessary.

Clintrial mapping definitions with explicit visits

Patient (always 0)



```
0.vstBASE.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
0.vstCORE1.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
0.vstCORE2.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
0.vstCORE4.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
0.vstCORE8.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
0.vstUnschVisit.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
```

Clintrial mapping definition with generic visit

Generic visit (always 0)

```
0.0.frmVS.sctVS.0.itmVSPulseRte.txtVSPulseRte
```

When you create a Clintrial database with automatically generated mapping, the InForm Architect application creates mappings that specify the generic visit by default. When you create a Clintrial database with manual mapping, you can specify the generic visit when you create a mapping definition for a form control.

Setting up Clintrial data transfer processing

Setting up data transfer to a Clintrial database involves the following steps:

- 1 Define Clintrial data mappings through the InForm Architect application.
- 2 Create a connection for data transfer.

Mapping considerations—InForm application to Clintrials software

InForm metadata objects correlate to Clintrials metadata objects..

Mapping definitions specify how InForm metadata objects correlate to Clintrials metadata objects. The following table describes considerations to be aware of when mapping metadata from an InForm trial to a Clintrials protocol in a hybrid trial.

InForm metadata	Considerations
Association	Associations between forms have no direct corollary in the Clintrials software.
Checkbox group	Maps to a single item for each check box and one additional item for each Other field associated with a check box.
Date field	<p>The following options are available for mapping date fields:</p> <ul style="list-style-type: none"> You can specify whether to map a complete InForm Architect date component to a Clintrials date field, individual string fields, or both. You can specify whether to map an incomplete InForm Architect date component to a single Clintrials string field, individual string fields, or both. If you split the individual parts of the date into separate columns in the Clintrials database, you can choose whether to map them to text or fixed data types.
Design note	<p>In the InForm Architect application, the definition of any type of trial component can include design notes. These design notes are not translated to the Clintrials protocol.</p> <p>However, when generating mappings for InForm items, the mapping generator populates the Design Note property of the CTITEM mapping definition with the text of the Question property of the item. This value is inserted into the Description field for the Clintrials item.</p>
Form	<p>Maps to a:</p> <ul style="list-style-type: none"> Page template (hybrid trial only, metadata element may not be defined in an EDC-only trial). Panel (if mappings created in the InForm Architect application using automatic mappings). <p>Special mapping considerations apply if page templates are not unique within the study book. For more information, see <i>Using page keys and block keys</i> (on page 227).</p>
Item	Maps to a Clintrials item. To mark an item as required, you must set the Item Required property for the item in the InForm Architect application.

InForm metadata	Considerations
Item question	Maps to a field label (hybrid trial only; metadata element may not be defined in an EDC-only trial).
Itemset	Each item within an itemset maps to a single item within the panel. Itemsets in the InForm Architect application derived from page sections in a Clintrial protocol correspond to repeating items in the Clintrial software (page section with the Has Repeats attribute set).
Pulldown control	Maps to a single item within the panel. The item stores the string value associated with the pulldown control.
Radio button	Maps to a single item within the panel. The item stores the numeric value associated with the radio group. The mapping generator automatically calculates the maximum string length of each radio group control.
Radio group/Pulldown consisting of a group of radio buttons or a drop down list and one or more Other fields.	Maps to one item representing the radio group or pulldown list and an additional number of items representing the Other fields.
Repeating form	Corresponds to a single panel with items. The FormIndex value in the Clintrial database table holds form instance information (numeric value). A repeating form in the InForm Architect application derived from a page template in a Clintrial protocol corresponds to a page template with the Has Repeats attribute set.
Repeating visit	Single panels, that comprise forms within visit, contain a VisitIndex value. The Clintrial database table (panel) holds visit instance information (numeric value).
Required item	Items marked as required in the InForm Architect application do not translate to required items in the Clintrial software. To force an item to be required in the Clintrial software, set the Item Required property for the item to True in a mapping definition.
Rule	Corresponds to a rule or derivation in the Clintrial software. Rules are maintained separately in the InForm application and the Clintrial software, but the rules created in each application can affect data in the other: <ul style="list-style-type: none"> • CIS passes Clintrial rules and derivations through mappings from the InForm application to the Clintrial software. • Clintrial rules and derivations can run against EDC data in the Clintrial protocol database.
Section	Page section (panels Types 1, 2, 3, 4). Hybrid trial only, metadata element may not be defined in an EDC-only trial.
Study version	Corresponds to a study book (hybrid trial only; metadata element may not be defined in an EDC-only trial).

InForm metadata	Considerations
Study (Trial)	Corresponds to a protocol (Type 1 - or clinical data protocols - only).
Unit	Controls in the InForm Architect application can be assigned a units property. The units information is not transferred to the Clintrial protocol.

Autogenerating Clintrial mappings

Overview of autogenerating mappings

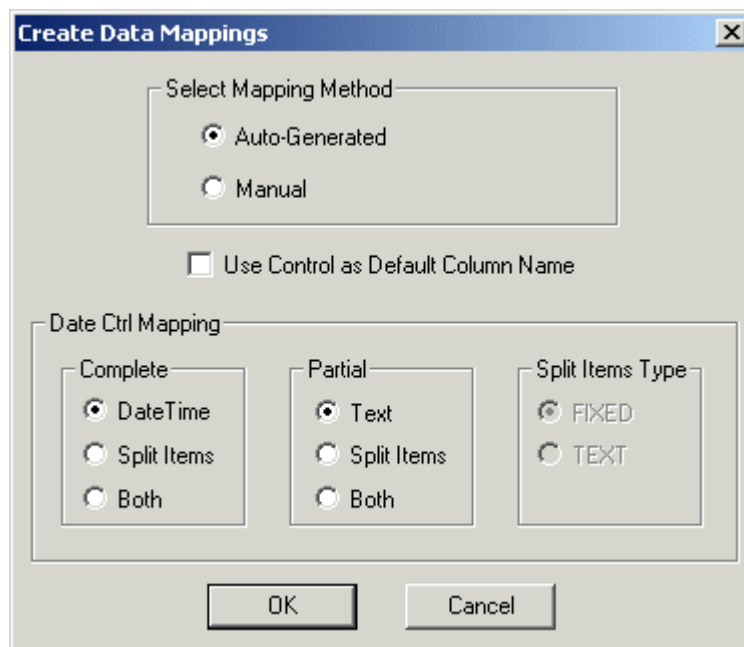
You can automatically or manually create mappings for all objects in an integrated trial. You can always modify the mappings once they are initially created.

Creating an autogenerated mapping definition

To autogenerate Clintrial mappings, use the InForm Architect application as follows:

- 1 In the InForm Architect **Trial Objects** window, select the **Data Mappings** entry.
- 2 Right-click, and from the right-click menu, select **Create Clintrial**.

The Create Data Mappings dialog box appears.



- 3 In the **Select Mapping Method** group, select one of the following:
 - Select the **Auto-Generated** option to automatically generate data mappings.

Select the **Use Control as Default Column Name** checkbox if you prefer item names to reflect control RefNames. If you leave this unselected, the item names reflect the concatenated RefNames of the control path (i.e. form_section_item).
 - Select the **Manual** option to generate mappings manually. In this instance the **Use Control as Default Column Name** checkbox is grayed out. The mappings generated only reflect those of the context panel. You can generate all other mappings manually.
- 4 In the **Date Ctrl Mapping** group, specify how you want complete or partially complete date time items to be mapped. For more information, see *Mapping date time controls—principles* (on page 200) and *Mapping date time controls—automatic mappings* (on page 202).

- 5 Click **OK**.

A new mappings object, with a generic name (for example, mappings1), appears under the Data Mappings icon in the Trial Objects window. You can rename the mappings object by selecting it and modifying its Name property in the Properties window.

Note: You must install the mappings into the InForm trial before you can synchronize with the Clintrial protocol.

Mapping date time controls—Principles

How CIS synchronizes date time data to Clintrial database columns depends on how you define the controls and how you define the mappings for the controls in the InForm Architect application.

The combination of these specifications provides flexibility in the storage of date time data in the Clintrial database. This section describes:

- Clintrial data types used for mapping date time data.
- Differences in how CIS handles complete and partial date time data, and how to define controls that CIS will consider complete.

The following table describes the Clintrial data types that are used for date time data.

Clintrial data types for dates	Description
DATE	This data type holds a complete date, including month, day, and year. The DATE data type corresponds to the Oracle DATE database format. If you map a complete InForm date time including hours, minutes, seconds to a DATE data type, CIS drops the time portion from the date time when synchronizing.
DATETIME	This data type holds a complete date time, including month, day, year, hour, minute, and second. The date and time portions are separated by a space. The DATETIME data type corresponds to the Oracle DATE database format.
FIXED	This data type is an integer, corresponding to the Oracle NUMBER(xx) database format.
TEXT	This data type is a string, corresponding to the Oracle VARCHAR2(n) database format.

When synchronizing date time data to Clintrial, CIS considers it to be complete or partial:

- **Complete**—A date or date time in which all parts are certain to be present. By default, CIS maps a complete date to a Clintrial DATE data type, and maps a complete date time to a DATETIME data type.

To ensure that CIS considers a date time control complete and synchronizes it as a single data item to a DATE or DATETIME column, you must define the control with the Required property set to “True” for all relevant date time parts:

- For a complete *date*, set the Required property to True for the Month, Day, and Year date time parts.
- For a complete *date time*, set the Required property to True for all date and time parts.
- **Partial**—All other dates, times, or date times. By default, CIS maps a partial date time control to a Clintrial TEXT data type.

If you define any part of a date time control with the Unknown property set to “True,” allowing users to specify UNK as a value of the date time part, CIS treats the control as a partial date, time, or date time.

You can map complete or partial date time controls to any of the following:

- A single Clintrial database column.
- A separate Clintrial database column for each date time part.
- A column for the whole control and a column for each part.

The following table gives examples of how CIS maps complete, partial and split date parts to the columns defined with the various Clintrial data types.

Mapping definition	Date time part property in InForm Architect application	Value entered by InForm user	Value in Clintrial table column
Complete date mapped to DATE data type	Required = True; date time definition includes month, day, year	Jun 8 2004	6/8/2004
Complete date mapped to DATETIME data type	Required = True; date time definition includes month, day, year	Jun 8 2004	6/8/2004 00:00:00
Partial date mapped to TEXT data type	Required = False; date time definition includes month, day, year, hour, minute	Jun 8 2004	6/8/2002 12:00 AM
	Required = False; date time definition includes month, day, year, hour, minute Allow Unknown = True	UNK UNK UNK UNK UNK	UNK/UNK/UNK UNK:UNK:NUL

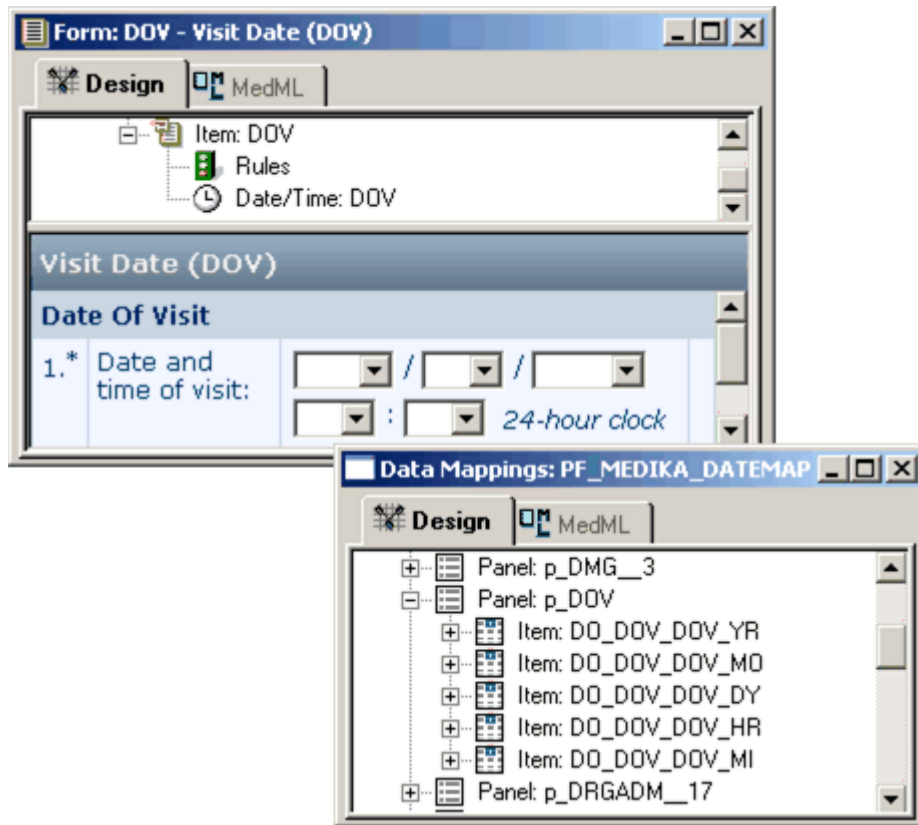
Mapping definition	Date time part property in InForm Architect application	Value entered by InForm user	Value in Clintrial table column
	Required = False: date time definition includes month, day, year	(blank) (blank) 2004	NUL/NUL/2004 NUL:NUL:NUL
Split date part mapped to FIXED data type	Required = False Allow Unknown = True	Some date parts blank Some date parts UNK	Blank or UNK date parts synchronize as integer -99
Split date part mapped to TEXT data type	Required = False Allow Unknown = True	Some date parts blank Some date parts UNK	Blank or UNK date parts synchronize as string -99

Mapping date time controls—Automatic mappings

The following table indicates the settings to select in the Date Ctrl Mapping group of the Create Data Mappings dialog box to generate each type of available mapping.

To generate this mapping...	Select...
Complete date maps to a single DATE column; complete date time maps to a single DATETIME column	DateTime in the Complete group.
Complete date or date time maps to multiple columns	Split Items in the Complete group. In the Split Items Type group, specify whether to map the date time parts to FIXED or TEXT columns.
Complete date or date time maps both to a single DATE column and to multiple FIXED or TEXT columns	Both in the Complete group. In the Split Items Type group, specify whether to map the date time parts to FIXED or TEXT columns.
Partial date, time, or date time maps to a single TEXT column	Text in the Partial group.
Partial date, time, or date time maps to multiple columns	Split Items in the Partial group. In the Split Items Type group, specify whether to map the date time parts to FIXED or TEXT columns.
Partial date, time, or date time maps both to a single TEXT column and to multiple FIXED or TEXT columns	Both in the Partial group. In the Split Items Type group, specify whether to map the date time parts to FIXED or TEXT columns.

The following figure illustrates the automatic mapping of a date time control into separate columns.



Mapping new forms to an autogenerated mapping definition

When you create a new form in the InForm Architect application or load an XML file containing a form definition, the form is added to the list of forms in the Trial Objects window so that you can add to or update its design.

To map the form and its controls to an automatically-generated Clintrial mapping definition:

- 1 Open the **Form** window in the Design Workspace.
- 2 Right-click the **Form RefName**, and select **Add to Auto-Generated Data Mappings**.

The InForm Architect application generates table, table column, and control path mappings for the components of the form definition in each automatically-generated mapping definition in the trial.

Creating manual Clintrial mapping definitions

To set up data mapping definitions for a Clintrial database, create the following definitions:


- 1 Clintrial mapping object.
- 2 Panels.
- 3 Items for each panel.
- 4 Control path for each item.

Creating a Clintrial mapping definition object

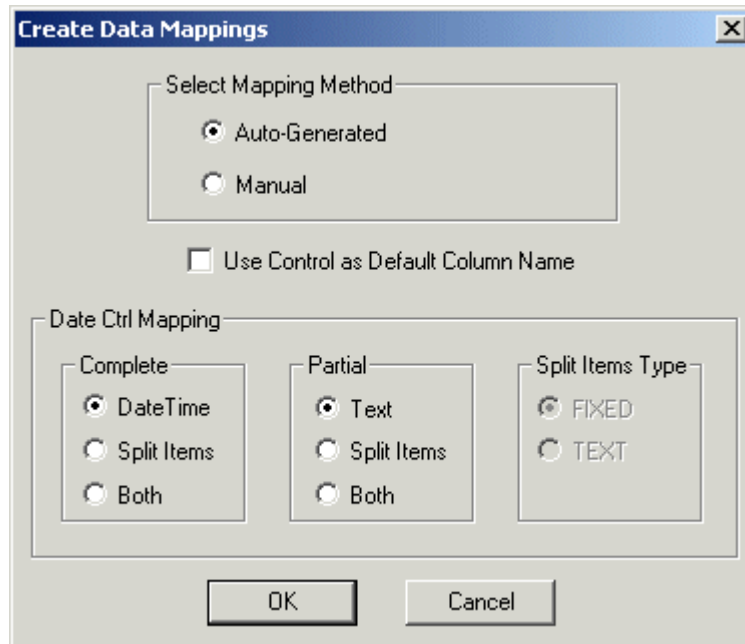
In the InForm Architect application, you can create a data mapping definition by using the data mappings menu or toolbar commands or by loading an XML file containing MedML tags that specify a data mapping definition.

Note: Oracle recommends that you create data mapping definitions only after installing your completed trial definition in the strict mode of the MedML Installer tool. This insures that your component definitions are complete and all dependencies among definitions are satisfied. If you do not use strict mode (the default MedML Installer mode), any incomplete component definitions that nonstrict mode allows to process are carried over into the data mapping definition.

To create a Clintrial mapping definition object:

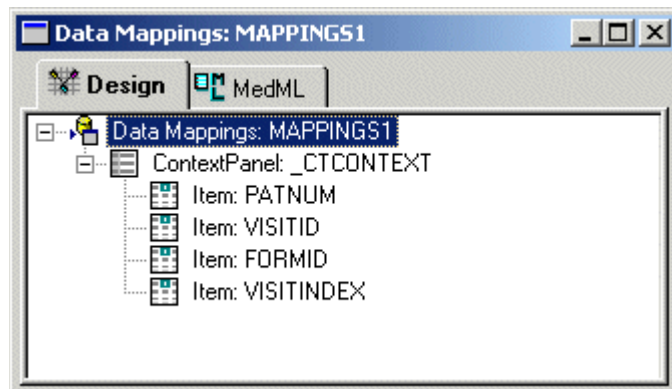
- 1 Do one of the following:
 - Select **Trial > Create Data Mappings > Clintrial**.
 - In the **Trial** toolbar, click the **Data Mappings** button . In the **Data Mappings Type** dialog box, select **Clintrial** and click **OK**.
 - In the **Trial Objects** window, right-click the **Data Mappings** node and select **Clintrial** from the right-click menu.

The Create Data Mappings dialog box appears.

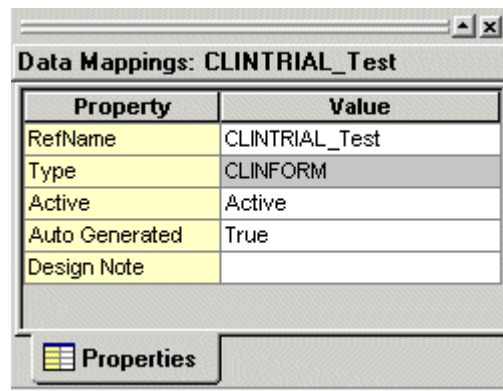


- 2 Select **Manual** and click **OK**.

The Data Mappings window opens in the Trial Design workspace. The mapping definition generated by the InForm Architect application includes the definition of a context panel along with the standard context panel items, PATNUM, VISITID, FORMID, VISITINDEX, and FORMINDEX.



- 3 Edit the Clintrial mapping property values in the **Properties** window. For more information, see *Mapping definition object properties* (on page 206).



Mapping definition object properties

The following table describes the properties of a Clintrial mapping definition.

Property	Description
RefName	RefName of the component. REQUIRED.
Type	CLIFORM. READ-ONLY.
Active	Active or Inactive , indicating whether the Clintrial mapping definition is accessible with the InForm Architect application. The default is Active. To disable a Clintrial mapping definition, set the value to Inactive and install the definition by saving it with the Install MedML When Saving option enabled or by using the MedML Installer utility. The Clintrial definition becomes inaccessible with the InForm Architect application. OPTIONAL.
Auto Generated	True or False , indicating whether the Clintrial mapping is in Auto-Generate mode. In this mode, when you add a new control to a form in the Form window, the InForm Architect application automatically generates a Clintrial mapping for it at the end of the list of column definitions in the target table. REQUIRED..
Design Note	Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only. OPTIONAL.

Deleting a data mapping definition component

To delete any component of a data mapping definition:

- 1 Open the **Data Mappings** window for the data mapping definition you want to edit by double-clicking its icon in the **Trial Objects** window.
- 2 Expand the mapping definition nodes as necessary to expose the component you want to update.
- 3 Right-click the component.
- 4 From the right-click menu, select the applicable **Delete** command.

When you delete a component that has child components, all of the children are deleted along with the parent. For example, when you delete a table definition in a CDD mapping definition, all of its column and control definitions are deleted as well.

Note: To undo a deletion, select **Undo** from the **Edit** menu or click the **Undo** button on the **Edit** toolbar. If you delete a component with multiple children, each **Undo** operation restores one child component. To restore all child components, perform the **Undo** operation as many times as there are child components.

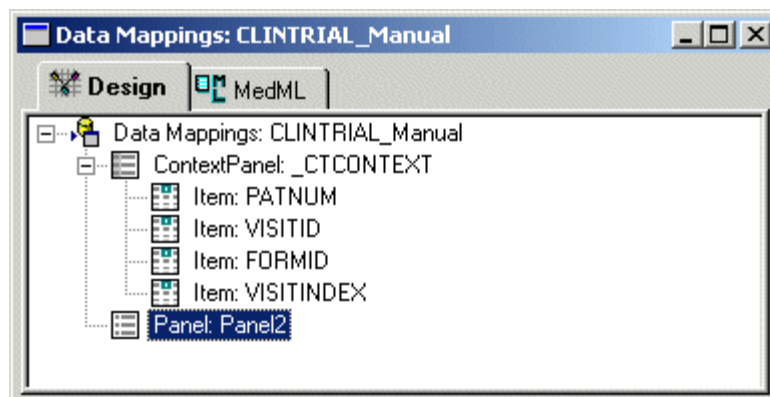
Creating a panel mapping definition

A Clintrial panel represents a set of related items. When you autogenerate a Clintrial mapping, each form in an InForm trial corresponds to one Clintrial panel.

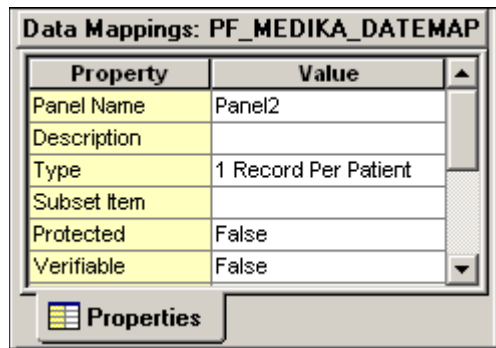
To create a panel mapping definition manually:

- 1 Open the **Data Mappings** window for the mapping in which you want to create a new panel mapping.
- 2 Right-click the **Data Mappings** node and select **New Panel** from the right-click menu.

The InForm Architect application generates a panel definition.



- 3 Edit the Clintrial panel property values in the **Properties** window. For more information, see *Panel mapping properties* (on page 208).



Panel mapping properties

The following table describes the properties of a Clintrial software panel mapping definition.

Property	Description
Panel Name	Name of the panel. REQUIRED..
Description	Description of the panel. REQUIRED..
Type	A specification of how the database tables associated with the panel are structured: <ul style="list-style-type: none"> • One record per patient • One record per patient visit • More than one record per patient • More than one record per patient visit • Enrollment panel REQUIRED..
Subset Item	Name of the item specified as the subset key for subset page sections based on the panel. OPTIONAL. A subset page section can occur multiple times on a trial page in the following types of panels, with each value of the subset key item representing distinct rows (subsets) of data: <ul style="list-style-type: none"> • More than one record per patient • More than one record per patient visit For more information, see <i>Implementing Clintrial subsets</i> (on page 231).
Protected	True or False, indicating whether access rights to the panel are limited in the Clintrial software. False is the default. OPTIONAL.

Property	Description
Verifiable	True or False, indicating whether double-entry of data in panel items is required for verification. False is the default. OPTIONAL.
Detail	True or False (default), indicating whether the panel definition participates in a detail page section in a master-detail relationship. A master-detail relationship is a relationship between two page sections on a study page, in which each record in one page section (the master page section) can have one or more associated records in the other section (the detail page section). During data entry the displayed records in the detail page section are associated with the selected record in the master page section. REQUIRED..
Detail CTItem	Name of the item identified as the detail key item, if the panel definition is part of a detail page section. OPTIONAL.
Master Panel	Panel name of the master panel with which this panel definition participates in a master-detail relationship. This property applies only if the Is Detail property is True. OPTIONAL.
Master Item	Name of the item on the associated master panel that corresponds to the detail key item specified in the Detail CTItem property. This property applies only if the Is Detail property is True. OPTIONAL.
SAS Name	Name of the panel when data is sent to SAS through the Clintrial SAS interface. OPTIONAL; if entered, the name must be 8 characters or fewer and conform to SAS naming requirements. Note: Panel SAS names must be unique within a protocol. You must make sure of their uniqueness, as the CIS software does not enforce this restriction.
Lock Status	Indicates whether the protocol in which the panel is included is locked: <ul style="list-style-type: none"> • The panel is not modifiable and cannot be made modifiable. • The panel is modifiable. • The panel is not modifiable, but it can be reset to modifiable.

Creating a panel item mapping definition

Creating a Clintrial panel item mapping definition with the InForm Architect application has two parts:

- 1 Defining an item in a Clintrial panel.
- 2 Identifying the form controls that map to the item.

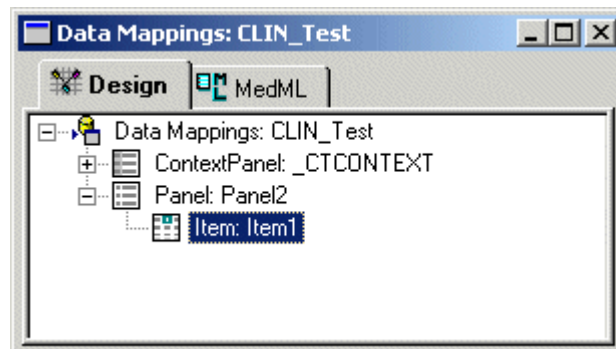
You can do these activities as separate steps, or you can create both the item and the control mapping in a single step.

Creating only an item definition

To create an item definition that does not include a control mapping:

- 1 In the **Data Mappings** window, right-click the name of the panel in which you want to create a new item.
- 2 From the right-click menu, select **New Item Definition**.

InForm Architect application adds a new item line to the panel node in the Data Mappings window.



- 3 Edit the item mapping properties in the **Properties** window. For more information, see *Item mapping properties* (on page 215).

Creating an item definition and mapping a control

To create an item definition and control mapping in a single step, you can do either of the following:

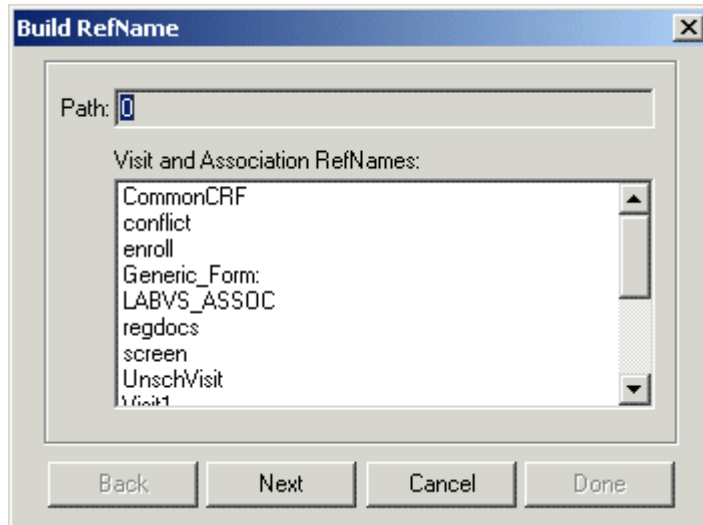
- **In the Data Mappings window**, use the New Item right-click menu command for the table in which you want to create the column and control definition. For more information, see *Using the Data Mapping window to create an item definition and control mapping* (on page 211).
- **In the Form window**, use the Add Data Mapping right-click menu command for the control you want to map, or you can drag the control from the Form window onto the Data Mappings window. For more information, see *Using the Form window to create an item definition and control mapping* (on page 212).

Using the Data Mappings window to create an item definition and control mapping

To create the item definition and control mapping using the Data Mapping window:

- 1 In the **Data Mappings** window, right-click the name of the panel in which you want to create a new item.
- 2 From the right-click menu, select **New Item**.

The Build RefName dialog box appears.



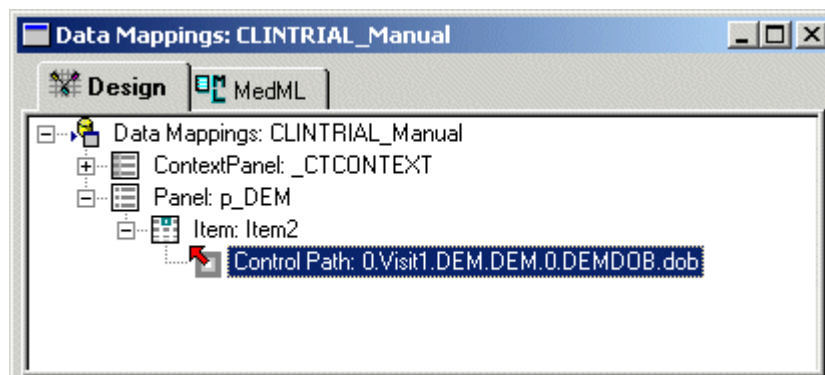
- 3 For each component of the RefName path, select the component from the list in the dialog box. To specify that you want the control to be mapped from all visits in which it occurs, select **Generic_Form** as the visit RefName.
- 4 Click **Next**, or, to backtrack to a previous component, click **Back**.

The InForm Architect application builds the path from your selections and displays it in the **Path** field at the top of the dialog box.

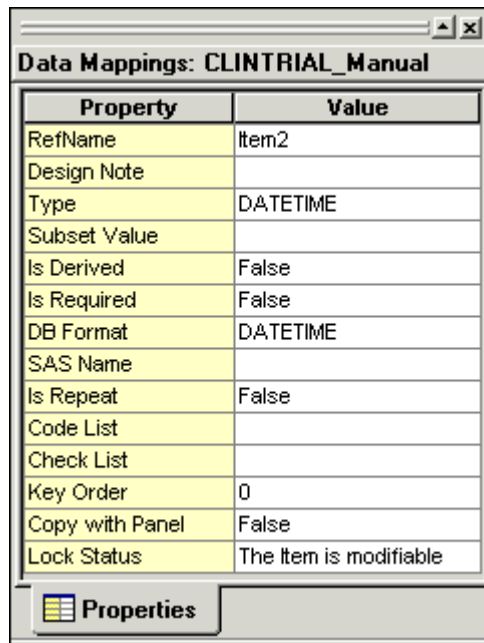
- 5 When the RefName path is complete, click **Done**.

The InForm Architect application inserts the path into the component definition you are creating.

- 6 Repeat this procedure as often as necessary to create additional item mappings for the child controls of group controls such as radio buttons. To view control path properties for any path, click the path in the **Data Mappings** window.



- Specify the properties of the item by clicking the item icon and editing the values in the **Properties** window. For more information, see *Item mapping properties* (on page 215).



Property	Value
RefName	Item2
Design Note	
Type	DATETIME
Subset Value	
Is Derived	False
Is Required	False
DB Format	DATETIME
SAS Name	
Is Repeat	False
Code List	
Check List	
Key Order	0
Copy with Panel	False
Lock Status	The Item is modifiable

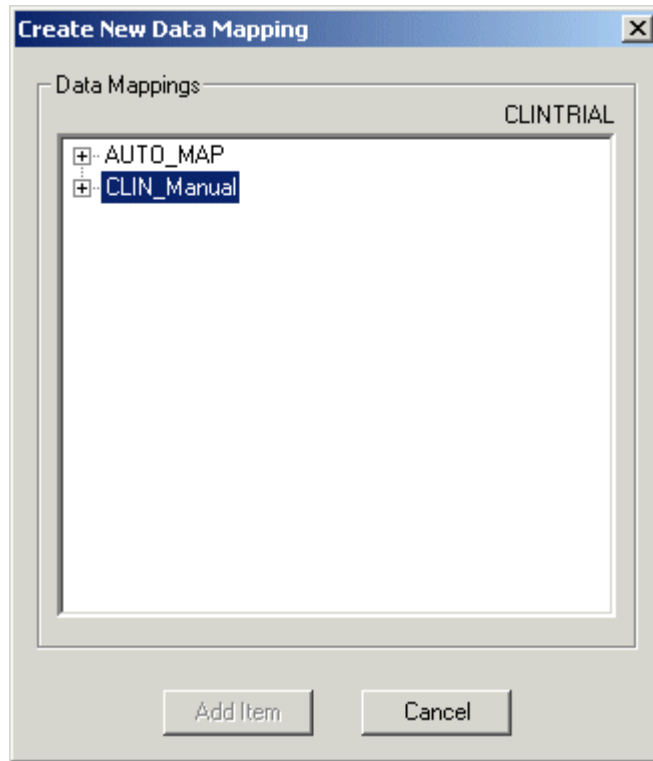
Using the Form window to create an item and control mapping definition

You can create an item and control mapping definition from the Form window. However, before you do, the panel in which you want to create the item and control definition must exist in the Clintrial mapping.

To create an item and control mapping definition:

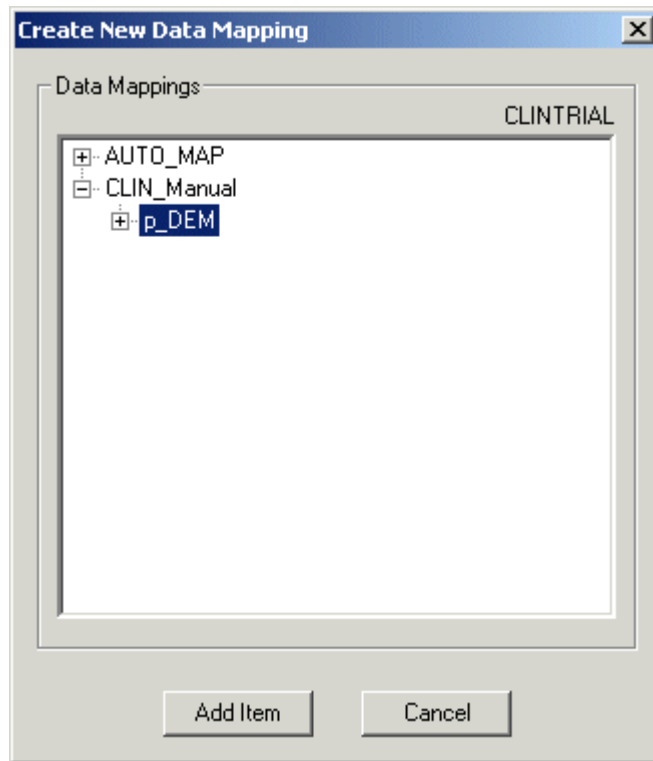
- Open the **Form** window containing the control you want to map and expand the tree so that the control is visible.
- Right-click the control, and select **Add Data Mapping** from the right-click menu.

The **Create New Data Mapping** dialog box appears.



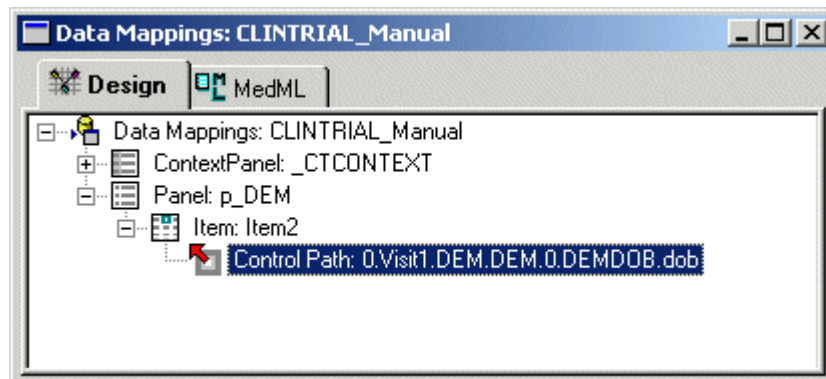
- 3 Expand the tree so that the panel in which you want to create the item definition is visible.
- 4 Select the panel.

The **Add Item** button becomes available.



- 5 Click **Add Item**.

InForm Architect application generates a definition for the item mapping and the control path.



- 6 Specify the properties of the item by clicking the item icon and editing the values in the **Properties** window. For more information, see *Item mapping properties* (on page 215).

You can also quickly generate a definition by dragging the control from the Form window onto the Data Mappings window.

Note: The InForm Architect application only allows you to drag and drop controls for which you can legitimately create mappings. You cannot drag and drop a simple control or a group control.

To drag a control from the Form window to the Data Mappings window:

- 1 Open the **Form** window containing the control you want to map and expand the tree so that the control is visible.
- 2 Open the **Data Mappings** window and expand the definition so that the table in which you want to create the column definition is visible.
- 3 Drag the control from the **Form** window and drop it onto the table in the **Data Mappings** window.
- 4 Specify the properties of the item by clicking the item icon and editing the values in the **Properties** window. For more information, see *Item mapping properties* (on page 215).

Item mapping properties

The following table defines the properties of an item mapping definition:

Property	Definition
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
Type	Data type for the value of the item: <ul style="list-style-type: none"> • DATE • DATETIME • FIXED • FLOAT • TEXT REQUIRED..
Length	Number of characters that can be entered in this item. The default is 38. OPTIONAL. This property is visible when the Type is TEXT, FIXED, or FLOAT.
Precision	Number of characters that can be added after the decimal place in an item with a type of FLOAT. This value can be a number between 1 and 15. The default is 10. If you enter 0, the precision defaults back to 10. If you enter a number greater than 15 is chosen, the precision defaults back to 15. OPTIONAL. This property is visible when the Type is FLOAT.

Property	Definition
DB Format	<p>Format in which Clintrial software stores values for the item in the Oracle database, for example:</p> <ul style="list-style-type: none"> • VARCHAR2(n) • DATE • DATETIME • NUMBER(xx) • NUMBER(xx,yy) • NUMBER(xx,0).x <p>READ ONLY.</p>
Date Part	<p>Part of a date time control to be mapped to a Clintrial software item. Use this property if you are mapping parts of a date time control to separate Clintrial software items. Values are:</p> <ul style="list-style-type: none"> • Year • Month • Day • Hour • Minute • Second • Undefined <p>OPTIONAL.</p>
Derived	<p>True or False, indicating whether the value of the item is determined from a derivation associated with the panel. False is the default.</p> <p>OPTIONAL.</p>
Required	<p>True or False, indicating whether the item is required. False is the default.</p> <p>OPTIONAL.</p>
SAS Name	<p>Name of the item when data is sent to SAS through the Clintrial SAS interface.</p> <p>OPTIONAL; if entered, the name must be eight characters or fewer and conform to SAS naming requirements.</p>
Repeat	<p>True or False, indicating whether an item is one for which multiple values can be entered within a page section. False is the default.</p> <p>OPTIONAL.</p>

Property	Definition
Code List	<p>Name of a Clintrial codelist associated with the item. A codelist encodes entered values. Only codes or values in the codelist can be entered as values of the item.</p> <p>OPTIONAL; Code List and Check List are mutually exclusive.</p>
Check List	<p>Name of a Clintrial checklist associated with the item. A checklist is a type of codelist used to view suggested entries for a field.</p> <p>OPTIONAL; Code List and Check List are mutually exclusive.</p>
Range Lower Bound	<p>Minimum value that can be entered for the value of the item.</p> <p>OPTIONAL. This property is visible if the Type is FIXED or FLOAT.</p>
Range Upper Bound	<p>Maximum value that can be entered for the value of the item.</p> <p>OPTIONAL. This property is visible if the Type is FIXED or FLOAT.</p>
Key Order	<p>The order in which the item appears in the concatenation of key items, if the item is part of the panel's key. 0 (not a key item) is the default.</p> <p>OPTIONAL.</p>
Copy with Panel	<p>Name of a Clintrial codelist associated with the item. A codelist encodes entered values. Only codes or values in the codelist can be entered as values of the item.</p> <p>OPTIONAL.</p>
Lock Status	<p>Indicates whether the protocol in which the item is included is locked:</p> <ul style="list-style-type: none"> • The item is not modifiable and cannot be made modifiable • The item is modifiable • The item is not modifiable, but it can be reset to modifiable <p>OPTIONAL.</p>
Design Note	<p>Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only.</p> <p>OPTIONAL.</p> <p>Design Note text is not transferred to the target tables.</p>

Creating a control path mapping definition

If you want to create new or additional mappings to an existing item definition, you must create a control mapping definition. This definition consists of the RefName path of the control, which is displayed but cannot be edited in the Properties window.

To create a control mapping, you can do either of the following:

- **In the Data Mappings window**, select the Add Control right-click menu command for the item in which you want to create the control path. For more information, see *Using the Data Mappings editor window to create a control mapping* (on page 181).
- **In the Form window**, select the Add Data Mapping right-click menu command for the control you want to map, or you can drag the control from the Form window onto the Data Mappings window. For more information, see *Using the Form window to create a control mapping* (on page 182).

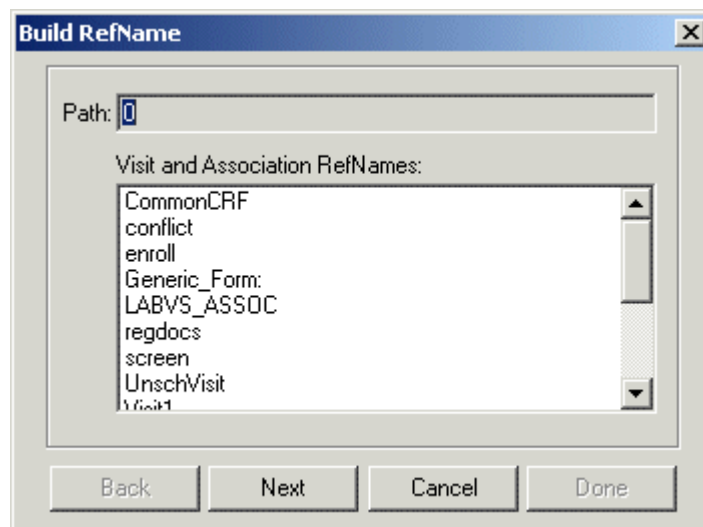
When you create a control mapping, The InForm Architect application generates control paths that use the Generic_Form visit type.

Using the Data Mappings window to create a control mapping

To create a control mapping definition In the Data Mappings window:

- 1 Right-click the name of the column in which you want to create a new control mapping.
- 2 From the right-click menu, select **Add Control**.

The Build RefName dialog box appears.



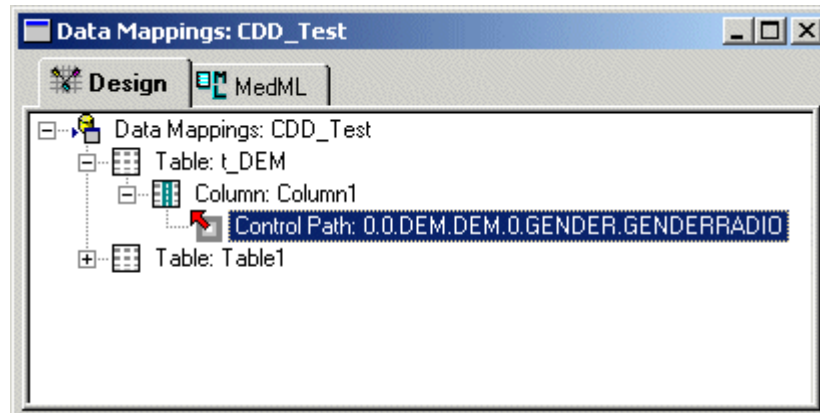
- 3 For each component of the RefName path, select the component from the list in the dialog box. To specify that you want the control to be mapped from all visits in which it occurs, select **Generic_Form** as the visit RefName.

- Click **Next**, or, to backtrack to a previous component, click **Back**.

The InForm Architect application builds the path from your selections and displays it in the **Path** field at the top of the dialog box.

- When the RefName path is complete, click **Done**.

The InForm Architect application inserts the path into the component definition you are creating.



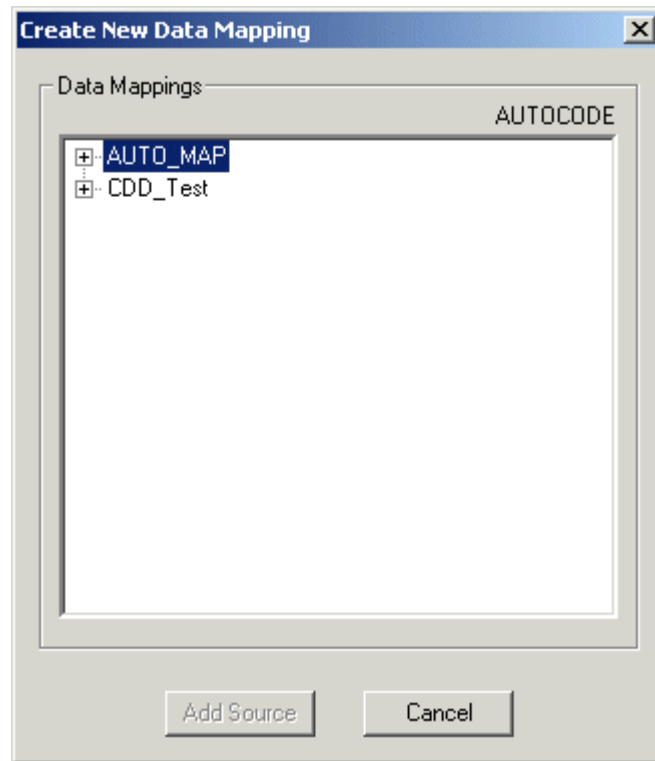
- Edit the control mapping properties in the **Properties** window. For more information, see *Control path mapping properties* (on page 222).

Using the Form window to create a control mapping

To create a control mapping definition from the Form window:

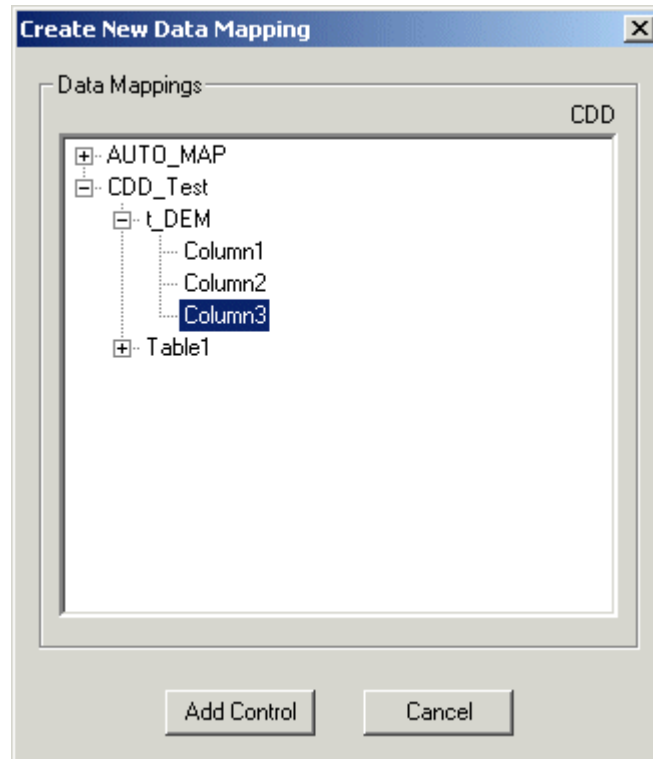
- Open the **Form** window containing the control you want to map and expand the tree so that the control is visible.
- Right-click the control, and select **Add Data Mapping** from the right-click menu.

The Create New Data Mapping dialog box appears.



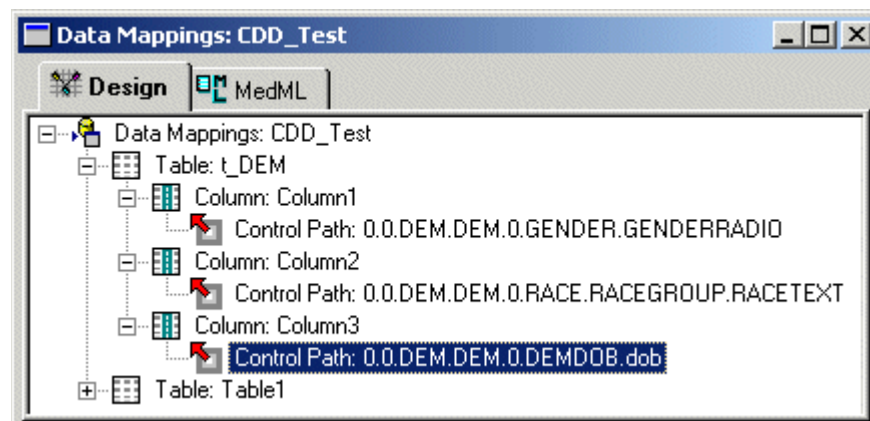
- 3 Expand the tree so that the column in which you want to create the control mapping definition is visible.
- 4 Select the column.

The Add Control button becomes available.



- 5 Click **Add Control**.

The InForm Architect application generates a definition for the control path.



- 6 Edit the control mapping properties in the **Properties** window. For more information, see *Control path mapping properties* (on page 222).

You can also quickly generate a definition by dragging the control from the Form window onto the Data Mappings window.

Note: The InForm Architect application only allows you to drag and drop controls for which you can legitimately create mappings. You cannot drag and drop a simple control or a group control.

To drag a control from the Form window to the Data Mappings window:

- 1 Open the **Form** window containing the control you want to map and expand the tree so that the control is visible.
- 2 Open the **Data Mappings** window and expand the definition so that the table in which you want to create the column definition is visible.
- 3 Drag the control from the **Form** window and drop it onto the table in the **Data Mappings** window.

Control path mapping properties

The following table describes the properties of a Clintrial control path mapping definition.

Property	Definition
Control Path	RefName path of the control path mapping. READ-ONLY.
BlockKey Value	Value of the Clintrial block key. If you specify this value, it overrides the visit RefName as the block key. For more information, see <i>Using page keys and block keys</i> (on page 227). OPTIONAL.
PageKey Value	Value of the Clintrial page key. If you specify this value, it overrides the form RefName as the page key. For more information, see <i>Using page keys and block keys</i> (on page 227). OPTIONAL.
Subset Value	Value the item takes if it is the subset key for subset page sections based on the panel. For more information, see <i>Implementing Clintrial subsets</i> (on page 231). OPTIONAL.

Customizing Clintrial mappings

Mapping definitions enable you to store data in specific ways in the Clintrial software or to enable Clintrial features through synchronization. The following trial component specifications can require mapping customizations:

- Autogenerated codelists.
- CONTEXT panel.
- Date time controls.
- Mappings without a control path.
- Item order in a panel.
- Page keys and block keys.
- Panel properties.
- Subsets.
- Type 0 panels.
- Visits and forms.

Autogenerated codelists and mapping definitions

To autogenerate codelists, the CIS software checks:

- The Code List property of an InForm Architect mapping definition for a Clintrial item.
- New mapping definitions for a pulldown or radio group control.

Autogenerated codelists and the Code List property

An InForm Architect mapping definition for a Clintrial item includes a Code List property. This enables a trial designer to specify the name of an existing Clintrial codelist as the target for the data in the item being mapped.

When CIS synchronization processes mapping definitions and autogenerates codelists, it does *not* autogenerate codelists for items that have a value in the Code List property, as the Code List property is intended to refer to an existing codelist. Therefore, if you want CIS to autogenerate a codelist for an item, leave the Code List property blank.

Autogenerated codelists and new mappings

When CIS synchronization processes a new mapping definition for autogenerating codelists, it does the following:

- Ignores the mapping definition if the synchronization connection is configured not to autogenerate codelists.
- Ignores the mapping definition if the Code List property of the item has a value.
- Determines which Clintrial codelist to use for the item:
 - If the item has only one pulldown or radio group control mapped to it, CIS creates a codelist if it does not yet exist and attaches the new or existing codelist for that control to the Clintrial item.
 - If the item has multiple pulldown or radio group controls mapped to it, CIS creates a codelist if it does not yet exist.

CONTEXT panel mappings

CONTEXT panel item updates

During trial design or when a trial is in production, you may need to change the definition of a CONTEXT panel item. For example, if you autogenerate patient numbers in your InForm trial, you may need to increase the size of the SUBJECT item.

You can change the definition of a CONTEXT panel item by changing its mapping definition in the InForm Architect application. As CONTEXT panel items are appended to all Clintrial data tables, you do not specify a control path; you change only the panel item (CTITEM) definition. Changes must follow the standard rules for changing item definitions:

- The new item definition must have the same TYPE property as the previous item.
- If you change the item size, you must change it to a larger size.

When CIS receives a valid new CONTEXT panel item definition during synchronization, it applies the definition to all panels in the Clintrial protocol.

Updating a CONTEXT panel item

To update a CONTEXT panel item:

- 1 Start the InForm Architect application.
- 2 Open the trial containing the mapping definition for the CONTEXT panel item you want to update.
- 3 In the **Trial Objects** window, expand the **Data Mappings** node.
- 4 Double-click the mapping definition.

The Data Mappings window opens in the Design Workspace.

- 5 Expand the **ContextPanel** node, and select the item to update.
The properties of the CONTEXT panel item appear in the Properties window.
- 6 Change the item properties as desired.
- 7 Make sure that the **Strict MedML Checking** button is selected.
- 8 Select **File > Save Component** or **File > Save Component As**, and specify where you want to save the updated mapping definition.

The InForm Architect application saves the mapping definition in the specified location.

Mapping date time controls—Manual mappings

When creating a mapping definition for a date time control, you can map the control to a single Clintrial item with a data type of DATE or TEXT, or you can split the date time control into its component parts and map each one to separate Clintrial items. You can specify either of these options for all date time controls when you autogenerate mappings for a trial. To map a specific date time control to multiple Clintrial items, you must do it manually.

To map a date time control manually:

- 1 Create a new Clintrial mapping item.
- 2 Select the new item in the **Data Mappings** window and update its properties as follows:
 - a Change the **RefName** to an appropriate name for the date time component part.
 - b Change the **Type** to FIXED or TEXT. A warning message indicates a potential incompatibility if controls are already mapped to the item.
 - c Click **Yes** in the message to acknowledge the warning.
 - d In the **Date Part** property, select the date time component for which you are creating a mapping.
 - e If desired, change the **Length** to an appropriate value for the date time component.

Property	Value
RefName	DemDOB_Year
Design Note	
Type	FIXED
Length	4
DB Format	NUMBER(4)
Subset Value	
Date Part	Year
Is Derived	False
Is Required	False
SAS Name	
Is Repeat	False

- 3 Repeat these steps for each part of the date time component for which you want to create a separate Clintrial item mapping.

Creating mappings without a control path

It is possible to create a mapping definition that does not include a control path, either manually or by deleting the control path from the autogenerated mapping definition. When you do this, update the trial, and synchronize to the Clintrial software, CIS optionally creates the panel and item, if necessary. Since there is no control path, there is no InForm item into which you can enter data, and no data is synchronized from the InForm application. Therefore, data never appears in the Clintrial database table.

This may be the desired behavior if the panel is a type 0 panel in which no patient data is stored. For example, you can define such a mapping to create a panel and items for Lab Loader data. To propagate data in the items, you could use the Lab Loader module or define derivations on the items to fill them based on the value of other items. For more information, see *Creating a type 0 panel* (on page 231).

You may also want to create this type of mapping for PDC-only items if you use InForm Architect application as the design tool for Clintrial panels.

Modifying Clintrial Item Order in a panel

In a set of mappings, the order of items within a form determines the default autogenerated Item Order of each item in the corresponding Clintrial panel. Item Order defines the order of the item in Clintrial reports and in the default page section layout.

You can change the order of items within a Panel mapping, by dragging and dropping the Item nodes as desired in the Data Mappings window. This changes the Item Order of the items in the Clintrial panel.

The Item Order in Clintrial software is determined when CIS first creates the item in the Clintrial protocol; CIS does not change Item Order after it has been created. Therefore, if you want to change the order through the data mappings created in the InForm application, you must do it before the item mapping is synchronized to the Clintrial software for the first time. Usually, this is the first time you synchronize the trial to the Clintrial software. (You can still change the order by using the Design module of the Clintrial software.)

Using page keys and block keys

In the Clintrial Design module, you assign each trial page a page key and block key value when you add it to a trial book and block. Page key and block key values determine how the Clintrial software stores data in records for a panel.

For example, suppose a block has two trial pages that contain page sections using items from the same panel. Each page section has a different page key. When a user enters data into each of these page sections, the Clintrial software stores the data in different database records because the page key of each is different.

When CIS synchronizes data from the InForm application to the Clintrial software, it resolves the information in mapping definitions to Clintrial page keys and block keys by doing one of the following:

- Processing customized page keys and block keys from mapping definitions.
- Generating page keys and block keys from form and visit RefNames.

How CIS autogenerates page keys and block keys in the Clintrial software

If you do not specify the page key and block key for a data mapping, CIS uses form and visit RefNames to generate page keys and block keys during synchronization. It processes these RefNames differently based on whether they contain a double underscore:

- **No double underscore**—A visit RefName resolves to a Clintrial block key, and a form RefName resolves to a Clintrial page key. For example, a visit with the RefName VISIT1 synchronizes to a data table in which the block key column value is VISIT1, and the VITALS form synchronizes to a data table in which the page key column value is VITALS.

You can safely let CIS autogenerate page keys and block keys in this way if all of the following are true in your Clintrial protocol:

- Pages are unique within a trial book.
- You use the same naming conventions when assigning RefNames with the InForm Architect application and when assigning page key item values in the Clintrial software.
- The page key column has an alphanumeric data type if the RefNames assigned in the InForm Architect application are textual.
- **Double underscore**—To handle the case where page keys are not unique within a trial book in the Clintrial protocol design, CIS uses a special RefName format in which the RefName is concatenated with the page key item value, separated by a double underscore. For example, the RefName used to map the VITALS form to data tables representing a Clintrial VITALS trial page with a page key item value of 61 would be VITALS__61. During synchronization, the VITALS form synchronizes to a data table in which the page key column is 61.

When you import a Clintrial protocol to a CIS library, CIS automatically generates mappings with RefNames in this format. Additionally, it assigns the value following the double underscore to the BlockKey Value or PageKey Value property of each generated control path mapping definition.

When you create mappings from the InForm Architect application to a previously defined Clintrial protocol in which the naming convention uses different page key item values and page names, you must do one of the following:

- **Before generating mappings**—Manually update visit and form RefNames to include the corresponding block key and page key item value before generating mapping definitions. Use the format *RefName__page-key-item-value*.
- **After generating mappings**—Specify the appropriate block key and page key values as properties of the control path mapping definitions. For more information, see *Customizing page keys and block keys* (on page 228).

Customizing page keys and block keys

You can specify a customized page key and block key for each InForm data point mapped to the Clintrial software by using the Page Key and Block Key properties of an item control path mapping definition. When you specify a Page Key and Block Key value, CIS uses those values instead of autogenerating page key and block key values from the form and visit RefNames.

To customize page keys and block keys for a control path mapping definition:

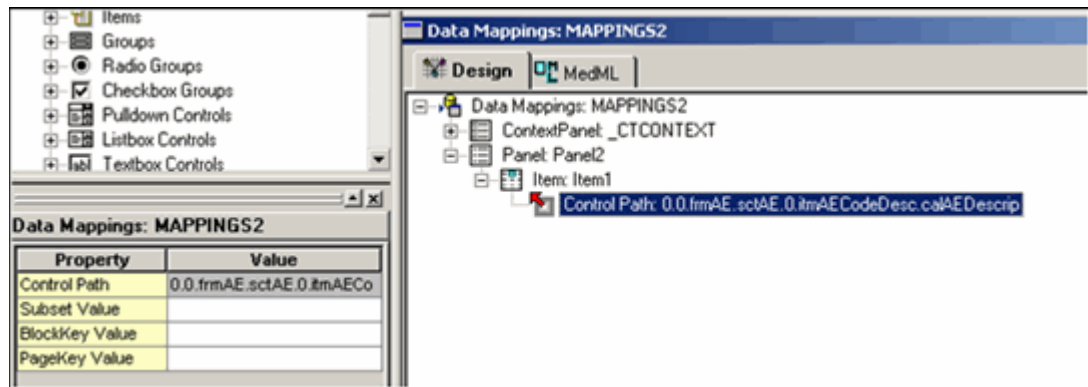
- 1 Using the InForm Architect application, autogenerate mappings for the InForm trial. For more information, see *Autogenerating mappings* (on page 199).

The InForm Architect application generates mappings and includes a node for the mappings definition in the **Trial Objects** tree.

- 2 In the **Trial Objects** tree, double-click the mappings definition node.

The Data Mappings window appears.

- 3 Select the control path definition for which you want to customize the page key and block key.



- 4 In the **BlockKey Value** property, type the value of the block key to which the visit in the control path mapping corresponds.

- In the **PageKey Value** property, type the value of the page key to which the form in the control path mapping corresponds.

Note: Make sure that the length of the BlockKey Value and PageKey Value that you specify does not exceed the size of the visit key and page key columns in the Clintrial CONTEXT panel. Also, make sure that the CONTEXT panel columns have an alphanumeric data type if the BlockKey Value and PageKey Value that you specify include alphabetic characters.

- Save the mapping definition.

Making panel property changes

After a panel is installed in the Clintrial software, you can change the following properties by marking the panel for revision in the Design module:

- SAS Name
- Validation Priority
- Verify
- Protected
- Description

Some of these properties are included in an InForm Architect panel mapping definition. By updating panel mapping properties with the InForm Architect application, you can make changes to a panel definition. After you update a trial with the panel mapping properties changes, CIS synchronizes the changes to the Clintrial software.

Panel properties you can change

The following table lists the Clintrial panel properties you can change through CIS synchronization and the InForm Architect panel mapping definition properties to which they correspond.

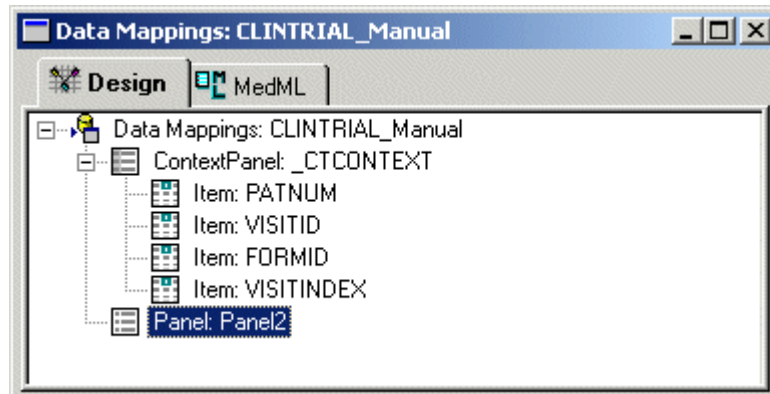
Clintrial panel property	InForm Architect panel mapping property
SAS Name	SAS Name
Verify	Verifiable
Protected	Protected
Description	Description

Changing panel properties

To update Clintrial panel properties:

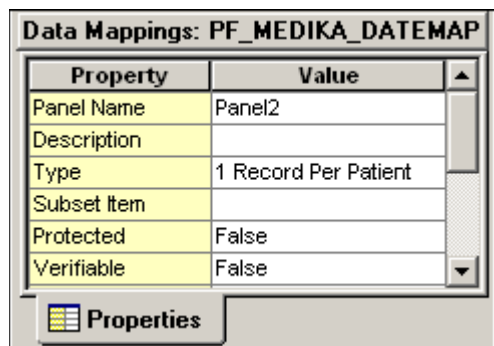
- 1 In the **Trial Objects** tree of the InForm Architect application, double-click the node for the mapping definition containing the panel you want to modify.

The Data Mappings window appears.



- 2 Expand the data mappings tree to expose the panel.
- 3 Select the panel.

The Properties window for the panel appears.



- 4 In the **Properties** window, change the panel properties as desired, and click **Enter**.
The InForm Architect application updates the panel mapping properties.
- 5 Save the mapping definition.

Implementing Clintrial subsets

In an integrated trial, you can use InForm Architect mapping definitions to group data that is entered in an InForm trial into Clintrial subsets.

Subsets in the Clintrial software

In the Clintrial software, subsets provide a way to group similar items together in a page section. A subset page section can occur multiple times on a trial page in a Type 0, Type 2, or Type 4 panel, with each different value of a subset key item representing distinct rows (subsets) of data. Each occurrence of a subset page section constitutes a separate observation.

For example, in the Medika-Clinical sample trial, the LABLNG page template has subset page sections for blood chemistry (LABCHEM), hematology (LABHEM), and urinalysis (LABURN). Each of these sections is made up of the same panel items, including the subset key item TEST_TYPE. The value of TEST_TYPE distinguishes the type of lab test represented by each subset of lab data.

Subset mapping properties

InForm Architect mapping definitions include properties that enable you to specify that CIS should treat certain data items as subset data when synchronizing to the Clintrial software. These properties are:

- **Subset Item**—This property of a panel mapping definition specifies the name of the Clintrial panel item designated as the subset key item. The value of this item determines with which record a group of subset data is stored.

If this property is specified, the item mappings for the panel should include an item with the name specified as the subset item name. No controls from an InForm form should be mapped to that item.

- **Subset Value**—This property of a control path mapping definition specifies the value of the subset key item with which the data in the control is stored. When CIS synchronization stores the data in the control in the Clintrial database, the data goes into a record for which the subset key item has the specified Subset Value.

If a panel has a subset item, every control path mapping to that panel must specify the subset value.

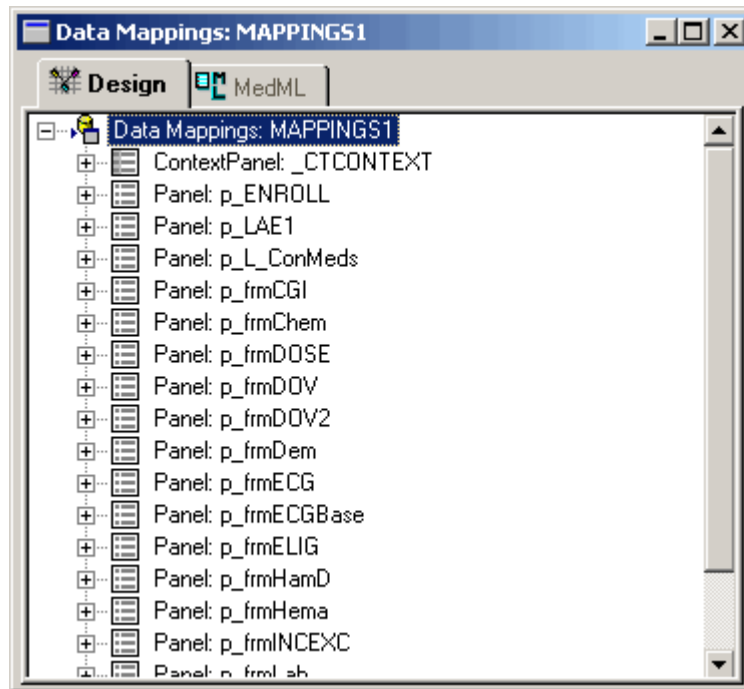
Creating a type 0 panel

Type 0 panels are for storing non-patient data in a Clintrial protocol. For example, the MEDIKA_CLINICAL sample trial in the Clintrial software includes the type 0 SUBJECT_LOCK panel to record the locked status of patients in a trial. You can use the InForm Architect application to create a type 0 panel by defining a mapping, and then populate the panel by using the Clintrial software.

To create a type 0 panel:

- 1 In the InForm Architect application, select the **Data Mappings** item.
- 2 Double-click the appropriate mapping definition.

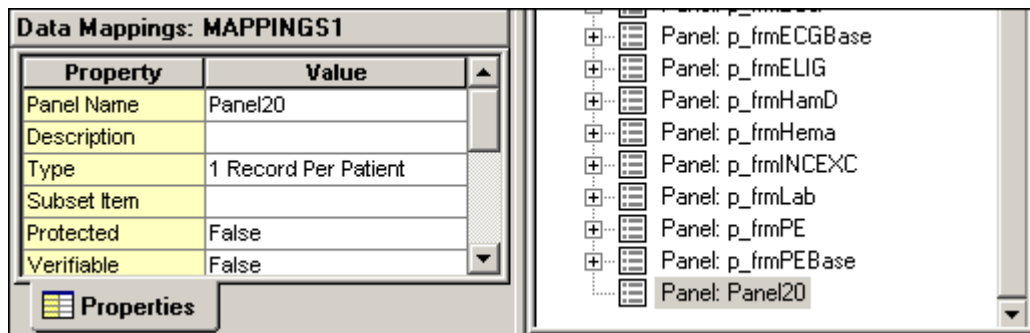
The Data Mappings window appears.



- 3 In the **Data Mappings** window, right-click the mapping node and select **New Panel**.

A new panel node appears.

- 4 Edit the panel properties as needed. Specify **Non-Patient Data** as the value of the **Type** property.



Mapping autocoded data

InForm Architect application enables you to specify mapping definitions for controls that hold verbatim text and controls that hold the coded values assigned by an external autocoding utility.

Autocode mapping workflow

To use the InForm Architect application autocode mapping feature in autocoding activities:

- 1 Design target calculated controls in forms where you want autocoded values to be inserted. For information about creating calculated controls, see *Creating a new calculated control* (on page 131).

Note: The **Calculated** property of the item in which you create a calculated control must be set to **True**.

- 2 Map these controls to source controls containing values to be autocoded.
- 3 In a production trial, export data using the **Autocode** option of InForm Data Export. For information about using InForm Data Export, see *InForm Utilities Guide*.
- 4 Use an external autocoding utility to assign codes to the exported data.
- 5 Use the **Autocode Import file** option of InForm Data Import to import the code values to the calculated controls you created in the trial database. For information about using InForm Data Import, see *InForm Utilities Guide*.

Differences between Autocode mapping and Central Coding mapping


The Autocode mapping and Central Coding mapping features of the InForm Architect application both help you to design your trial so that you can export certain data as verbatim text, process it with an external coding application, and reimport it to the trial in coded form. The following table summarizes the differences between these mapping features:

	Autocode mapping	Central Coding mapping
Coding application	Any external coding application that can be customized to accept a data file from InForm Data Export.	Central Coding
Mechanism for exporting and importing data	<ul style="list-style-type: none"> • Verbatim data—Exported with InForm Data Export to a file that can be imported to an external coding application. • Coded data—Imported with InForm Data Import from a file to the InForm application. 	Exported and imported by InForm Adapter.
Number of code targets per verbatim	One	As many as the specified dictionary allows.

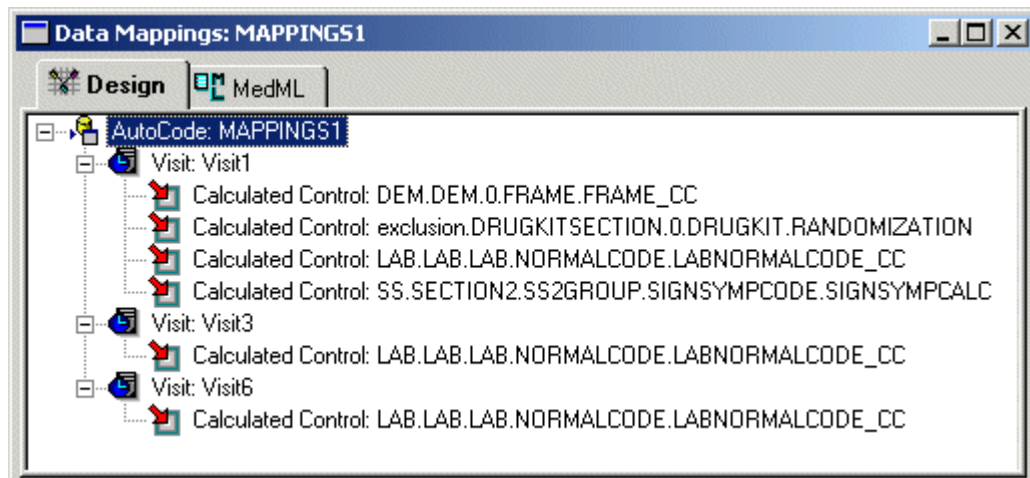
Number of context items per verbatim	None	As many as the specified dictionary allows
Type of control for storing coded data	Must be a calculated control.	Can be a text control or calculated control. Text controls provide an audit trail; defining them as read-only is recommended.
Dictionary support	None	MedDRA and WHODD

Creating an autocode mapping

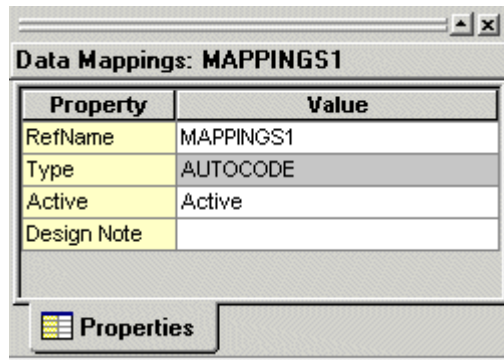
To create an autocode mapping:

- Do one of the following:
 - Select **Trial > Create Data Mappings > Autocode**.
 - In the **Trial toolbar**, click the **Data Mappings** button (). In the **Data Mappings Type** dialog box, select **Autocode** and click **OK**.
 - In the **Trial Objects** window, right-click the **Data Mappings** node and select **Create Autocode** from the pop-up menu.

The InForm Architect application generates an autocode mapping and opens the **Data Mappings** window. The autocode mapping tree consists of the visits in which calculated controls have been defined and the control paths of each calculated control.



- Select the **AutoCode** node and edit the autocode mapping property values in the **Properties** window.



The following table describes the properties of an autocode mapping definition.

Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89).
Type	AUTOCODE. READ ONLY.
Active	Active or Inactive, indicating whether the autocode mapping definition is accessible with the InForm Architect application. The default is Active. To disable an autocode mapping definition, set the value to Inactive and install the definition by saving it with the Install MedML When Saving option enabled or by using the MedML Installer utility. The autocode definition becomes inaccessible with InForm Architect application. OPTIONAL.
Design Note	Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only. OPTIONAL. Design Note text is not transferred to the target tables.

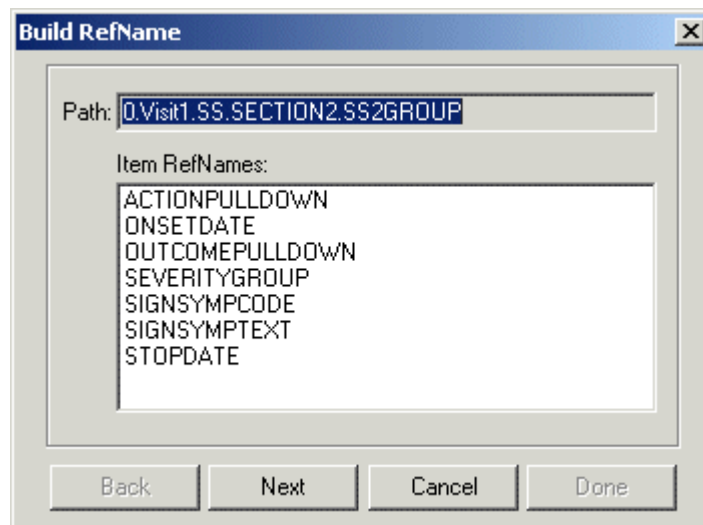
Specifying a source control

For each calculated control created to hold a code value imported from an external coding tool, you must specify a source control. In a trial, the source control holds the value exported to the coding tool. This value is used by the coding tool as the basis for determining the coded value.

To create a mapping between a source and target control:

- 1 Open the **Data Mappings** window for the autocode definition in which you want to create control mappings.
- 2 Expand the tree until the control you want to map is visible.
- 3 Right-click the control and select **Add Source** from the popup menu. The Build RefName dialog box appears.

Note: If the target calculated control is in a repeating visit, repeating form, or itemset, the source control must be in the same visit, form, or itemset. In these cases, the Path field in the Build RefName dialog box is partially filled in with the required RefNames.

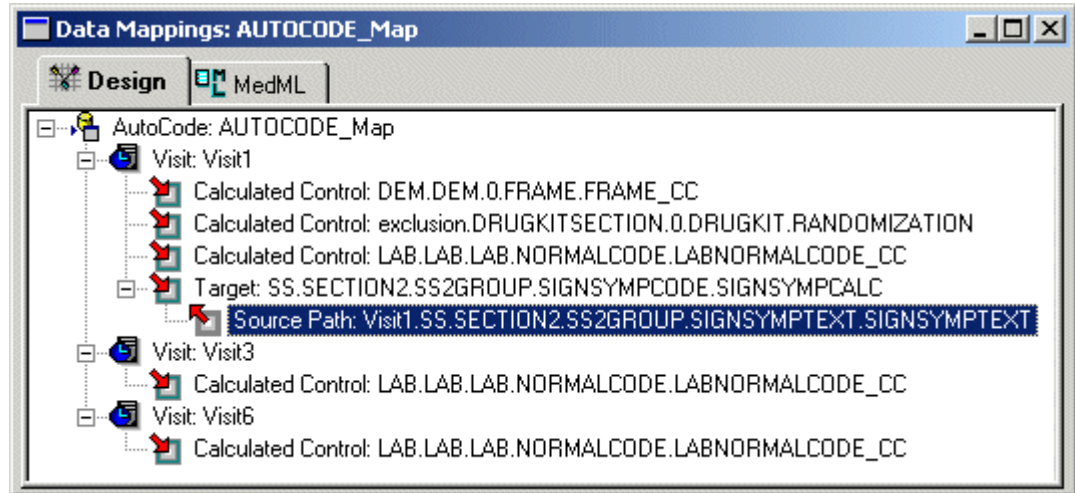


- 4 For each component of the RefName path, select the component from the list in the dialog box.
- 5 Click **Next**, or, to backtrack to a previous component, click **Back**.

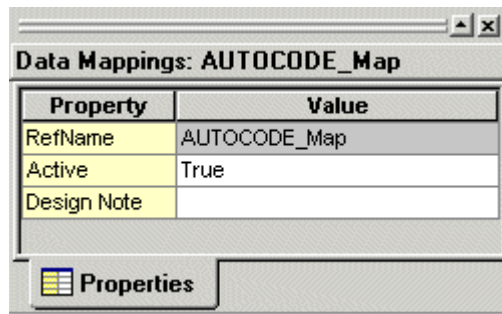
The InForm Architect application builds the path from your selections and displays it in the **Path** field at the top of the dialog box.

- 6 When the RefName path is complete, click **Done**.

The InForm Architect application inserts the path into the component definition you are creating. The following figure shows the Data Mappings window with a completed source control mapping definition.



- 7 Select the **Target node** and edit the autocode target mapping property values in the **Properties** window.



The following table describes the properties of an autocode target mapping definition.

Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89).
Active	Active or Inactive, indicating whether the target mapping definition will be in effect once it is installed and setup is complete. The default is Active. OPTIONAL.

Property	Description
Design Note	Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only. OPTIONAL.

Mapping data for coding with Central Coding

InForm Architect application enables you to specify mapping definitions for controls that hold verbatim text and controls that hold the coded values assigned by the Oracle Central Coding application.

Definitions of Central Coding mapping terms

The following table defines terms you should understand when developing Central Coding mapping definitions:

Term	Definition
Central Coding	Oracle application used to identify and apply appropriate codes to verbatim items.
code target	Coding dictionary construct that helps to classify data to be coded. Also, a corresponding form control holding data that has been coded.
context item	Coding dictionary construct that helps to classify context information for data to be coded. Also, a corresponding form control that provides additional information to enable coding of a verbatim. The WHODD dictionary uses context items to provide the following information: <ul style="list-style-type: none"> • Route of Administration—The route by which the drug was administered. • Indication—The disease or disorder for which the drug was taken.
dictionary	Standard thesaurus containing terminology used for coding. Central Coding supports the MedDRA and WHODD dictionaries.
InForm Adapter	Oracle application used to transfer verbatim and coded data between the InForm application and Central Coding.
MedDRA	Medical Dictionary for Drug Regulatory Activities —Standard thesaurus containing terminology applicable to all phases of drug development and to the health effects of devices.
verbatim	Form item holding data to be coded.
verbatim type	Specific type of verbatim defined in a coding dictionary. The following verbatim types are defined for the dictionaries supported by Central Coding: <ul style="list-style-type: none"> • Valid in MedDRA: <ul style="list-style-type: none"> ▪ AE—Adverse event ▪ DISEASE—Disease ▪ LABDATA—Lab data • Valid in WHODD: <ul style="list-style-type: none"> ▪ MEDPROD—Medical product

WHODD

World Health Organization Drug Dictionary—Standard thesaurus containing terminology about drugs.

Central Coding mapping workflow

To use the InForm Architect application coding mapping feature in coding activities with Central Coding:

- 1 For each form that contains data to be coded, design controls to hold data as entered (verbatim), controls to hold coded data (code targets), and if required by the dictionary, controls to hold additional context information (context items).

Verbatim, code target, and context item controls must be:

- In the same visit if the visit is repeating.
- On the same form if the form is repeating.
- In the same itemset if the coded data appears in an itemset.
- Different from each other.

Note: Oracle recommends that you do not place rules on items defined as code targets. This practice simplifies signature invalidation processing.

- 2 Create a mapping definition that specifies:
 - Each verbatim to be coded.
 - The dictionary used for coding.
 - Each code target to which the Central Coding software returns a coded value.
 - Each context item used to provide additional context data to the Central Coding software.
 - Each applicable verbatim type.
- 3 In a production trial, use Central Coding software to:
 - a Export the verbatim and context data from the InForm software to the Central Coding software.
 - b Code the data.
 - c Import the coded data from the Central Coding software to the InForm software.

Differences between Autocode mapping and Central Coding mapping

The Autocode mapping and Central Coding mapping features of the InForm Architect application both help you to design your trial so that you can export certain data as verbatim text, process it with an external coding application, and reimport it to the trial in coded form. The following table summarizes the differences between these mapping features:

	Autocode mapping	Central Coding mapping
Coding application	Any external coding application that can be customized to accept a data file from InForm Data Export.	Central Coding
Mechanism for exporting and importing data	<ul style="list-style-type: none"> • Verbatim data—Exported with InForm Data Export to a file that can be imported to an external coding application. • Coded data—Imported with InForm Data Import from a file to the InForm application. 	Exported and imported by InForm Adapter.
Number of code targets per verbatim	One	As many as the specified dictionary allows.
Number of context items per verbatim	None	As many as the specified dictionary allows
Type of control for storing coded data	Must be a calculated control.	Can be a text control or calculated control. Text controls provide an audit trail; defining them as read-only is recommended.
Dictionary support	None	MedDRA and WHODD

Creating coding mapping definitions

Use InForm Architect application to create the following mapping definitions:

- Coding mapping object for the trial.
- Verbatim control mapping for each verbatim.
- Code target path for each code target control.
- Context item path for each context item needed to code the verbatim.

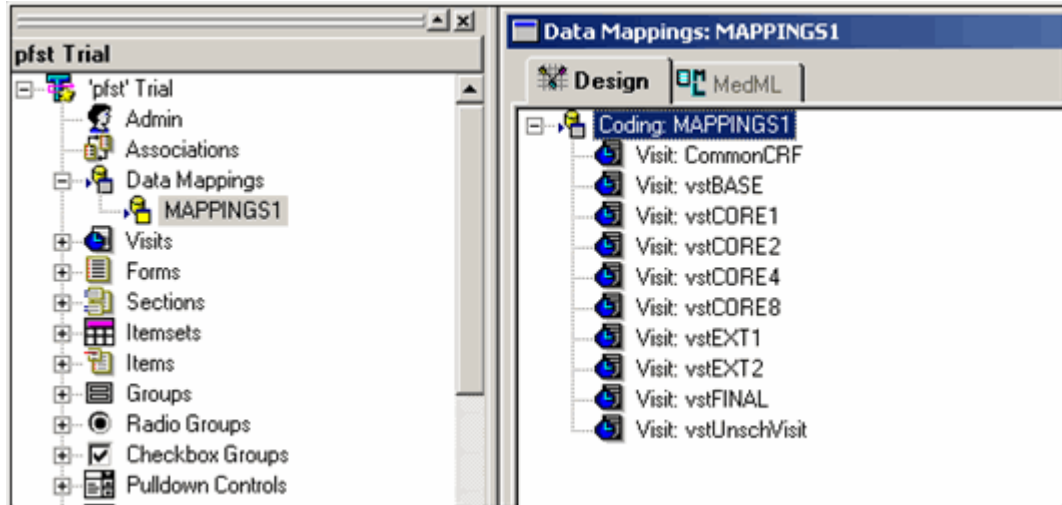
Creating a coding mapping object

The coding mapping object for a trial holds the mapping definitions for all coded items. Before creating a coding mapping object, use MedML Installer to install the XML defining the dictionary in the InForm software database.

To create a coding mapping object:

- 1 In the **Trial Objects** window, right-click **Data Mappings** and select **Create Coding**.

The **Data Mappings** window opens in the trial workspace.



- 2 Edit the Central Coding mapping property values in the **Properties** window. For more information, see *Coding mapping object properties* (on page 242).

Property	Value
RefName	CC_MAPSET1
Type	CODINGMAP
Active	Active
Design Note	

Coding mapping object properties

The following table describes the properties of a coding mapping object:

Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
Type	CODINGMAP . READ ONLY.

Property	Description
Active	<p>Active or Inactive, indicating whether the coding mapping definition is accessible with the InForm Architect application. The default is Active.</p> <p>To disable a coding mapping definition, set the value to Inactive and install the definition by saving it with the Install MedML When Saving option enabled or by using the MedML Installer utility. The coding definition becomes inaccessible with the InForm Architect application.</p> <p>OPTIONAL.</p>
Design Note	<p>Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only.</p> <p>OPTIONAL.</p>

Disabling a coding mapping object

To disable a coding mapping definition:

- 1 In the **Trial Objects** window, double-click the name of the coding mapping object.
The **Data Mappings** window opens with the mapping object name selected.
- 2 In the **Properties** window, set the value of the Active property to **Inactive**.

Property	Value
RefName	CC_MAPSET1
Type	CODINGMAP
Active	Active
Design Note	

- 3 Install the definition by saving it with the **Install MedML When Saving** option enabled or by using the MedML Installer utility.

The coding mapping definition becomes inaccessible with the InForm Architect application.

Identifying controls to be coded (verbatim)

Verbatim control mappings identify the controls on an InForm application form that hold data to be coded. You can add, change, or remove a verbatim control mapping.

To add a verbatim control mapping, use either of the following methods:

- In the Data Mappings window, specify a RefName path. For more information, see *Specifying a RefName path to add a verbatim control mapping* (on page 244).
- Drag the control from the Form window to the Data Mappings window. For more information, see *Dragging a control to add a verbatim control mapping* (on page 245).

Specifying a RefName path to add a verbatim control mapping

To add a verbatim control mapping by specifying a RefName path:

- 1 In the **Trial Objects** window, double-click the name of the coding mapping object.
The **Data Mappings** window opens with the mapping object name selected.
- 2 Right-click the node for the visit in which the verbatim occurs, and select **Add Verbatim** from the right-click menu.

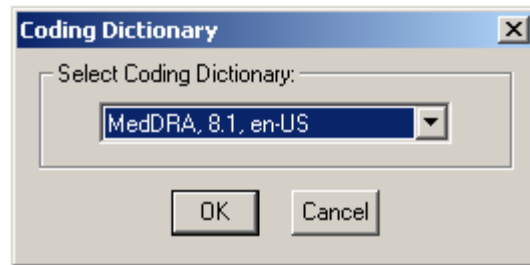
The Build RefName dialog box appears.

- 3 Use the Build RefName dialog box to specify the RefName path. For more information, see *General instructions for using the Build RefName tool* (on page 259).

The InForm Architect application enforces restrictions on verbatim control path definition. For more information, see *Verbatim control path restrictions* (on page 248).

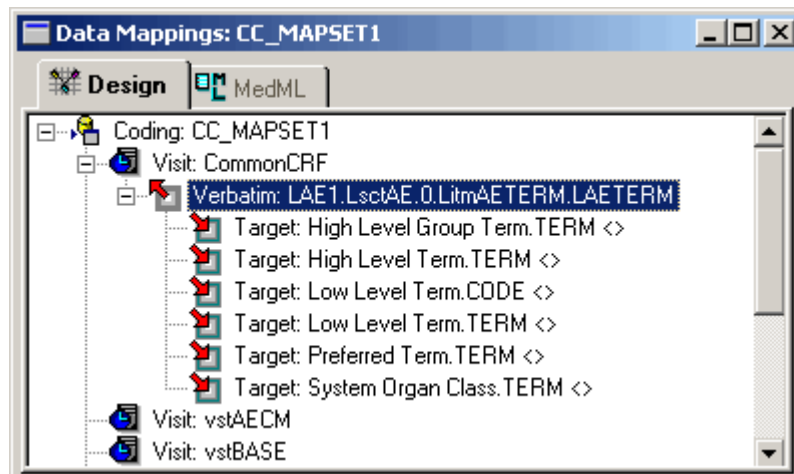
- 4 When the RefName path is complete, click **Done**.

The **Coding Dictionary** dialog box appears.



- 5 From the **Select Coding Dictionary** drop-down list, select the dictionary to use for coding the verbatim.

Click **OK**. The InForm Architect software inserts the path into the component definition you are creating. A new verbatim node appears under the visit, along with empty nodes for the code targets and context items defined for the selected dictionary.



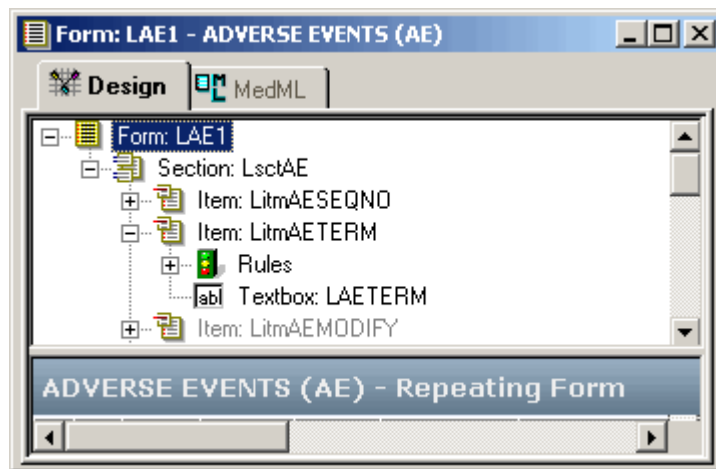
- 6 Edit the verbatim properties in the Properties window. For more information, see *Verbatim control mapping properties* (on page 246).

Property	Value
RefName	CC_MAPSET1
Control Path	0.CommonCRF.LAE1.LsctAE.
Dictionary	MedDRA, 8.1, en-US
Verbatim Type	

Dragging a control to add a verbatim control mapping

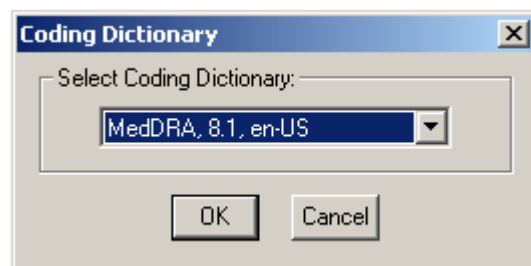
To add a verbatim control mapping by dragging a control:

- 1 In the **Trial Objects** window, double-click the name of the coding mapping object.
The **Data Mappings** window opens with the mapping object name selected.
- 2 In the **Trial Objects** window, double-click the name of the form containing the verbatim for which you want to create a mapping.
The **Form** window appears.
- 3 In the **Form** window tree, expose the verbatim control.



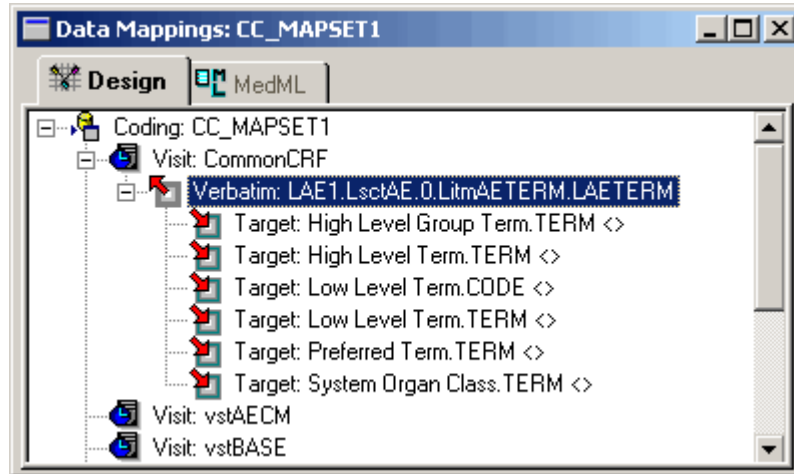
- 4 Drag the verbatim control from the **Form** window to the **Data Mappings** window and release it over the visit node where you want to create the mapping. The InForm Architect application enforces restrictions on verbatim control path definition. For more information, see *Verbatim control path restrictions* (on page 248).

The **Coding Dictionary** dialog box appears.



- From the **Select Coding Dictionary** drop-down list, select the dictionary to use for coding the verbatim.

Click **OK**. The InForm Architect software inserts the path into the component definition you are creating. A new verbatim node appears under the visit, along with empty nodes for the code targets and context items defined for the selected dictionary.



- Edit the verbatim properties in the Properties window. For more information, see *Verbatim control mapping properties* (on page 246).

Property	Value
RefName	CC_MAPSET1
Control Path	0.CommonCRF.LAE1.LsctAE.
Dictionary	MedDRA, 8.1, en-US
Verbatim Type	

Verbatim control mapping properties

The following table describes the properties of a verbatim control mapping:

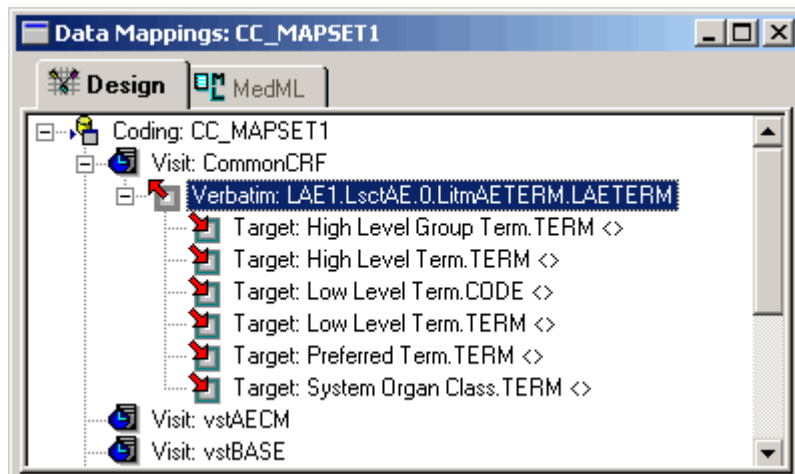
Property	Description
Control Path	RefName path of the verbatim control. READ ONLY.
Dictionary	Coding thesaurus used for coding the verbatim. If you change this property, the InForm Architect application updates the dictionary and, as necessary, the code target and context item control mappings. For more information, see <i>Verbatim dictionary updates</i> (on page 248). REQUIRED..

Property	Description
Verbatim Type	<p>Specific type of verbatim defined in a coding dictionary. The following verbatim types are defined for the dictionaries supported by Central Coding:</p> <ul style="list-style-type: none"> • Valid in MedDRA: <ul style="list-style-type: none"> ▪ AE—Adverse event ▪ DISEASE—Disease ▪ LABDATA—Lab data • Valid in WHODD: <ul style="list-style-type: none"> ▪ MEDPROD—Medical product <p>OPTIONAL.</p>

Changing a verbatim control path

To change a verbatim control path:

- 1 In the **Trial Objects** window, double-click the name of the coding mapping object.
The **Data Mappings** window opens with the mapping object name selected.
- 2 Expand the node of the visit that contains the verbatim.



- 3 Right-click the verbatim node, and select **Edit Verbatim Path** from the right-click menu.
The Build RefName dialog box appears.
- 4 Click Back until the RefName list contains the level from which to rebuild the RefName path.

Note: You can change any RefName except the visit. To create a verbatim control path starting with a different visit, you must create a new verbatim mapping in that visit.

- 5 Use the Build RefName dialog box to specify the RefName path. For more information, see *General instructions for using the Build RefName tool* (on page 259).
- 6 When the RefName path is complete, click **Done**.

- 7 InForm Architect application enforces restrictions on verbatim control path definition. For more information, see *Verbatim control path restrictions* (on page 248). If the selected control results in a valid control path, The InForm Architect application replaces the path in the component definition you are editing.

Verbatim control path restrictions

The InForm Architect application enforces the following restrictions on verbatim control paths:

- The verbatim control path must be unique within a mapping.

Note: To use the same verbatim with multiple dictionaries, create a separate mapping for each dictionary.

- Verbatim, code target, and context item control paths must be:
 - In the same visit if the visit is repeating.
 - On the same form if the form is repeating.
 - In the same itemset if the coded data appears in an itemset.
 - Different from each other.

Verbatim dictionary updates

You can change the dictionary for a verbatim by updating the Dictionary property of the verbatim. For example, if you install a new version of a dictionary, you can update the verbatim mapping definition to use the new dictionary version.

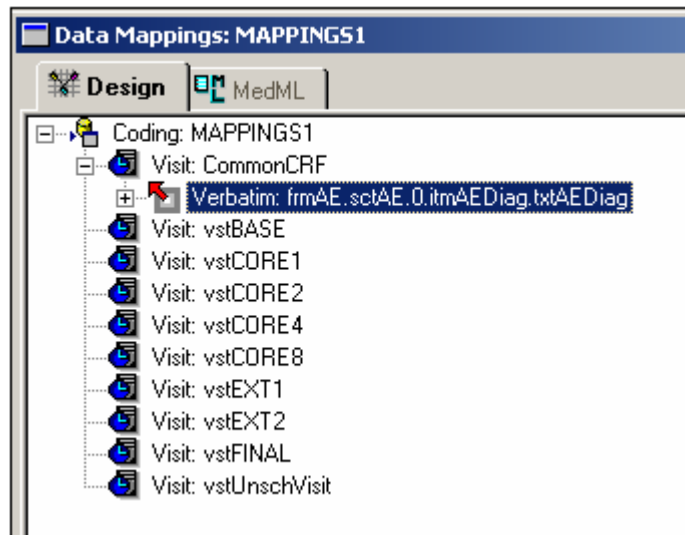
When you select a new dictionary definition from the Dictionary property dropdown list, the InForm Architect application:

- Checks for matching code targets and context items in the two dictionary definitions and preserves the existing control paths defined for code targets and context items that match.
- Adds new code target and context item nodes to the verbatim control mapping definition.
- Warns you if any code target and context item mapping definitions do not exist in the new dictionary definition.
- Removes those code target and context item mapping definitions if you confirm the Dictionary property change.

Removing a verbatim control mapping

To remove a verbatim control mapping:

- 1 In the **Trial Objects** window, double-click the name of the coding mapping object.
The **Data Mappings** window opens with the mapping object name selected.
- 2 Expand the **Visit** node to expose the verbatim control mapping you want to remove.



- 3 Right-click the node for the verbatim control mapping, and select **Remove Verbatim** from the right-click menu.

The InForm Architect application removes the verbatim control mapping definition.

Identifying code targets

Code target mappings identify the controls on an InForm application form that hold data that has been coded with Central Coding. The dictionary you choose for a verbatim determines the targets you can set. For example, when using the MedDRA, 8.1, en-US dictionary, you can set the following targets for any verbatim:

- System Organ Class.CODE
- System Organ Class.TERM
- High Level Group Term.CODE
- High Level Group Term.TERM
- High Level Term.CODE
- High Level Term.TERM
- Preferred Term.CODE
- Preferred Term.TERM
- Low Level Term.CODE
- Low Level Term.TERM

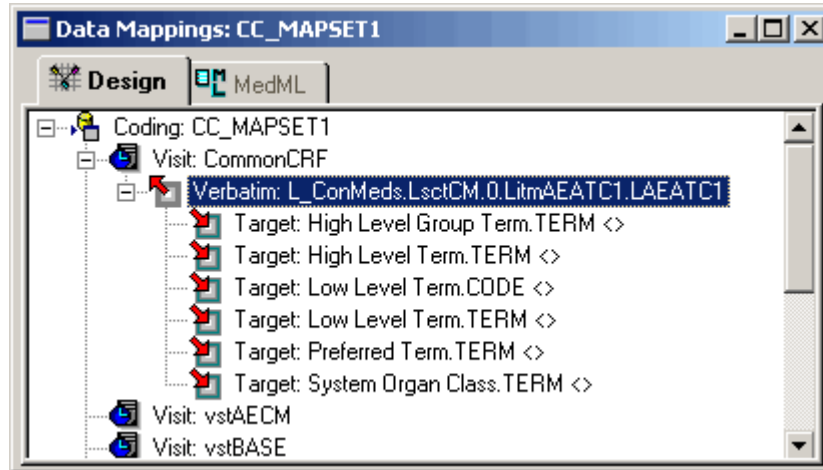
You can add, update, or remove a code target mapping. To add or update a code target control mapping, use either of the following methods:

- In the Data Mappings window, specify a RefName path. For more information, see *Specifying a RefName path to add or update a code target path* (on page 250).
- Drag the control from the Form window to the Data Mappings window. For more information, see *Dragging a control to add or update a code target path* (on page 251).

Specifying a RefName path to add or update a code target path

To add or update a code target control path by specifying a RefName path:

- 1 In the **Trial Objects** window, double-click the name of the coding mapping object.
The **Data Mappings** window opens with the mapping object name selected.
- 2 In the **Data Mappings** tree, expand the Verbatim node.

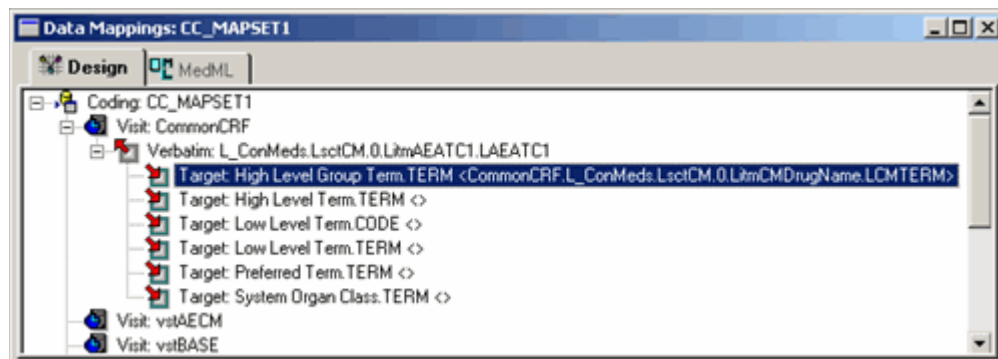


- 3 Right-click the **Target** to which you want to map a form control, and select **Set Target Path** from the right-click menu.

The Build RefName dialog box appears.

- 4 Use the Build RefName dialog box to specify the RefName path. For more information, see *General instructions for using the Build RefName tool* (on page 259).
- 5 When the RefName path is complete, click **Done**.

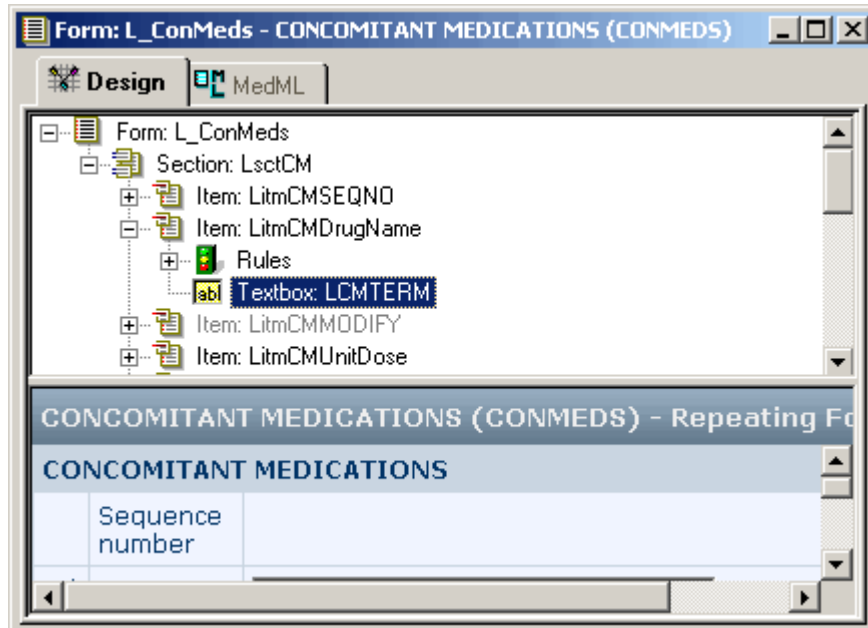
The InForm Architect application enforces restrictions on code target control path definition. For more information, see *Code target control path restrictions* (on page 252). If the selected control results in a valid code target control path, the InForm Architect application inserts or replaces the code target control path in the **Target** node.



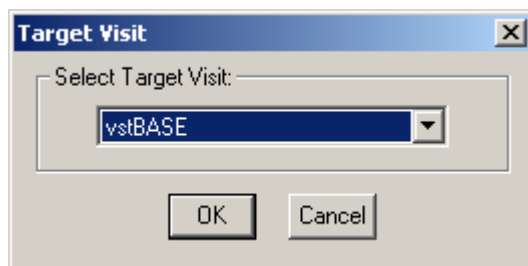
Dragging a control to add or update a code target path

To add or update a code target path by dragging a control:

- 1 In the **Trial Objects** window, double-click the name of the coding mapping object.
The **Data Mappings** window opens with the mapping object name selected.
- 2 In the **Trial Objects** window, double-click the name of the form containing the code target control for which you want to create or update a mapping.
The **Form** window appears.
- 3 In the **Form** window tree, expose the code target control.

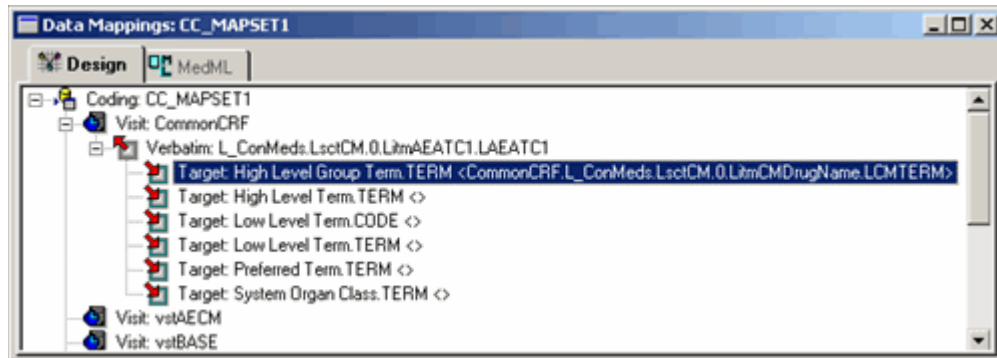


- 4 Drag the code target control from the **Form** window to the **Data Mappings** window and release it over the **Code Target** node where you want to create or update the mapping.
The **Target Visit** dialog box appears.



- 5 Select the visit to use in the code target path and click **OK**.

The InForm Architect application enforces restrictions on code target control path definition. For more information, see *Code target control path restrictions* (on page 252). If the selected control results in a valid code target control path, the InForm Architect application inserts or replaces the code target control path in the **Target** node.



Code target mapping properties

The following table describes the properties of a code target mapping:

Property	Description
Name	Name of the code target in the dictionary. READ ONLY.
Control Path	RefName path of the control mapped to the dictionary code target. READ ONLY.

Code target control path restrictions

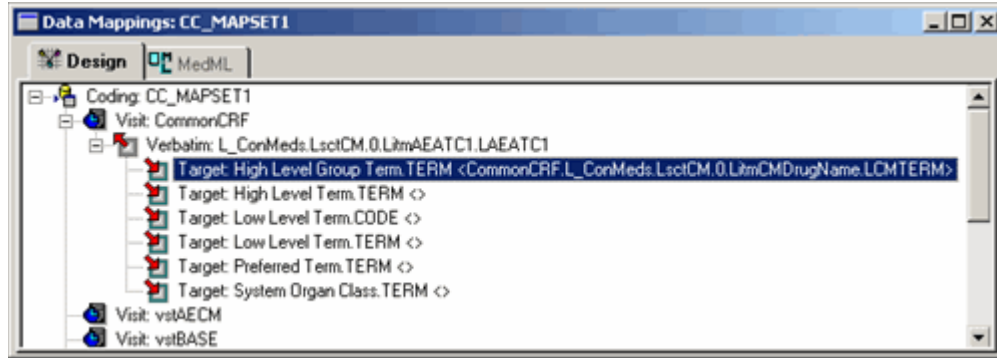
The InForm Architect application enforces the following restrictions on code target control paths:

- The code target control path must be unique within a mapping.
- The code target control path must be different from the verbatim and any context item control paths within a mapping.
- The code target control path must point to the top-level text box control or calculated control.
- Verbatim and code target or context item controls must be:
 - In the same visit if the visit is repeating.
 - On the same form if the form is repeating.
 - In the same itemset if the coded data appears in an itemset.
 - Different from each other.

Removing a code target path

To remove a code target path:

- 1 In the **Trial Objects** window, double-click the name of the coding mapping object.
The **Data Mappings** window opens with the mapping object name selected.
- 2 Expand the **Visit** and **Verbatim** nodes containing the code target path you want to remove.



- 3 Right-click the **Target** node, and select **Clear Target Path** from the right-click menu.

The InForm Architect application removes the RefName path from the definition of the code target.

Identifying context items

A context item provides additional information to enable coding of a verbatim. The WHODD dictionary uses context items to provide the following information:

- **Route of Administration**—The route by which the drug was administered.
- **Indication**—The disease or disorder for which the drug was taken.

The mapping for a context item identifies the RefName path of the control whose data provides the context information. You can add, update, or remove a context item mapping. To add or update a context item control mapping, use either of the following methods:

- In the Data Mappings window, specify a RefName path. For more information, see *Specifying a RefName path to add or update a context item path* (on page 253).
- Drag the control from the Form window to the Data Mappings window. For more information, see *Dragging a control to add or update a context item path* (on page 255).

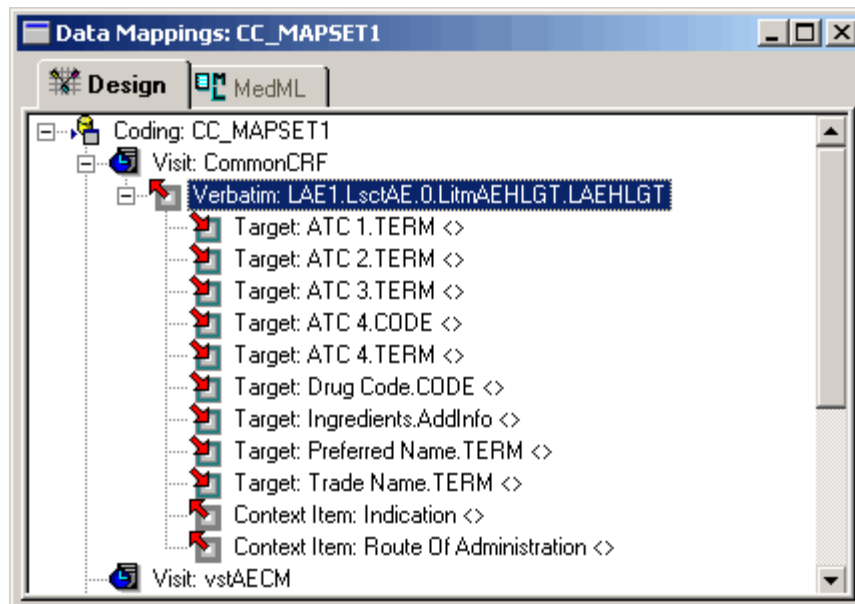
Specifying a RefName path to add or update a context item path

To add or update a context item control path by specifying a RefName path:

- 1 In the **Trial Objects** window, double-click the name of the coding mapping object.

The **Data Mappings** window opens with the mapping object name selected.

- In the **Data Mappings** tree, expand the Verbatim node.

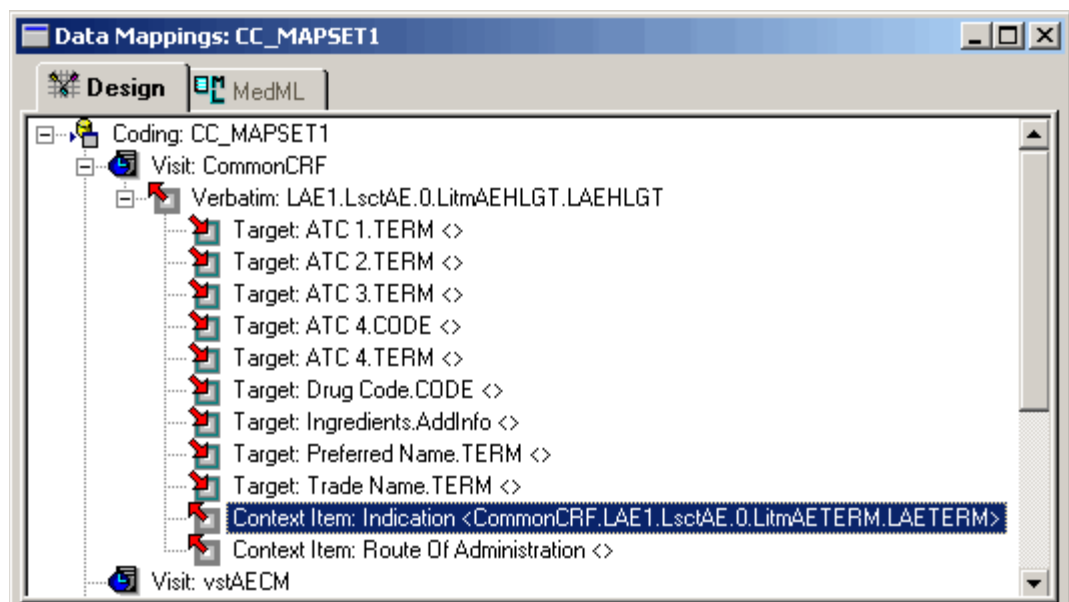


- Right-click the **Context Item** to which you want to map a form control, and select **Set Context Item Path** from the right-click menu.

The Build RefName dialog box appears.

- Use the Build RefName dialog box to specify the RefName path. For more information, see *General instructions for using the Build RefName tool* (on page 259).
- When the RefName path is complete, click **Done**.

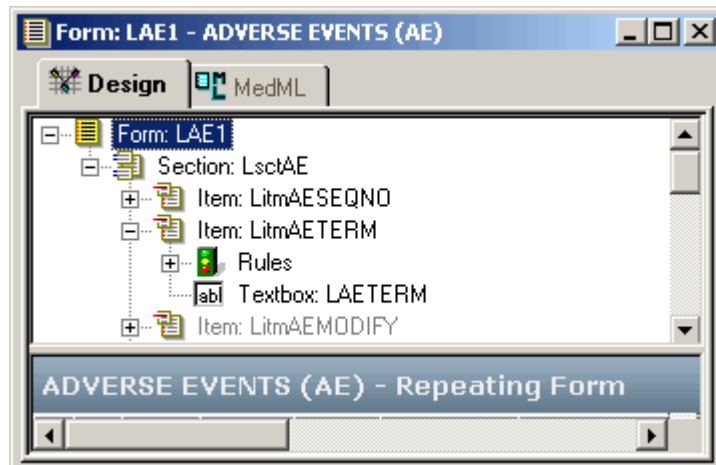
The InForm Architect application enforces restrictions on context item control path definition. For more information, see *Context item control path restrictions* (on page 256). If the selected control results in a valid context item control path, the InForm Architect application inserts or replaces the context item control path in the **Context Item** node.



Dragging a control to add or update a context item path

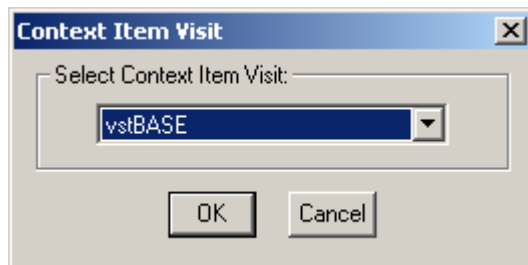
To add or update a context item control path by dragging a control:

- 1 In the **Trial Objects** window, double-click the name of the coding mapping object.
The **Data Mappings** window opens with the mapping object name selected.
- 2 In the **Trial Objects** window, double-click the name of the form containing the context item control for which you want to create or update a mapping.
- 3 The **Form** window appears.



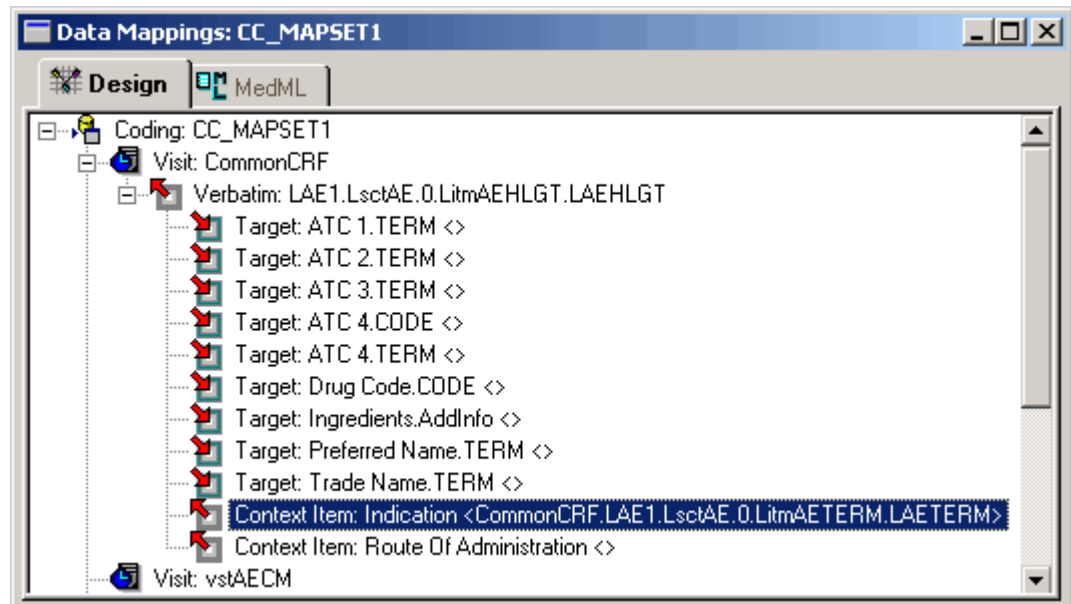
- 4 In the **Form** window tree, expose the context item control.
- 5 Drag the context item control from the **Form** window to the **Data Mappings** window and release it over the **Context Item** node where you want to create or update the mapping.

The **Context Item Visit** dialog box appears.



- 6 Select the visit to use in the context item control path and click **OK**.

The InForm Architect application enforces restrictions on context item control path definition. For more information, see *Context item control path restrictions* (on page 256). If the selected control results in a valid context item control path, the InForm Architect application inserts or replaces the context item control path in the **Context Item** node.



Context item mapping properties

The following table describes the properties of a context item mapping:

Property	Description
Name	Name of the context item in the dictionary. READ ONLY.
Control Path	RefName path of the control mapped to the dictionary context item. READ ONLY.

Context item control path restrictions

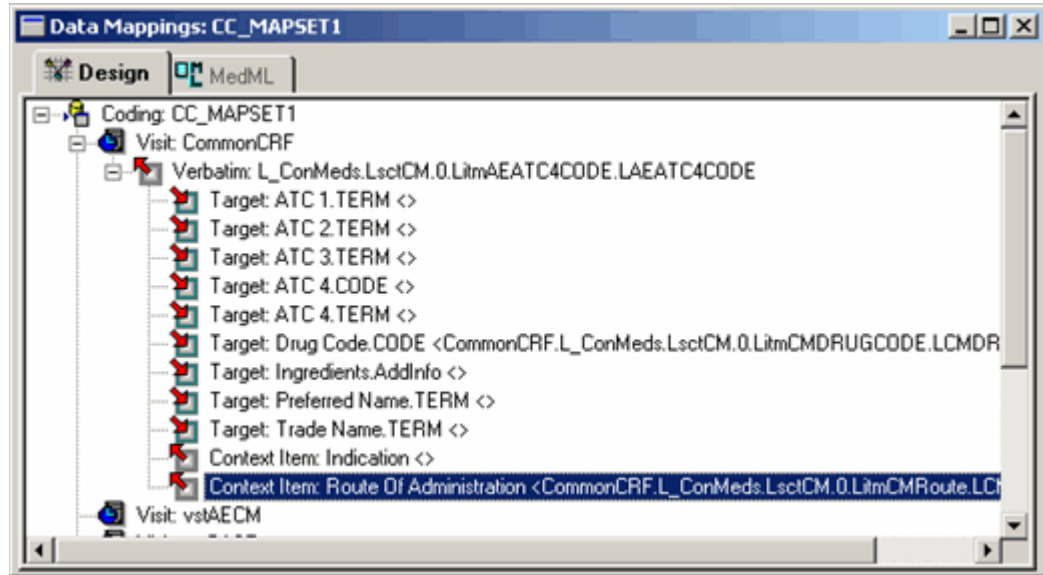
The InForm Architect application enforces the following restrictions on context item control paths:

- The context item control path must be different from any code target control paths within a mapping.
- Verbatim, code target, and context item controls must be:
 - In the same visit if the visit is repeating.
 - On the same form if the form is repeating.
 - In the same itemset if the coded data appears in an itemset.
 - Different from each other.

Removing a context item path

To remove a context item path:

- 1 In the **Trial Objects** window, double-click the name of the coding mapping object.
The **Data Mappings** window opens with the mapping object name selected.
- 2 Expand the **Visit** and **Verbatim** nodes containing the context item path you want to remove.



- 3 Right-click the **Context Item** node, and select **Clear Context Item Path** from the right-click menu.
The InForm Architect application removes the RefName path from the definition of the context item.

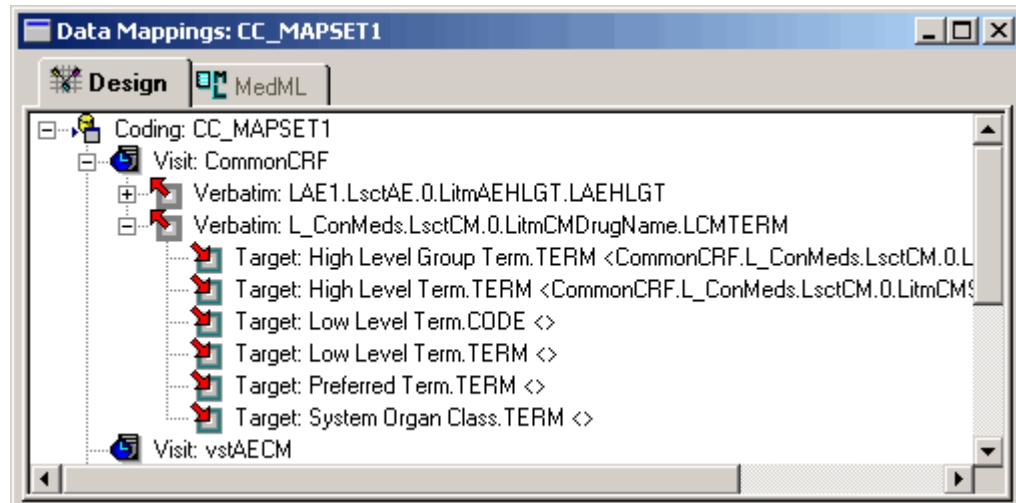
Viewing coding mappings

You can browse the coding mapping definitions you have created by opening the Data Mappings window.

To view a coding mapping definition:

- 1 In the **Trial Objects** window, double-click the name of the coding mapping object.
The **Data Mappings** window opens with the mapping object name selected.
- 2 Expand **Visit** and **Verbatim** nodes as needed.

Note: You can use the commands on the Data mappings window right-click menu to expand or collapse all nodes or visit nodes as needed. For more information, see *Expanding and collapsing the view of mappings* (on page 258).



Expanding and collapsing the view of mappings

You can expand or collapse all nodes in the Data Mapping window at once. Right-click in the window and select one of the right-click menu options.

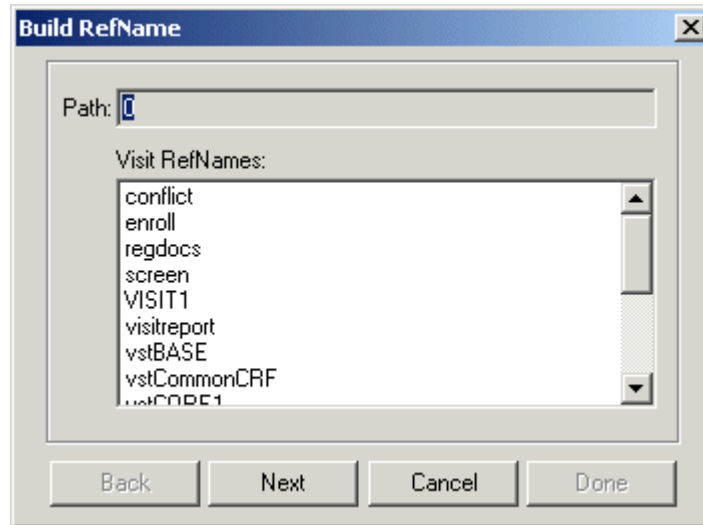
The following table describes the options for expanding the collapsing the view of mappings:

Command	Resulting view in Data Mappings window
Collapse All	All nodes collapse so that only the mapping object node is visible
Expand All	All nodes expand so that visits, verbatims, targets, and context items are visible
Collapse Visits	Visit nodes collapse so that only visits are visible
Expand Visits	Visit nodes expand so that visits and verbatims are visible

General instructions for using the Build RefName tool

- 1 Display the **Build RefName** dialog box.

Note that the InForm Architect application partially fills in the **Path** field if it can obtain information from the location from which you display the dialog box. The label for the list of RefNames and the RefNames in the list change according to the context. For example, after you select a visit RefName, the label and list prompt you to select an itemset or item name.



- 2 For each component of the RefName path, select the component from the list in the dialog box.
- 3 Click **Next**, or, to backtrack to a previous component, click **Back**.

The InForm Architect application builds the path from your selections and displays it in the **Path** field at the top of the dialog box.

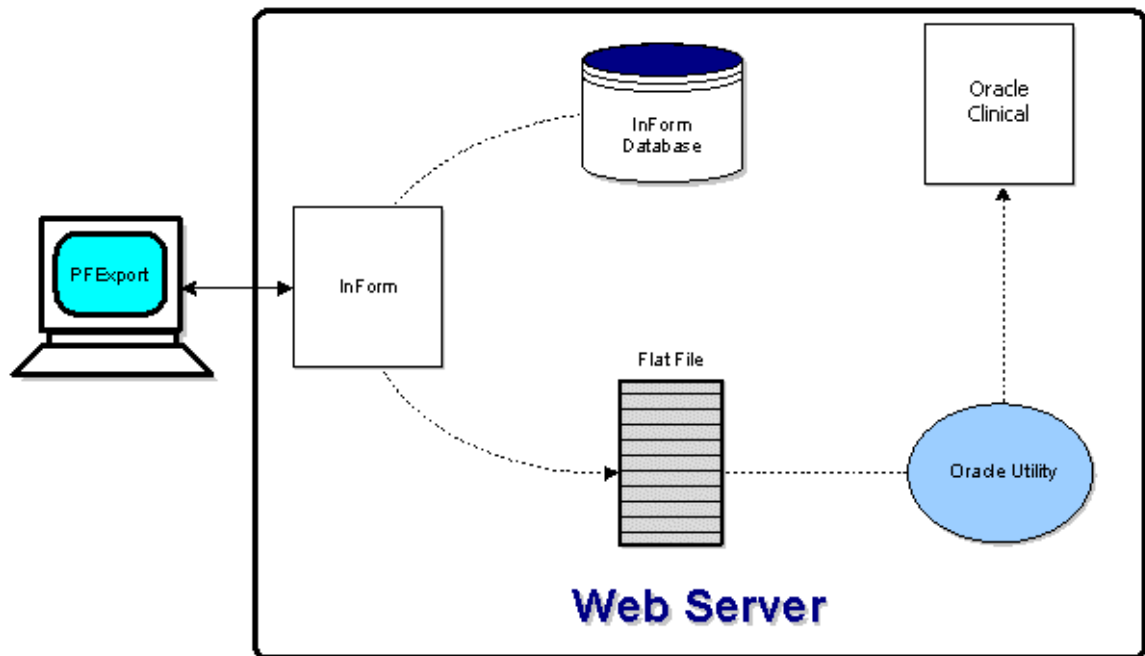
- 4 When the RefName path is complete, click **Done**.

The InForm Architect application inserts the path into the component definition you are creating.

Mapping data to Oracle Clinical

The Oracle Clinical application enables users to collect and report on clinical trial data. Using Oracle Clinical reporting features, sponsors can review data as needed to spot results that might require protocol adjustments or warrant early statistical analysis.

The InForm application provides a *bulk data transfer* tool that enables data collected in a trial to be exported in a format that can be uploaded to the Oracle Clinical application. This tool, the Oracle Clinical Export component of the InForm Data Export data transfer utility, transfers data to a flat file that can be imported into the Oracle Clinical application. An export file can include incremental transactions from a specified date and time or can be cumulative. For information on using the Oracle Clinical option of the InForm Export utility, see *InForm Utilities Guide*.



Setting up Oracle Clinical processing

To set up data transfer to the Oracle Clinical application, do the following:

- 1 Define Oracle Clinical data mappings through the InForm Architect application.
- 2 Produce a fixed-length or character-separated flat file of the mapped data using the InForm Data Export tool.
- 3 Import the data mappings into an Oracle Clinical application.

Note: This section describes how to define and maintain Oracle Clinical mapping definitions in the InForm Architect application. For information on exporting data, see *InForm Utilities Guide*.

About Oracle Clinical mappings

To specify how to transfer data from an InForm application trial database to an Oracle Clinical application for interim analysis, you create a set of Oracle Clinical mapping definitions.

The definitions of Oracle Clinical mappings specify:


- Where each mapped data point comes from in the InForm application source trial.
- The CPE Reference to which each data point belongs. A CPE Reference is a reference to a Clinically Planned Event, which corresponds to a visit in an InForm application trial.
- How the data is grouped in the Oracle Clinical application.
- Optional, supplemental text about the design of the components of the Oracle Clinical mapping definition.

The sections that follow describe the steps for defining Oracle Clinical mappings. Note that you must manually define the mappings; no auto-generate option is available. To define Oracle Clinical mappings:

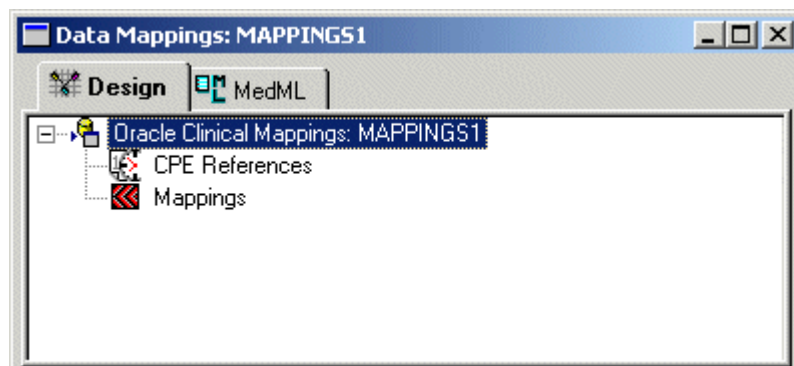
- 1 Create an Oracle Clinical mapping definition.
- 2 Define CPE References.
- 3 Define control mappings.

Creating an Oracle Clinical mapping definition

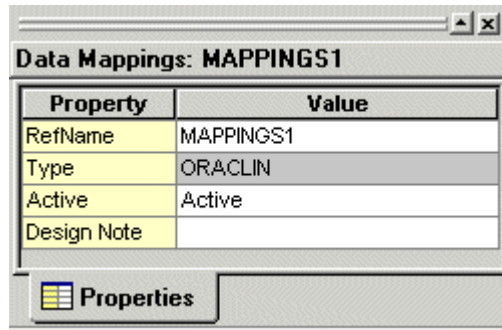
To create an Oracle Clinical mapping:

- 1 Do one of the following:
 - Select **Trial > Create Data Mappings > Oracle Clinical**.
 - In the **Trial** toolbar, click the **Data Mappings** button. . In the **Data Mappings Type** dialog box, select **Oracle Clinical** and click **OK**.
 - In the **Trial Objects** window, right-click the **Data Mappings** node and select **Create Oracle Clinical** from the pop-up menu.

The InForm Architect application generates an Oracle Clinical mapping and opens the **Data Mappings** window. The Oracle Clinical mapping tree consists of a CPE Reference node and a Mappings node for individual control mappings.



- Select the **Oracle Clinical Mappings** node and edit the mapping property values in the **Properties** window.



The following table describes the properties of an Oracle Clinical mapping definition:

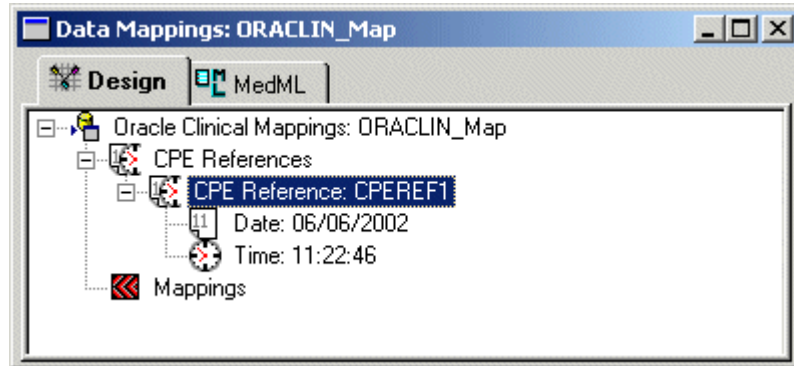
Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
Type	ORACLIN . READ ONLY.
Active	Active or Inactive, indicating whether the Oracle Clinical mapping definition is accessible with the InForm Architect application. The default is Active. To disable an Oracle Clinical mapping definition, set the value to Inactive and install the definition by saving it with the Install MedML When Saving option enabled or by using the MedML Installer utility. The Oracle Clinical definition becomes inaccessible with the InForm Architect application. OPTIONAL.
Design Note	Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only. OPTIONAL. Design Note text is not transferred to the target tables.

Defining a CPE Reference

To define a CPE Reference:

- 1 Open the **Data Mappings** window for the Oracle Clinical definition you want to update.
- 2 Right-click the **CPE Reference node**, and select **New CPE Reference**.

The InForm Architect application adds a CPE Reference to the CPE References node.



- 3 Set the date and, optionally, the time of the **CPE Reference** (on page 263), as described in *Defining the date of a CPE Reference* (on page 263) and *Defining the time of a CPE Reference* (on page 265).
- 4 Select the **CPE Reference node** and edit the mapping property values in the **Properties** window, as described in *Defining the properties of a CPE Reference* (on page 267).

Defining the date of a CPE Reference

A CPE Reference mapping includes the specification of the date and, optionally, the time of the Clinically Planned Event. You can define the CPE date by specifying a control that contains the visit date or by entering an explicit date.

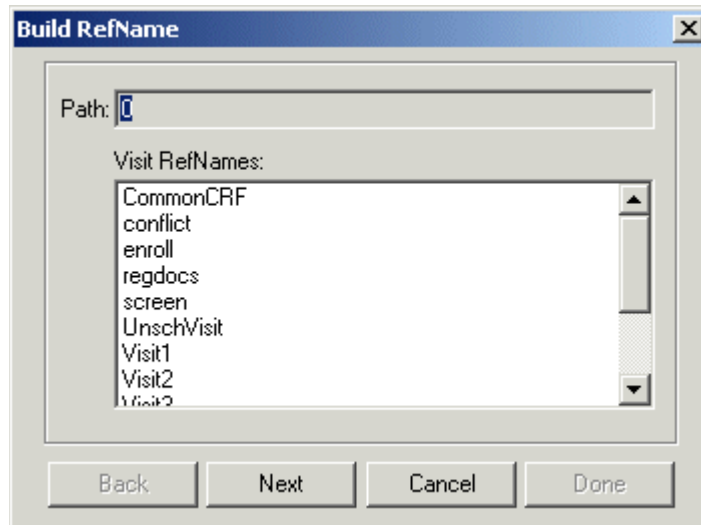
Note: If you choose to define the CPE Reference date by specifying a datetime control that holds the visit date, the control must be defined with all date subcomponents (Month, Day, and Year) required.

To define a CPE Reference date, right-click the Date node in the Data Mappings window. Select a command in the popup menu based on whether how you want to define the date:

- **To specify a datetime control in the InForm application trial:**

- 1 Select **InForm Path**.

The Build RefName dialog box appears.

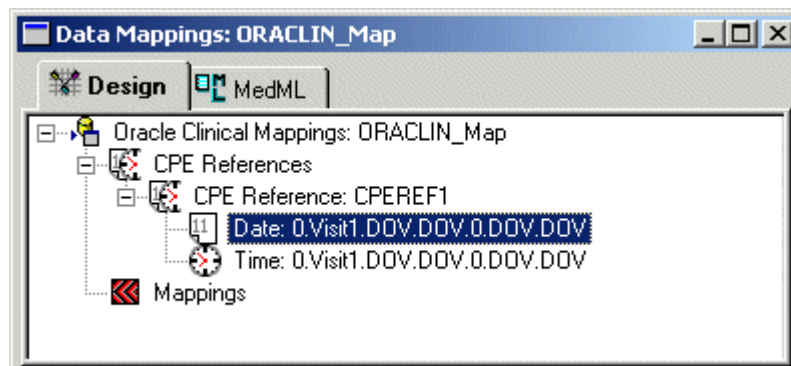


- 2 For each component of the RefName path, select the component from the list in the dialog box.
- 3 Click **Next**, or, to backtrack to a previous component, click **Back**.

The InForm Architect application builds the path from your selections and displays it in the **Path** field at the top of the dialog box.

- 4 When the RefName path is complete, click **Done**.

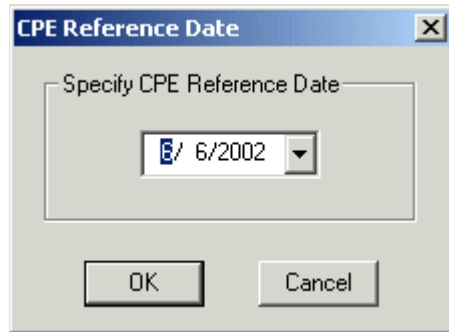
The InForm Architect application inserts the path into the component definition you are creating.



- **To enter an explicit date:**

- 1 Select **Specify**.

The **CPE Reference Date** dialog box appears.



- 2 Click the pulldown list and use the popup calendar to select the date you want.
- 3 Click **OK**.

The InForm Architect application updates the **Data Mappings** window with the date you selected.

Defining the time of a CPE Reference

A CPE Reference mapping includes the specification of the date and, optionally, the time of the Clinically Planned Event. You can define the CPE time by specifying a control that contains the visit time or by entering an explicit time.

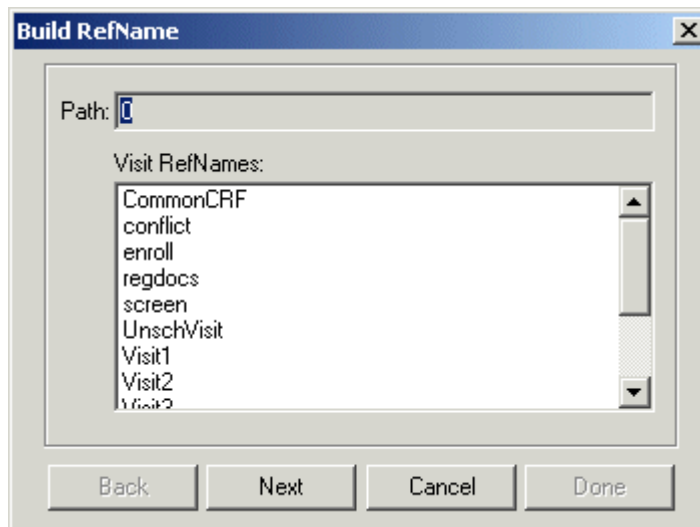
Note: If you choose to define the CPE Reference time by specifying a datetime control that holds the visit time, the control must be defined with the Hour and Minute subcomponents required.

To define a CPE Reference time, right-click the **Time** node. Select a command in the popup menu based on whether how you want to define the date:

- **To specify a datetime control in the InForm application trial:**

- 1 Select **InForm Path**.

The Build RefName dialog box appears.

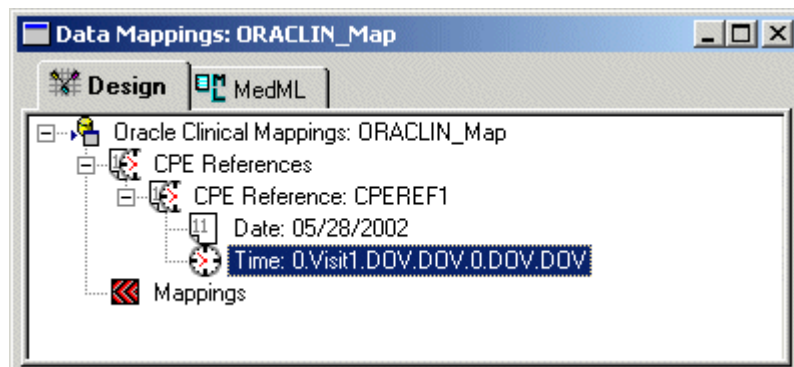


- 2 For each component of the RefName path, select the component from the list in the dialog box.
- 3 Click **Next**, or, to backtrack to a previous component, click **Back**.

The InForm Architect application builds the path from your selections and displays it in the **Path** field at the top of the dialog box.

- 4 When the RefName path is complete, click **Done**.

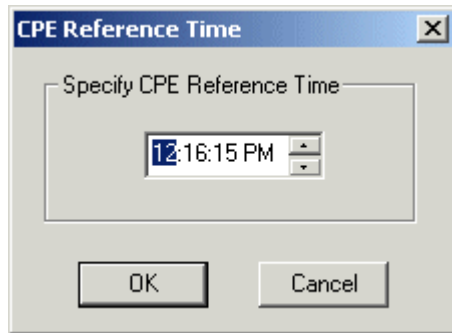
The InForm Architect application inserts the path into the component definition you are creating.



- **To enter an explicit time:**

- 1 Select **Specify**.

The **CPE Reference Time** dialog box opens.



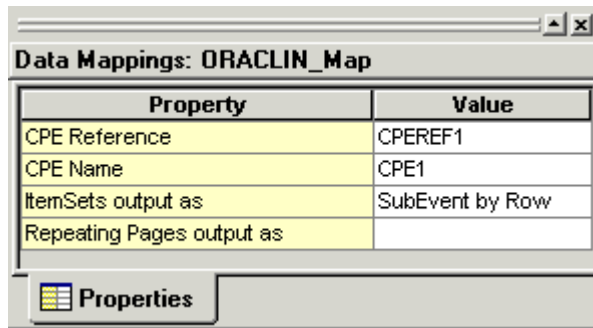
- 2 Click the hours, minutes, seconds, or AM/PM portion of the time and enter or scroll to the time you want.
- 3 Click **OK**.

The InForm Architect application updates the **Data Mappings** window with the time you selected.

- To specify that the CPE Reference mapping is defined by date but not time, select **None**.

Defining the properties of a CPE Reference

To define CPE Reference properties, select the CPE Reference note and edit the values in the Properties window.



The following table describes the properties of a CPE Reference mapping definition.

Property	Description
CPE Reference	Name of the subevent associated with the date and time values provided in the Date and Time nodes. Subevents are defined by their date and time attributes. In visits that are unscheduled and repeating, each instance of the visit corresponds to a subevent within a CPE in Oracle Clinical. REQUIRED..
CPE Name	Name of the CPE for the CPE Reference.

Property	Description
ItemSets output as	<p>Method to use for handling itemsets in exports.</p> <ul style="list-style-type: none"> • Repeat Sequence — Handle itemsets as repeating sequences. • Subevent By Date — Handle itemsets as subevents, with each different itemset date resulting in a new subevent. • Subevent By Row — Handle itemsets as subevents, with each itemset row resulting in a new subevent, regardless of whether the date is different from the previous row. <p>Note: Exporting as subevents incremented by row is not recommended. Items that are not in an itemset and share the same CPE name with items that are in an itemset run the risk of having their subevent numbers being incremented by those items in itemsets.</p> <p>REQUIRED..</p>
Repeating Pages output as	<p>Method to use for handling instances of repeating forms in exports. RepeatSequence is the default:</p> <ul style="list-style-type: none"> • Repeat Sequence — Handle repeating form instances as repeating sequences. • Subevent By Date — Handle repeating form instances as subevents, with each different form instance date resulting in a new subevent. • Subevent By Row — Handle repeating form instances as subevents, with each form instance resulting in a new subevent, regardless of whether the date is different from the previous instance. <p>OPTIONAL.</p>

Defining a control mapping

A control mapping definition consists of:

- The RefName path of the control.
- The CPE Reference identifying the subevent with which the control is associated.
- Additional properties used to characterize and group the data in Oracle Clinical.

If you are creating mappings for a whole trial, the simplest way is to let the InForm Architect application display RefName paths for all controls at once. Then you can then edit their properties individually to complete their definitions.

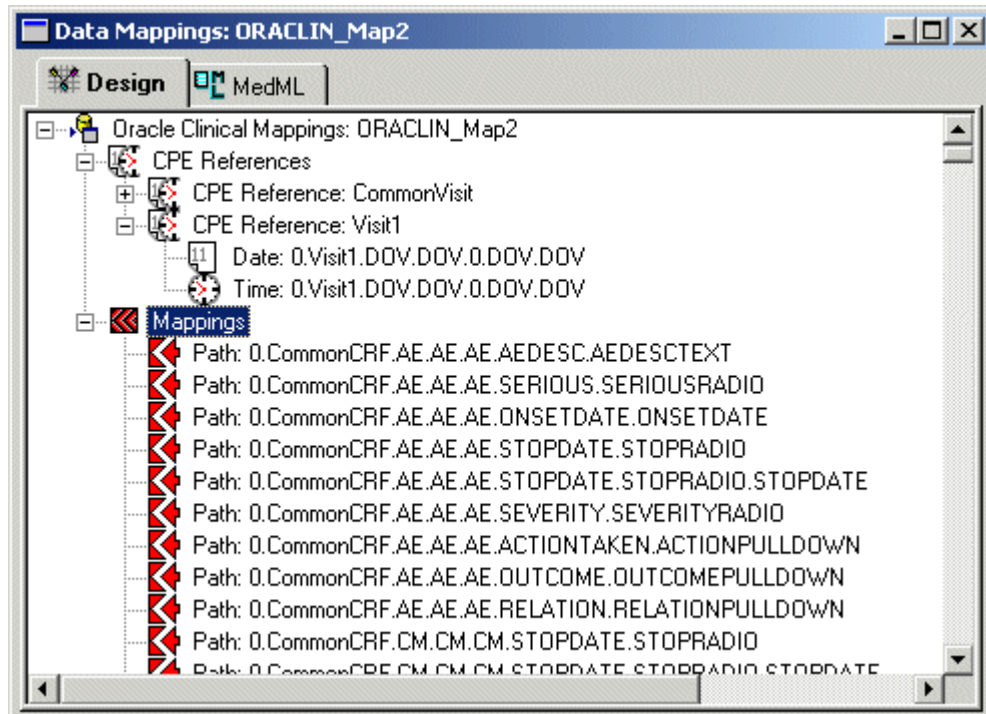
You can also create definitions of individual control mappings, specifying their RefNames and their properties.

Displaying RefName paths for all controls

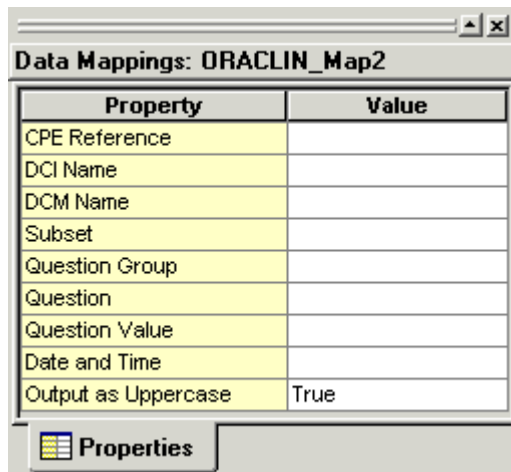
To define control path mappings by displaying the RefName paths for all controls in the trial:

- 1 Open the **Data Mappings** window for the Oracle Clinical mapping in which you want to create control path mappings.
- 2 Right-click the **Mappings** node and select **Show All Controls**.

The InForm Architect application updates the **Data Mappings** window with a list of RefName paths for all controls in the trial.



- 3 Select the **RefName path** of a control, and edit its mapping properties in the **Properties** window.

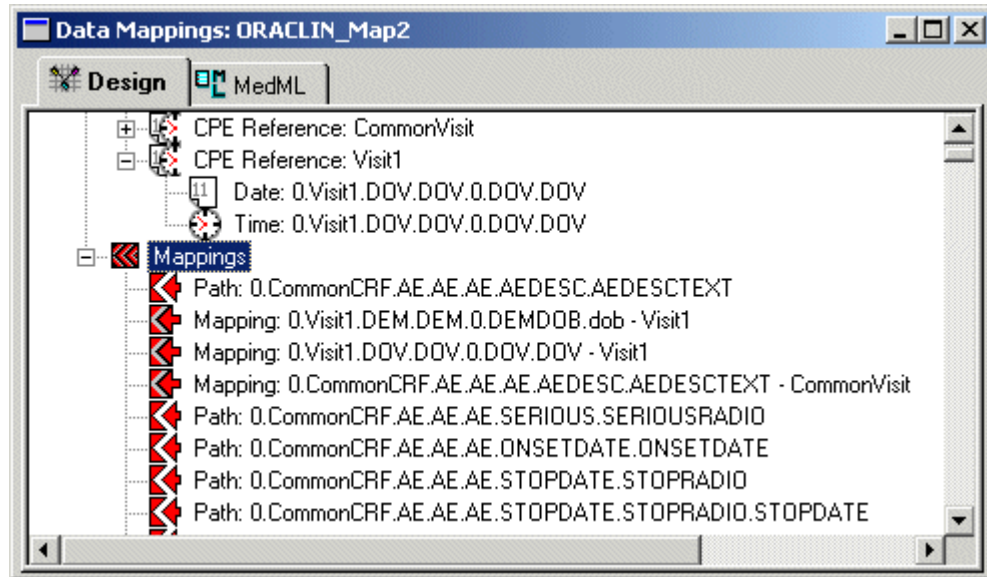


The following table describes the properties of a control mapping definition.

Property	Description
CPE Reference	Name of the subevent of the Clinically Planned Event (CPE) with which the data in the control is associated. This name matches the CPE Reference property specified for the CPE Reference mapping definition. REQUIRED..
DCI Name	Data Collection Instrument (DCI) name, a customer-defined grouping of Data Collection Modules. REQUIRED..
DCM Name	Data Collection Module (DCM) name, a customer-defined grouping of data items. REQUIRED..
Subset	Name of a customer-defined grouping by data type. REQUIRED..
Question Group	Name of a customer-defined grouping by question type. REQUIRED..
Question	Customer-defined standard variable name for the question (for example, lab test performed). REQUIRED..
Question Value	Answer to the question, or test result. REQUIRED..
Date and Time	If the field is a datetime field, the pertinent part of the data: Date and Time, Date part only, or Time part only. OPTIONAL
Output as Uppercase	True or False, indicating whether the data is translated to uppercase. True is the default. REQUIRED..
Repeat Sequence Number	Number incremented for each result. You can also use a repeat sequence number in conjunction with the Occurrence number used to indicate dependant or related data. A common repeat sequence number indicates a relationship between values. OPTIONAL.
Occurrence	Number of test performed; used to associate repeating data, for example repeated labs. A change in occurrence number indicates a retest value (for example, initial value=0, retest value =1). OPTIONAL; default value is zero.

Property	Description
SubEvent Number	Number 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. Use a unique subevent number for each such assignment. OPTIONAL; default value is zero.

The following figure shows the Data Mappings window after some controls have been mapped to their CPE References. Note that the CPE Reference is appended to the RefName path of mapped controls.



Defining an individual control mapping

To create a single control mapping, you can do the following:

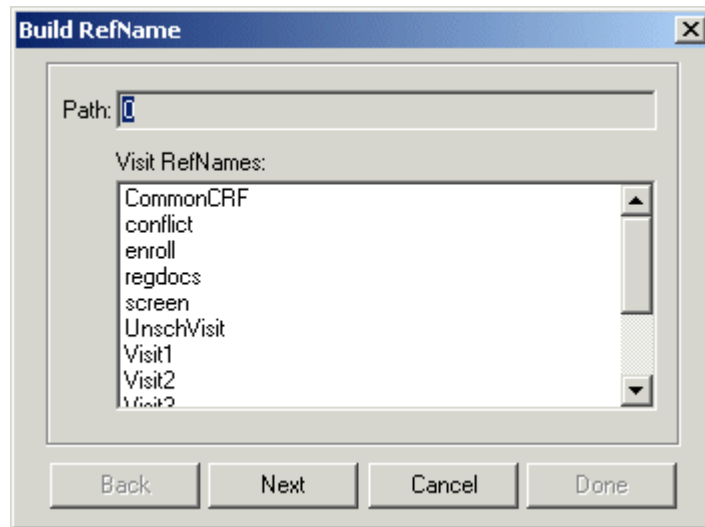
- **In the Data Mappings window** (on page 271), use the New Mapping pop-up menu command for the item in which you want to create the control path.
- **In the Form window** (on page 272), use the Add Data Mapping pop-up menu command for the control you want to map, or you can drag the control from the Form window onto the Data Mappings window.

Using the Data Mappings window to define an individual control mapping

To create a control mapping definition:

- 1 In the **Data Mappings** window, right-click the **Mappings** node.
- 2 From the pop-up menu, choose **New Mapping**.

The Build RefName dialog box appears.

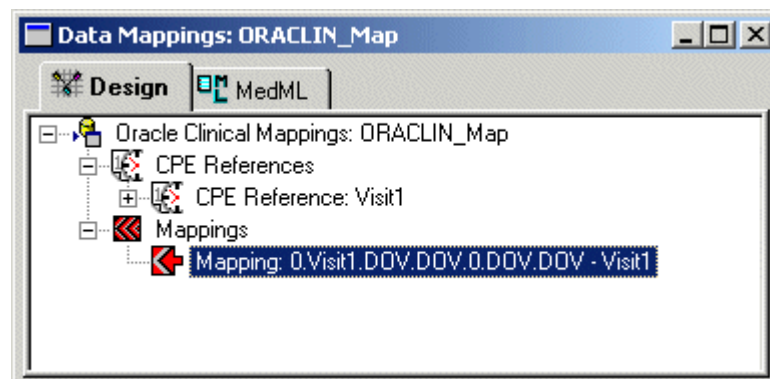


- 3 For each component of the RefName path, select the component from the list in the dialog box.
- 4 Click **Next**, or, to backtrack to a previous component, click **Back**.

The InForm Architect application builds the path from your selections and displays it in the **Path** field at the top of the dialog box.

- 5 When the RefName path is complete, click **Done**.

The InForm Architect application inserts the path into the component definition you are creating.



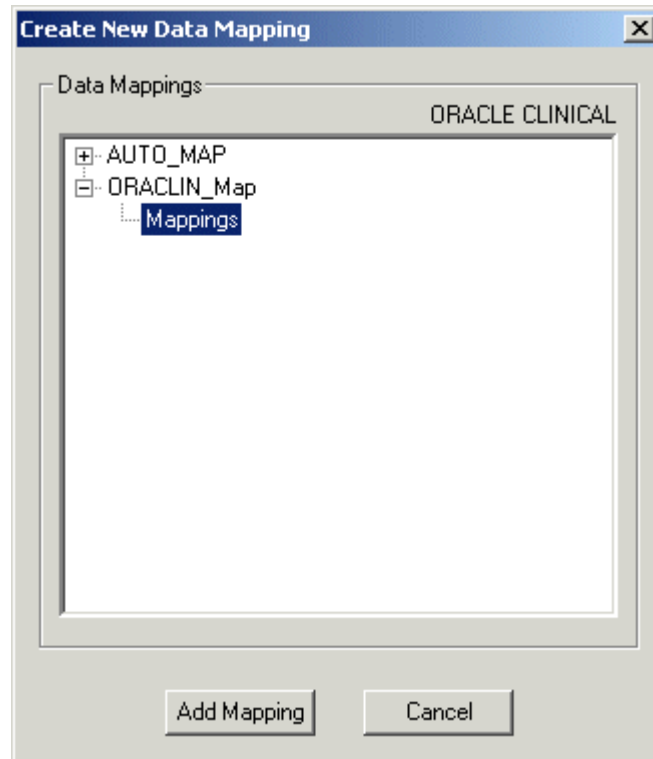
- 6 Select the control path, and edit its mapping properties in the **Properties** window as described in *Displaying RefName paths for all controls* (on page 269).

Using the Form window to define an individual control mapping

To create a control mapping definition:

- 1 Open the **Form** window containing the control you want to map and expand the tree so that the control is visible.
- 2 Right-click the control and choose **Add Data Mapping** from the pop-up menu.

The **Create New Data Mapping** dialog box appears.



- 3 Expand the **Oracle Clinical mapping definition** node where you want to create the control mapping and select the **Mappings** node.
- 4 Click **Add Mapping**.

The InForm Architect application generates a definition for the control path.

- 5 Select the control path and edit its mapping properties in the **Properties** window as described in *Displaying RefName paths for all controls* (on page 269).

You can also quickly generate a definition by dragging the control from the Form window onto the Data Mappings window.

Note: The InForm Architect application only allows you to drag and drop controls for which you can legitimately create mappings. You cannot drag and drop a simple control or a group control.

To drag a control from the Form window to the Data Mappings window:

- 1 Open the **Form** window containing the control you want to map and expand the tree so that the control is visible.
- 2 Open the **Data Mappings** window and expand the definition so that the table in which you want to create the column definition is visible.
- 3 Drag the control from the **Form** window and drop it onto the table in the **Data Mappings** window.
- 4 Select the control path and edit its mapping properties in the **Properties** window as described in *Displaying RefName paths for all controls* (on page 269).

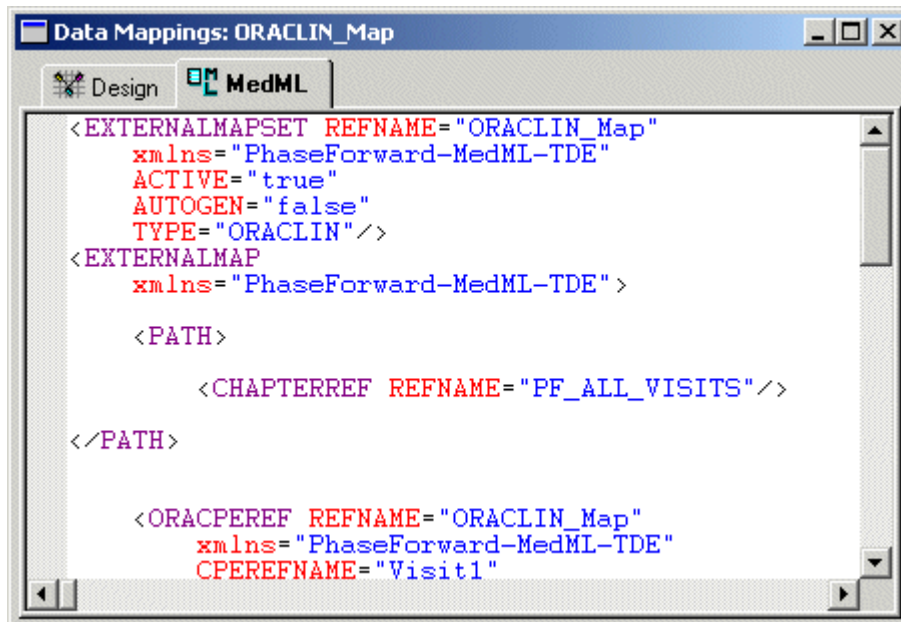
Maintaining mappings

Through the InForm Architect application, you can perform the following maintenance activities on data mapping definitions:

- View the MedML.
- Update mapping definitions.
- Delete mapping definitions.
- Navigate to forms with mapped controls.

Viewing mapping definition MedML

When you open a Data Mappings window, and as you generate and update mapping definition components, the InForm Architect application generates the MedML for the data mapping definition. To view this, click the **MedML** tab in the **Data Mappings** window. You cannot edit the MedML in the InForm Architect application.



Updating a data mapping definition

To update any component of a data mapping definition:

- 1 Open the **Data Mappings** window for the data mapping definition you want to edit by double-clicking its icon in the **Trial Objects** window.
- 2 Expand the mapping definition nodes as necessary to expose the component you want to update.
- 3 Select the component.
- 4 In the **Properties** window, edit the properties you want to change.

Deleting a data mapping definition component

To delete any component of a data mapping definition:

- 1 Open the **Data Mappings** window for the data mapping definition you want to edit by double-clicking its icon in the **Trial Objects** window.
- 2 Expand the mapping definition nodes as necessary to expose the component you want to update.
- 3 Right-click the component.
- 4 From the right-click menu, select the applicable **Delete** command.

When you delete a component that has child components, all of the children are deleted along with the parent. For example, when you delete a table definition in a CDD mapping definition, all of its column and control definitions are deleted as well.

Note: To undo a deletion, select **Undo** from the **Edit** menu or click the **Undo** button on the **Edit** toolbar. If you delete a component with multiple children, each **Undo** operation restores one child component. To restore all child components, perform the **Undo** operation as many times as there are child components.

Removing a mapping definition

You can disable a mapping definition by using the **Active** property of the definition. When a mapping definition is disabled, it is invisible in the InForm Architect application.

To disable a mapping definition:

- 1 In the **Trial Objects** window, double-click the icon for the mapping definition to open the definition in the **Data Mappings** window.
- 2 In the **Properties** window, change the value of the **Active** property to **Inactive**.
- 3 Save the mapping definition and install it by doing one of the following:
 - Save the definition with the **Install MedML When Saving** option enabled.
 - Install the definition with the MedML Installer utility.

The mapping definition disappears from the **Trial Objects** window. If the **Data Mappings** window is open at the time you install the new definition, it closes.

Viewing Data Mappings for a form control

A control on a form can be mapped to multiple locations in different types of data mapping definitions. The InForm Architect application provides an easy way to review all of the mappings for a specific control:

- 1 Open the **Form** window containing the control by double-clicking its icon in the **Trial Objects** window.
- 2 Expand the nodes of the **Form** window as necessary to expose the control you want to review.

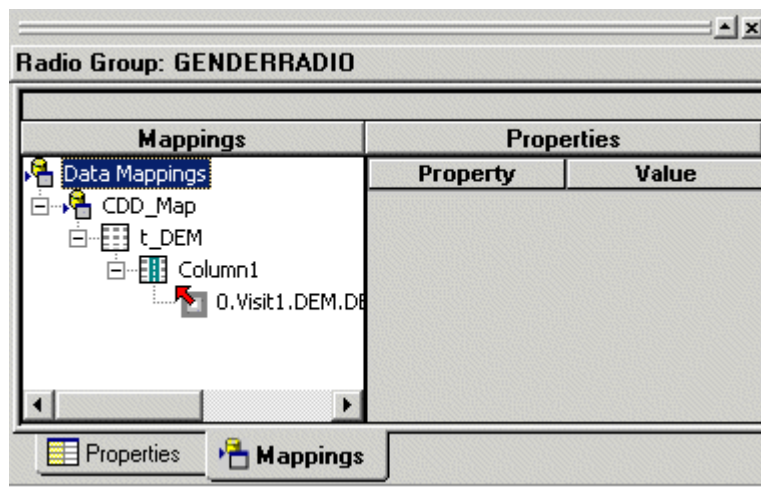
- 3 Click the control.

A Mappings tab appears in the Properties window.

- 4 Click the **Mappings** tab.

The Properties window splits. The left pane lists all Data Mappings to which the selected control is mapped. When you expand the Data Mappings nodes, you can see the mapping components in which the control appears. For names that are too long for the visible pane, you can change the size of the window or display a tool tip with the complete name by placing the cursor above the name. To open the Data Mappings window, right-click the data mapping node and select Data Mappings Editor from the right-click menu. The Data Mappings window opens with the selected node highlighted.

When you click a data mapping definition node in the left pane, the right pane displays the data mapping component's properties. This information is read-only.



Viewing the form to which a control is mapped

In addition to letting you view the Data Mappings window for a control on a form, the InForm Architect application enables you to go the other way and view the source form for a control path in a data mapping.

To open the Form window for a specific control path:

- 1 Open the **Data Mappings** window containing the control path.
- 2 Expand the tree so that the control path is visible.
- 3 Double-click the control path.

The Form window opens for the form containing the source control.

Viewing data mappings in annotated trial design

The annotated trial design display mode in the InForm Architect application enables you to display and print trial specifications, including data mapping definitions. To view data mappings in an annotated trial design:

- 1 In the **Trial Objects** window of the InForm Architect application, double-click the name of the form for which you want to review mappings.

The form appears in the Form window.

- 2 Click **View > Settings > Display Annotated View**.

The annotated view of the form appears in the Form window.

PFST45 : DEMOGRAPHICS (DEM)		
Demographics		
1.	Gender	[1] <input type="radio"/> Male [2] <input type="radio"/> Female
2.	Date of Birth	Req ▾ / Req ▾ / Req ▾ (1932-1987)
3.	Race	[4] <input type="radio"/> White, not of Hispanic origin [2] <input type="radio"/> Black, not of Hispanic origin [3] <input type="radio"/> Hispanic [1] <input type="radio"/> Asian or Pacific Islander [5] <input type="radio"/> American Indian or Alaskan Native [6] <input type="radio"/> Other or Unknown
4.	Screening Date	Req ▾ / Req ▾ / Req ▾ (2004-2010)
5.	Height	xxxx. <input type="radio"/> cm <input type="radio"/> in
Family History		
6.	Marital Status	Pulldown List 1 ▾

- 3 To display the names of target columns to which each item is mapped, click **View > Settings > Display Data Mappings in Annotated View**.

The annotated view changes to include the names of target columns next to each data item or control for which a mapping exists.

PFST45 : DEMOGRAPHICS (DEM)	
Demographics	
1. Gender	(MAPPINGS2:p_frmDem.sc_itmDe_rdcDemGendr) [1] <input type="radio"/> Male [2] <input type="radio"/> Female
2. Date of Birth	<input type="text" value="Req"/> / <input type="text" value="Req"/> / <input type="text" value="Req"/> (1932-1987) (MAPPINGS2:p_frmDem.sc_itmDe_dtmDemDO_DY) (MAPPINGS2:p_frmDem.sc_itmDe_dtmDemDO_MO) (MAPPINGS2:p_frmDem.sc_itmDe_dtmDemDO_YR)
3. Race	(MAPPINGS2:p_frmDem.sc_mitmR_mrdcRace) [4] <input type="radio"/> White, not of Hispanic origin [2] <input type="radio"/> Black, not of Hispanic origin [3] <input type="radio"/> Hispanic [1] <input type="radio"/> Asian or Pacific Islander [5] <input type="radio"/> American Indian or Alaskan Native [6] <input type="radio"/> Other or Unknown
4. Screening Date	<input type="text" value="Req"/> / <input type="text" value="Req"/> / <input type="text" value="Req"/> (2004-2010) (MAPPINGS2:p_frmDem.sc_itmDe_mGlobalD_DY) (MAPPINGS2:p_frmDem.sc_itmDe_mGlobalD_MO) (MAPPINGS2:p_frmDem.sc_itmDe_mGlobalD_YR)

- To display mapping reference tables, click **View > Settings > Display Data Mappings Tables in Annotated View**.

The annotated view changes to include a table. For a CDD mapping definition, the table shows item names, database formats, and design notes. For a Clintrial mapping definition, the table shows panel and item mapping properties.

ClinTrial: CT_MAP		Panel: p_frmDem		Panel Type: 1 (1 Record Per Patient)									
Description: DEMOGRAPHICS		Detail: False		Detail CItem:									
Protected: False		Verifiable: False											
Master Panel:		Master Item:											
SAS Name:		Lock Status: 0 (Modifiable)											
Item Name	Database Format	Date Part	SAS Name	Derived	Required	Repeat	Code List	Check List	Range	Key Order	Copy with Panel	Lock Status	Design Note
sc_itmDe_rdcDemGendr	NUMBER (1)											M	Gender
sc_itmDe_dtmDemDO_YR	NUMBER (4)	Year										M	Date of Birth
sc_itmDe_dtmDemDO_MO	NUMBER (4)	Month										M	Date of Birth
sc_itmDe_dtmDemDO_DY	NUMBER (4)	Day										M	Date of Birth
sc_mitmR_mrdcRace	NUMBER (1)											M	Race
sc_itmDe_mGlobalD_YR	NUMBER (4)	Year										M	Screening Date
sc_itmDe_mGlobalD_MO	NUMBER (4)	Month										M	Screening Date

CHAPTER 7

Defining rules

In this chapter

Overview	280
About rules.....	281
Creating rules	293
Rule objects and methods.....	319
Testing rules.....	323
Debugging rule scripts	327

Overview

The InForm Architect application enables you to create and use rules to:

- Perform validation checks on entered or updated data.
- Calculate data values.
- Perform unit conversions.
- Determine a patient's randomization stratification and assign a randomization number.

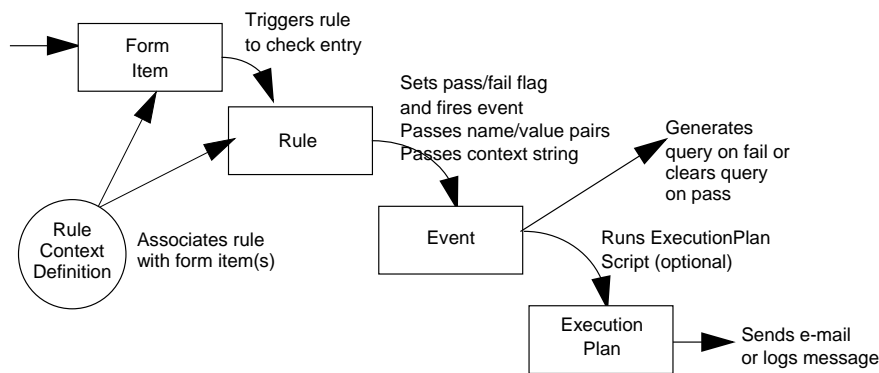
About rules

A *rule* is a piece of code that runs in the context of one or more specific items on a form. You code rule scripts with Visual Basic Script (VBScript) that refer to the script files, and associate the rule definitions with the items against which they run. This section provides an overview of rule and event processing, describes the types of rules the InForm Architect application supports, and discusses the concept of rule contexts.

Rules and events

A rule runs when an item on a form is submitted for the first time or is updated. Each rule can be associated with an *event*. Events fire on completion of the processing of a rule script. If the logic in the script determines that a rule has failed, the event generates a query against the item that originated the rule processing. If the rule passes, and a query exists against the item, the event clears the query. The definition of the event includes the text of the query.

Along with the definition of a query, an event definition can contain a reference to an execution plan that can send email (or, through an email gateway, send a fax or activate a beeper) or send a message to the Windows event log. The following figure illustrates the chain of rule processing that is set off by the submission of a form item.



Types of rules

The InForm Architect application enables you to create the following types of rules:

- **Conversions** (on page 282)
- **Calculations** (on page 283)
- **Form rules** (on page 283)
- **Randomization rules** (on page 283)

Conversions

Conversions are rules that convert values from one unit to another. You can define unit form components with reference to other units, so that data can be displayed online as one unit and stored in the database, after conversion, as the other. For example, you can design a form so that a CRC can choose whether to enter a weight in kilograms or pounds and specify that the weight is always stored in the database in kilograms. The unit in which data is stored in the database after conversion is called the base unit, and the data value, after conversion to the base unit, is called the normalized value.

To perform conversions between the entered and base unit values, you create conversion rules and associate them with unit definitions.

Calculations

Calculations enable you to set the value of a data item by performing a calculation that uses the value entered in one or more related items. The data item to which the calculation is applied is a control in an item on a form, most commonly a calculated control. When the values in all dependent items are submitted or updated, the InForm application runs the calculation rule and places the result in the calculated control.

In the sample trial installed with the InForm application, the Frame Size item on the Demographics form in Visit 1 is a calculated item. This item receives the resulting value of a rule that calculates frame size by dividing the value entered in the Height item by the value entered in the Wrist Circumference item, applying a constant, and comparing the result with specified ranges.

Form rules

Form rules can perform a wide range of edit checks on items entered or updated on a form. For example:

- You can check for conformity with a specified range of values.
- You can make sure that two items are in an appropriate relationship with each other; for example, that the date of the second visit is between ten days and two weeks later than the date of the first visit.
- You can ensure that all parts of a multiple-part control are filled in; for example, you make sure that users select a unit value on an item where units are specified, or that all parts of a required date and time control are specified.
- You can pass a message to an execution plan that logs an event in the Microsoft Windows event log or sends email.

Form rules run when both of the following are true:

- All items that the rule needs for processing have data values.
- The value of at least one of the items on which the rule depends is entered or updated.

Each form rule is associated with an *event*. Depending on the result to which the rule evaluates and on the definition of the event, the following can occur:

- The event can generate a query or clear an open query.
- The event can run one or more *execution plans* that:
 - Send e-mail
 - Send an entry to the Windows log file

For information on events, see *Defining Events and Execution Plans* (on page 331).

Randomization rules

A randomization rule is a specialized form of calculation that returns the number of a drug kit to assign to a randomized patient. In a randomization rule you test the criteria, if any, for patient stratification, set the properties of the Randomization object, and call a function that generates the drug kit number. This rule is attached to a special drug kit calculated control in a randomization item and section. The rule runs when a user clicks the Randomization button in the form window. For examples of randomization rules, see *Randomization rule scripts* (on page 320).

Note: A randomization rule can be attached to only one control, that is, it can have only one context. For more information, see *About rule contexts* (on page 284).

Browser and server rules

The InForm application supports the rules that run on the browser, before a user has submitted data, and rules that run on the server upon submission of new or updated data. Browser rules are primarily for data type checks; for example, to ensure that a float item is entered with the proper number of digits after the decimal place. Server rules can be used for many types of edit checks; for example, to check data ranges, or to calculate or check the value of one item based on the value of another.

About rule contexts

Rules are said to run in the *context* of the item they are evaluating or calculating and all items on whose values the rule script depends. A rule context answers the questions:

- What item does the rule run against?
- What values are substituted in the rule when it runs against this item?
- What other values does the rule need to run, and what items are those values found in?

By separating the definition of a rule from the specification of the item it runs against, contexts give you the flexibility to associate the same rule with many different items.

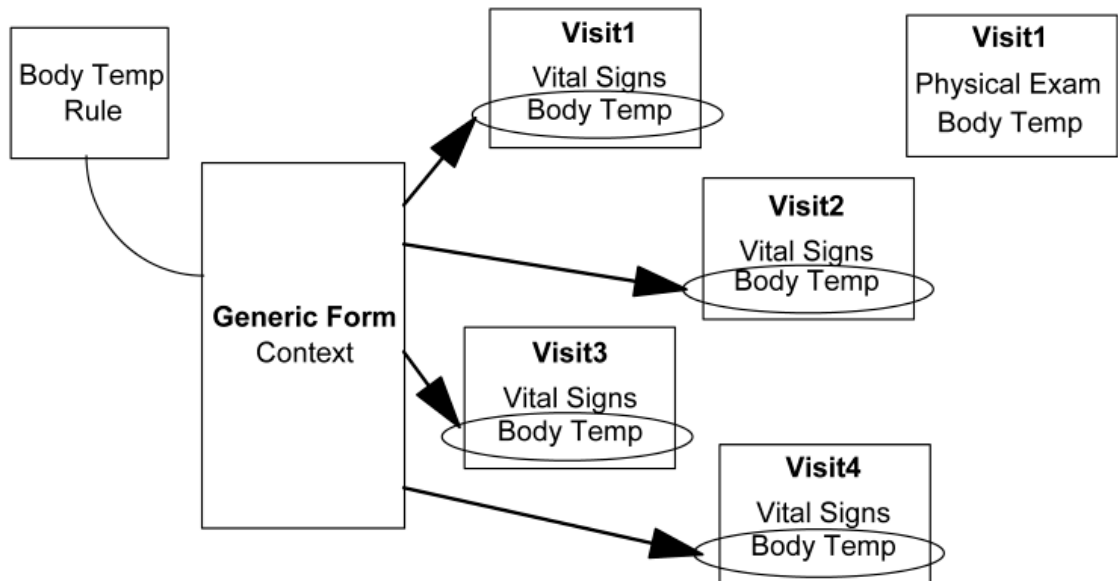
Context types

There are three types of rule contexts:

- Generic Form
- Generic Item
- Specific Visit

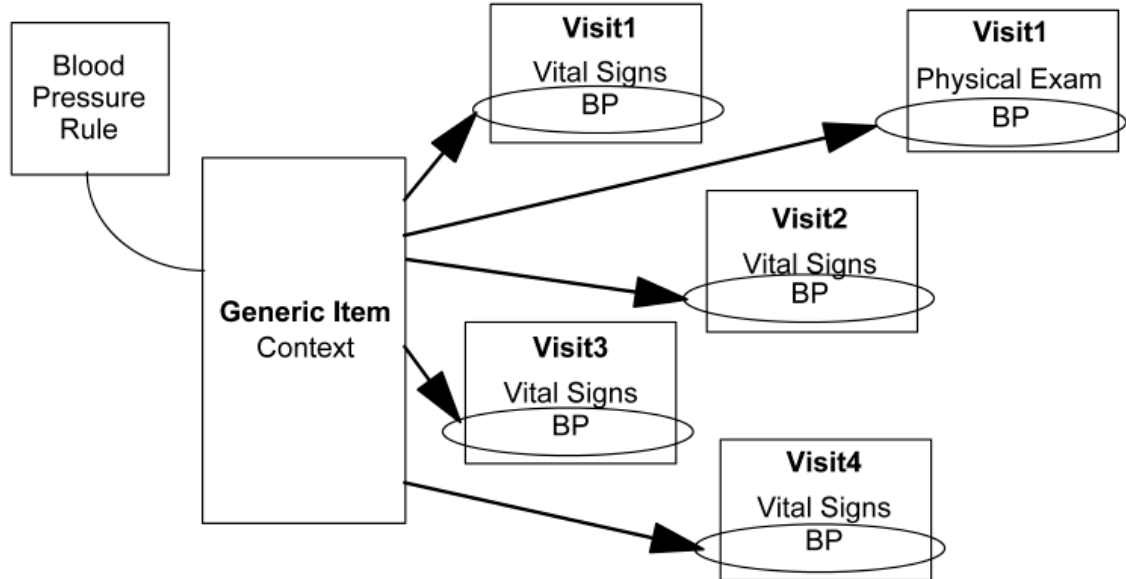
Generic form context

A Generic Form context associates the rule with an item in a form in every instance of the **form** in the trial. For example, you can associate a rule with the body temperature item on the Vital Signs form using a Generic Form context. Each time that data is entered or updated in the body temperature item on the Vital Signs form, the rule runs, regardless of the visit. If the body temperature item is used on a form other than Vital Signs, the rule is not associated with it.



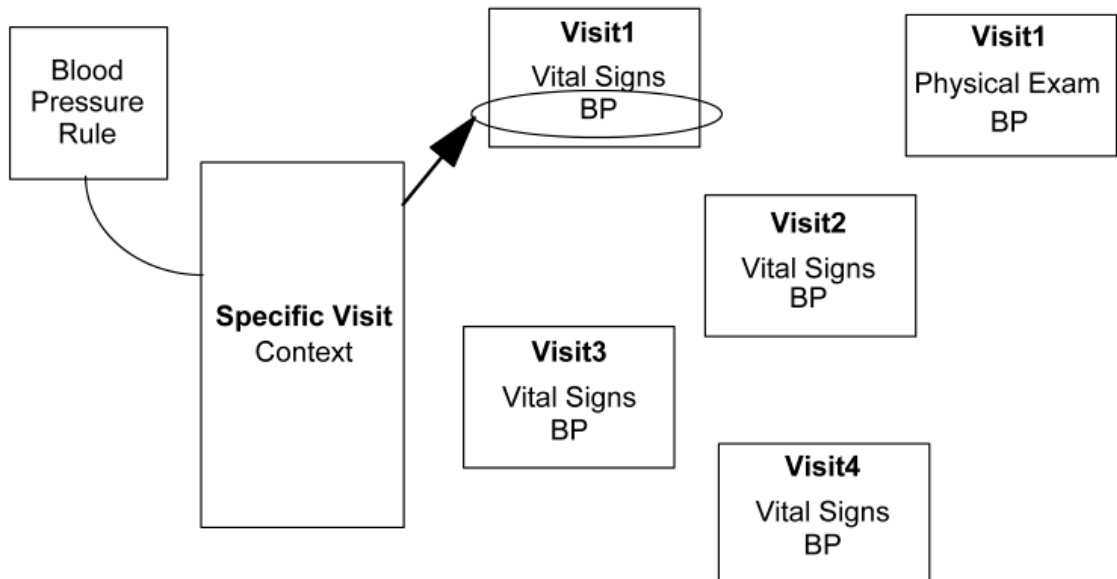
Generic item context

A Generic Item context associates the rule with a particular item in every instance of the **item** in the trial. For example, you can associate a rule with the blood pressure item on the Vital Signs form using a Generic Item context. Each time that data is entered or updated in the blood pressure item, regardless of form and visit, the rule runs.



Specific visit context

A Specific Visit context associates the rule with one item in a specific visit and form. For example, if you want to run a rule on the blood pressure item on the Vital Signs form in the first visit only, use the Specific Visit context. The rule does not run on other instances of the blood pressure item in other forms.



Context anatomy

A rule context consists of:

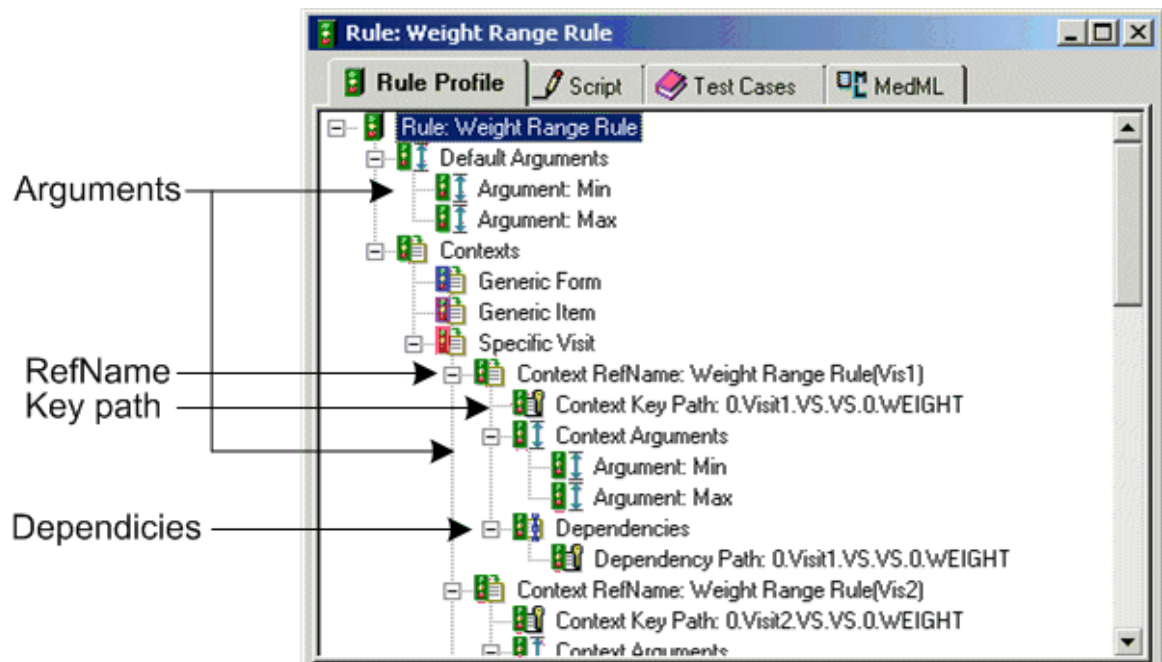
- A RefName that uniquely identifies the context.
- A key path specifying the item—in its section, form, and visit—to which the rule is applied. The format of a key path varies according to the context type:

Context type	Key path format
Generic Form	RefName path of item, with 0 in the visit position
Generic Item	Generic_Item: <i>itemname</i>
Specific Visit	RefName path of item

- **Arguments**—Any values that are substituted into the rule script when it runs against the item specified in the key path. For more information, see *About arguments* (on page 288).
- **Dependencies**—Additional items used in the processing of the rule script. For more information, see *About rule dependencies* (on page 289).
- **Properties**—Properties specifying the RefName, help text, and event specific to the context.

A single rule can have many contexts; for example, a rule that tests for range limits can be applied to very different items on multiple forms, passing arguments that are specific to the context each time it is applied. Thus a rule that checks for valid ranges on a height item can use arguments that specify minimum and maximum number of inches in the height context. When the same rule is applied to a pulse rate item, it can use a different set of arguments that specify a minimum and maximum number of heartbeats.

In the InForm Architect application, a context and all of its components are represented as nodes in the Rule window for the rule with which it is associated.

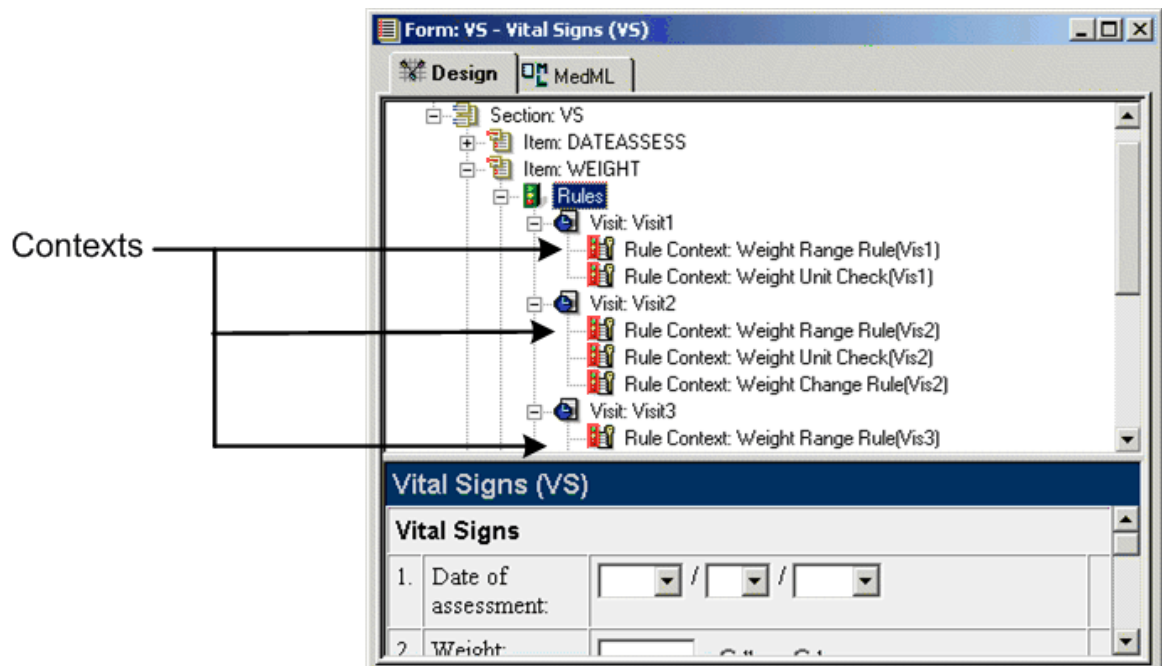


Context definition information appears in the Rule window, as part of the definition of the rule. When you select the Context RefName node in the Rule window, the Properties window reflects the properties of the context:

Rule: Weight Range Rule	
Property	Value
Context RefName	Weight Range Rule(Vis1)
Context Help Text	Weight is expected to be between 66-440 lbs or 30-200 kg.
Context Event	

Properties

In addition, the Form window shows the contexts associated with each item. The following figure illustrates:



Note: Only form rules, calculations, and randomization rules have contexts. Conversion rules, which are applied to unit definitions, do not.

About arguments

Arguments enable you to pass values to a rule script. By using arguments, you can create generic rule scripts that can be applied to many situations and reduce the overall number of rules you need to write.

Arguments are substituted into a rule script when the rule runs in context. For example, in the following script, “Min” and “Max” are arguments that represent allowable range limits. If the value of the item represented by the “item” argument is not within the range, then the rule fails, and a query with the text in the “qtext” argument is generated:

```
Min = patient.getargument("min")
Max = patient.getargument("max")
qtext = patient.getargument("qtext")
addpath = patient.getargument("addpath")
item = patient.getcurpath() & addpath
Patient.AddNamedValue "QUERYTEXT", qtext
result.rulepassed = 1

v_val = patient.getvaluerf(item, "", 0,0,0)

If cDbl(v_val) < cDbl(Min) or cDbl(v_val) > cDbl(Max) then
    result.rulepassed = 0
End If
```

When you create an argument definition as part of the process of defining a rule or a context, you specify the value to substitute. The value of an argument is one of its properties. The others are its name and type (date, float, numeric, or string).

Property	Value
Argument Name	Min
Argument Type	Numeric
Argument Value	0

Properties

Types of arguments

You can define arguments on two levels, default and context-specific:

- **Default** arguments provide default values to substitute in a rule. When the rule system evaluates a rule, it substitutes the values specified as default arguments unless context-specific arguments are defined. You define default arguments when creating a rule definition.
- **Context-specific** arguments apply to a rule only when it runs against the item specified by the context. Context-specific arguments override default arguments. You can only define a context-specific argument if there is a corresponding default argument for it to take the place of. You define context-specific arguments when creating a context definition.

Writing scripts to use arguments

To make an argument available to a rule script, set the value of a variable to the result of the `GetArgument` method of the Patient scripting object. `GetArgument` returns the value of the specified argument. For example, in the following script fragment, `GetArgument` returns the value of the Max argument:

```
HighTemp = Patient.GetArgument("Max")
Then, use the variable in the edit check logic.
```

Tip

Note that you can define an argument as an explicit value or as a `RefName` path. If you define it as a `RefName` path, first use `Patient.GetArgument` to retrieve the path, and then use `Patient.GetValueRF` to retrieve the value of the item that the path refers to. In the following example, the `PastValue` argument is defined with a `RefName` path as its `Value` property:

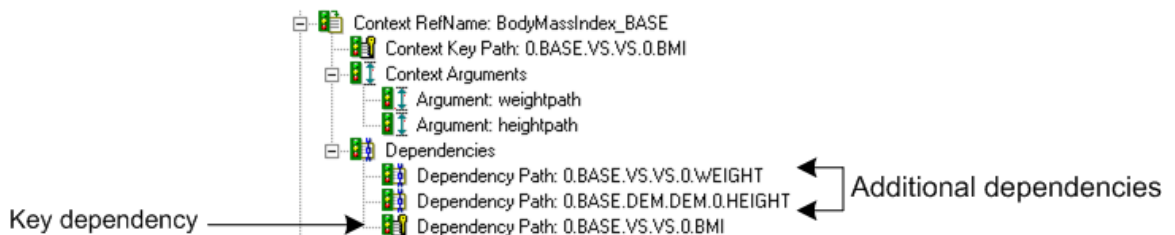
```
PastWeight = patient.getargument("PastValue")
BaseWt = Patient.GetValueRF(PastWeight, "", 0,0,0)
```

About rule dependencies

In addition to the context item, a rule may reference one or more additional items that contain information the rule needs in order to perform its evaluation or calculation. These items—each in its section, form, and visit—are called *dependencies*. In fact, the context item is also a rule dependency, thus, every rule has at least one dependency (the *key dependency*) representing the context item. If the rule script requires other items in addition to the context item, you define *additional dependencies* pointing to those items. These other items can be in different forms or visits than the context item.

Additionally, an association between two repeating forms can be a dependency of a rule context. When you define an association as a context dependency, the rule runs when a user selects or deselects the check box control that creates or dissolves an association between a specific instance of a repeating form and the other form in the association.

In the InForm Architect application, the key dependency is identified with a key icon:



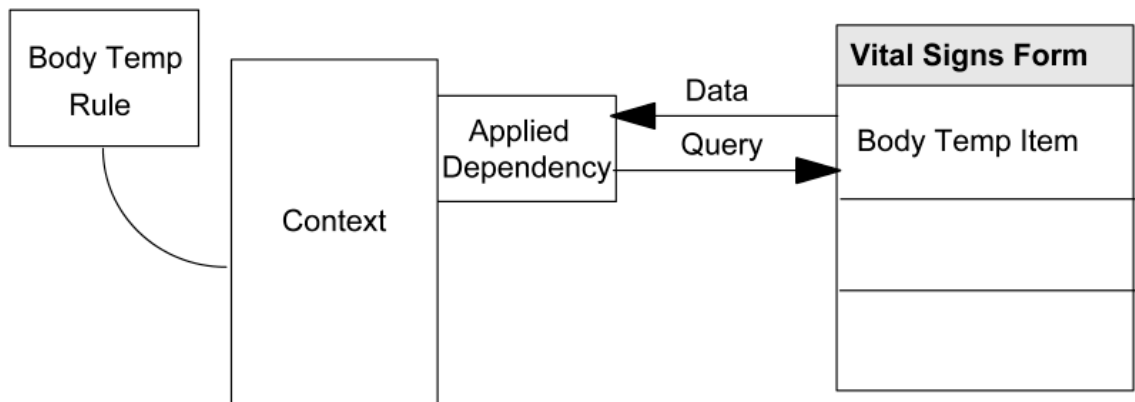
Types of dependencies

When you associate a rule with the items it runs against, you specify the type of dependency each item represents in its context: applied, dependency, or trigger. Rule processing begins when an item designated as any of these types of dependencies receives an initial entry of data or is updated.

Applied

The rule script applies the outcome of the rule to the item with the applied dependency. If a form rule fails, the item designated as the applied dependency receives a query. If the rule performs a calculation or assigns a randomization drug kit, the item designated as the applied dependency receives the result of the calculation or randomization. For a form rule to run, this item and all items classified as dependencies must have an entered value. For a calculation rule to run, this item does not need an entered value.

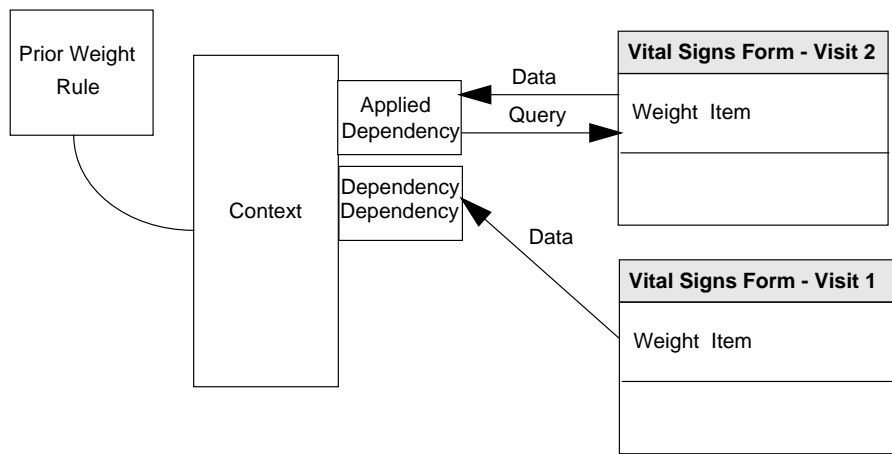
In the following example, the Body Temp item is an applied dependency of the context for the Body Temp rule. The rule runs when the Vital Signs form is submitted with data in the Body Temp item. If the rule fails, the Body Temp item receives a query.



Dependency

The rule script uses the value entered for this item in performing its calculations or edit checks. For the rule to run, this item and any items classified as applied dependencies must have an entered value.

In the following example, the Prior Weight rule compares the values of the Weight item in Visits 1 and 2 when the Vital Signs form in Visit 2 is submitted and issues a query on the item in Visit 2 if the rule fails. The rule runs only if both Weight items contain data.



Tip

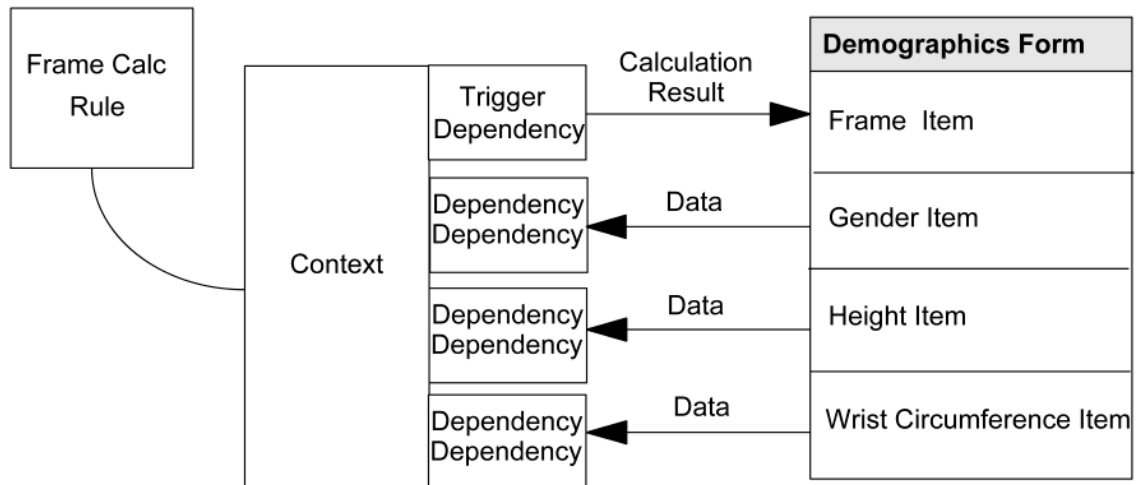
One way to code a script that uses this type of relationship is to define the RefName path of each Weight item as a context-specific argument, as illustrated in the following script. This script compares the current weight, an item identified as an applied dependency, with the weight in the previous visit, an item identified as a dependency type dependency. The PastValue argument, defined in the context for each visit, has a value of the RefName path of the Weight item in the previous visit.

```
PastValue = patient.getargument("PastValue")
BaseWt = Patient.GetValueRF(PastValue, "", 0,0,0)
CurrWt = Patient.GetValueRF(Patient.GetCurPath(), "", 0,0,0)
Patient.RulePassed = TRUE
If (BaseWt <> 0 and CurrWt <> 0) Then
  BaseLowWt = BaseWt - (BaseWt*.05)
  BaseHiWt = BaseWt + (BaseWt*.05)
  If (CurrWt < BaseLowWt or CurrWt > BaseHiWt) Then
    Patient.RulePassed = FALSE
    patient.addnamedvalue "QUERYTEXT",
      "Weight change is more than 5% difference from
      previous visit; please verify."
  Else
    Patient.RulePassed = TRUE
  End If
End If
```

Trigger

The rule script can use the value entered for this item in performing its calculations or edit checks. However, the rule does not require a value to be in this item in order to run.

In the following example, the Frame Calc rule determines the value of the Frame item by using the values of the Gender, Height, and Wrist Circumference items, which are Dependency dependencies. The rule runs if a submit occurs and the Gender, Height, and Wrist Circumference items all contain data. The Frame item, whose value is calculated by the rule, can be blank.



Note: This type of dependency is very useful when the rule is a calculation that depends on an item that is located in a different form. By firing the rule whenever a dependent item is updated, it ensures that the calculated item is recalculated even though the submission occurs in a form other than the one the rule is attached to. If you define an association as an additional dependency, the InForm Architect application enforces Trigger as the only allowed Dependency Type.

Considerations about additional dependencies

Consider the following when planning and defining additional dependencies:

- A Generic Item context cannot have additional dependencies.
- All dependencies associated with a context must have the same scope as the context. For example, if a context is defined as a Generic Form context, its dependencies must also be defined with Generic Forms.
- In a Generic Form context, additional dependencies can only be defined in the form specified in the key context.
- An association between two repeating forms can be a dependency of a rule context. When you define an association as a context dependency, the rule runs when a user selects or deselects the check box control that creates or dissolves an association between a specific instance of a repeating form and the other form in the association. A dependency defined for an association must be:
 - Attached to a Specific Visit context
 - Defined as a Trigger dependency

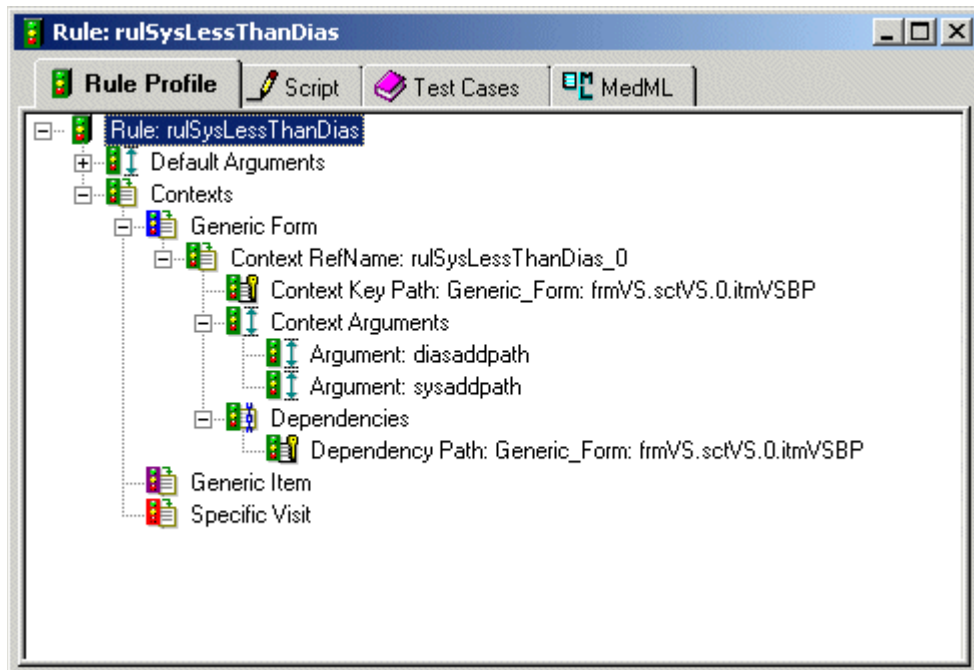
Creating rules

This section describes the steps to take in creating rules:

- 1 Create a new rule definition and specify its properties.
- 2 Write a rule script.
- 3 Define default arguments, if any, to pass to the script and specify their values.
- 4 Create contexts by attaching the rule to each context item against which it runs, and define context properties.
- 5 Define context-specific arguments and specify their values.
- 6 Define additional dependencies or adjust dependency definitions as necessary.

Creating a new rule definition

To create a new rule definition, select **File > New > Rule**. A new **Rule** window opens in the **Design Workspace**.



The Rule Profile tab of the Rule window displays a hierarchical view of the rule definition that provides access to the rule's default arguments and to the contexts in which the rule is applied. As you define each context, the window shows the arguments and dependencies specific to the context.

Note: When using the InForm Unplugged software, it is important to make sure you synchronize rules before making modifications to them.

Specifying rule properties

After opening a new Rule window, you can specify its properties. In the **Properties** tab of the **Properties** window, specify the definition of each applicable property. The following table describes the rule properties:

Property	Description
RefName	<p>RefName of the component.</p> <p>REQUIRED.</p> <p>For rules about the use of RefNames, see <i>RefNames</i> (on page 89)</p>
Description	<p>Description of the purpose of the rule. This text is displayed in the list of rules that appears on the Rules View screen in the Admin area of the InForm application.</p> <p>OPTIONAL</p>
Help Text	<p>Default description of the edit check being performed. This text is displayed along with CRF Help for the item against which the rule runs. Note that you can define context-specific help text by defining the Help Text property at the context level, as described in <i>Creating a context</i> (on page 300). If you define context-specific help text, it overrides the default text specified in the Help Text property.</p> <p>OPTIONAL</p>
Event	<p>RefName of the event that is triggered when the rule runs. Select one event from the list of defined events. The event determines the default text of a query generated if the rule fails. Note that you can define a context-specific event by specifying the Context Event property at the context level, as described in <i>Creating a context</i> (on page 300).</p> <p>If you define a context-specific event, it overrides the default event specified for the rule.</p> <p>REQUIRED. for a form rule, ignored for other types of rules.</p>
Active	<p>Active (the default) or Inactive, indicating whether a rule is active and will fire under the appropriate circumstances if associated with a rule context.</p> <p>OPTIONAL</p>
Type	<p>Rule type: calculation, conversion, form rule, or randomization rule.</p> <p>REQUIRED.</p>

Property	Description
UUID	String that uniquely identifies the component across all databases, trials, and machines. Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters.
Design Note	Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only. OPTIONAL.

Writing a rule script

To create a rule script, you can use any text editor and then copy the script into the Script tab of the Rule editor window, or you can enter script text directly into the Script tab.

When designing rule scripts, consider the options described in *Planning edit checks* (on page 10).

Using script editing features

The Script tab of the Rule editor window displays the text of a rule script. VBScript keywords and names of objects, methods, and arguments are color-coded for easy recognition. When working in the Script tab, you can use the following editing features:

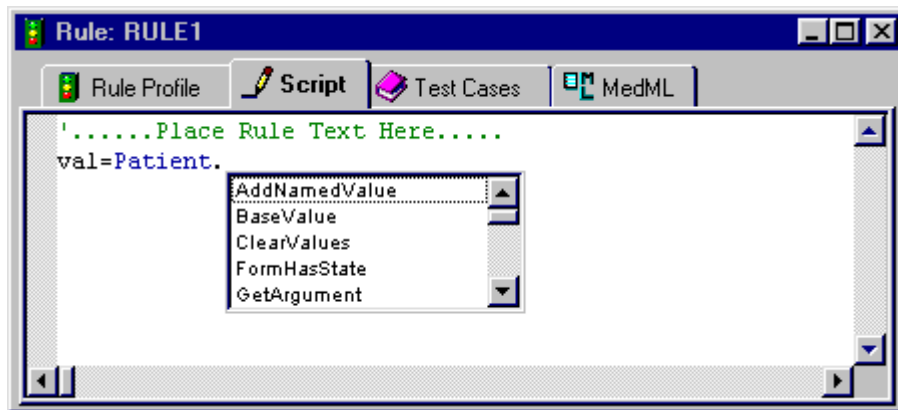
- Pick a method for a rule object by selecting the method from a pop-up list.
- Build the RefName path of a control by picking RefNames from a list.
- Insert or remove bookmarks.
- Cut, copy, paste, undo, and redo typing.

The following table provides details about each of these features:

Tool	Description
Pop-up methods list	When you type the name of a rule object and follow it with a period, the InForm Architect application displays a pop-up list of the methods associated with the object To insert the name of a method, click its name in the list or scroll to it and press Enter .
Build RefName	You can pick the RefName path of a control by using the Build RefName tool. For information on how to use this tool, see <i>Using the Build RefName tool</i> (on page 316).

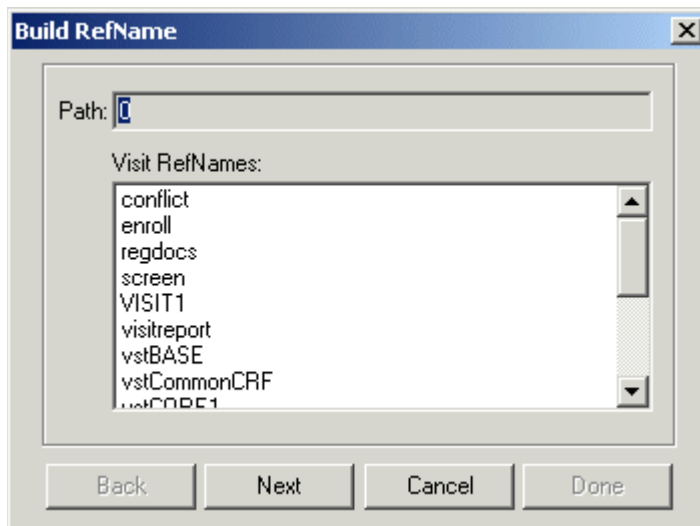
Tool	Description
Toggle Bookmark	<p>The InForm Architect application enables you to insert a bookmark into the body of your script text, or to remove a previously inserted bookmark. You can use the buttons in the Edit toolbar to navigate from bookmark to bookmark.</p> <p>To insert a bookmark:</p> <ol style="list-style-type: none"> 1. Place the cursor on the line where you want to insert the bookmark. 2. Right-click in the Rule editor window. 3. Choose Toggle Bookmark from the pop-up menu. <p>To remove a bookmark, follow the same steps, but begin by placing the cursor on the line containing the bookmark you want to remove.</p>
Undo	<p>When you change the text in a script window by typing or by cutting, copying, or pasting, you can undo the previous edit. Right-click in the Rule editor window, and choose Undo from the pop-up menu.</p>
Redo	<p>After undoing a change, you can reinstate it. Right-click the Rule editor window, and choose Redo from the pop-up menu.</p>
Cut	<p>To cut a selection of text and store it in the Microsoft Windows Clipboard:</p> <ol style="list-style-type: none"> 1. Highlight the text to cut. 2. Right-click in the Rule editor window. 3. Choose Cut from the pop-up menu.
Copy	<p>To copy a selection of text and store it in the Microsoft Windows Clipboard:</p> <ol style="list-style-type: none"> 1. Highlight the text to copy. 2. Right-click in the Rule editor window. 3. Choose Copy from the pop-up menu.
Paste	<p>To paste the contents of the Microsoft Windows Clipboard into a selected location:</p> <ol style="list-style-type: none"> 1. Place the cursor where you want to paste the text. 2. Right-click in the Rule editor window. 3. Choose Paste from the pop-up menu.

The following figure illustrates the **Pop-up methods** list:



Note: If you type the text of a rule script directly into the Rule editor window, each time you type the name of an object, the InForm Architect application displays the methods associated with the object in a pop-up list from which you can select the method you want.

The following figure illustrates the **Build RefName** tool dialog box:



Rule script processing

Examples of typical processing flows for each type of rule script follow. The objects and methods mentioned in this section are described in *Rule objects and methods* (on page 319), along with examples of several types of scripts.

Conversion rules

Conversion rule scripts specify the factor to use when converting units between the entered value of a data item and the normalized value, which is the value of the item stored in the database after conversion. A typical conversion rule, which is associated with a unit definition, loads the Result property of the Data object with the result of a conversion calculation.

Form rules

A typical form rule script:

- 1 Gets the normalized or entered value of all items it needs for performing an edit check by using the GetValueRF or GetEnteredValue method of the Patient object.
- 2 Gets the value of any arguments it needs by using the GetArgument method, if the rule is designed generically with parameters to accept arguments.
- 3 Checks the value against a standard. This standard can be hard coded or can be passed in as arguments, if the rule is designed generically with parameters to accept arguments.
- 4 Based on the result of the edit check, indicates whether the rule passes or fails by setting the RulePassed property of the Result object.

Calculation rules

A typical calculation rule script:

- 1 Gets the normalized or entered value of all items it needs for performing a calculation by using the GetValueRF or GetEnteredValue method of the Patient object.
- 2 Gets the value of any arguments it needs by using the GetArgument method, if the rule is designed generically with parameters to accept arguments.
- 3 Performs a calculation with the obtained data values.
- 4 Loads the Result property of the Result object with the result of the calculation.

Randomization rules

A randomization rule script:

- 1 Gets the patient data needed for stratification by using the GetValueRF or GetEnteredValue method of the Patient object.
- 2 Checks the patient data against a standard that determines the drug kit list from which the patient's trial medication is assigned.
- 3 Loads the Randomization object with patient data and stratification information.
- 4 Calls the GetNextKit method, which assigns the drug kit number.

Defining default arguments

When you create a rule script, you can set up parameters for values such as RefNames, comparison values, and constants. In this step, define default arguments to pass to those parameters. If the rule has no context-specific arguments, the InForm application passes the default arguments you define here to the rule script when it runs. Any arguments defined for a specific context override these arguments.

Note: If you hard-code the RefNames and values in the script, this step is not necessary. Instead, when you do not use arguments, you must define separate rules and rule scripts for every item you want to run a rule against.

Adding a default argument

To add a default argument:

- 1 With the **Rule Profile** tab of the **Rule** window visible, right-click the **Default Arguments** node and choose **Add** from the pop-up menu.

The InForm Architect application adds a new argument line under the **Default Arguments** icon.

- 2 In the **Properties** window, specify the following information in the **Value** column:

- Name of the argument.
- Argument type: String, Numeric (integer), Float, or Date.
- Value of the argument.

Rule Context: Pulse Range Check Rule	
Property	Value
Argument Name	Min
Argument Type	Numeric
Argument Value	0

Properties

Modifying a default argument

To modify a default argument:

- 1 In the **Rule Profile** tab of the **Rule** editor, double-click the argument you want to modify.
- 2 In the **Properties** window, edit the **Value** cell of any of the argument properties: Argument Name, Argument Value, or Argument Type.

Note: When you change the Argument Name or Argument Type, the InForm Architect application makes the same changes to each corresponding context-specific argument. In order to do this, it collapses the tree in the Rule editor.

Removing a default argument

To remove a default argument:

- 1 In the **Rule Profile** tab of the **Rule** editor, right-click the argument you want to remove.
- 2 Choose **Delete** from the pop-up menu.

The InForm Architect application deletes the default argument and also deletes each corresponding context-specific argument.

Creating a context

You can create a context in either of the following ways:

- In the **Form** window, drag the rule onto the appropriate location.
- In the **Rule** window, right-click in one of the three **Contexts** type nodes, and choose **Add** from the pop-up menu.

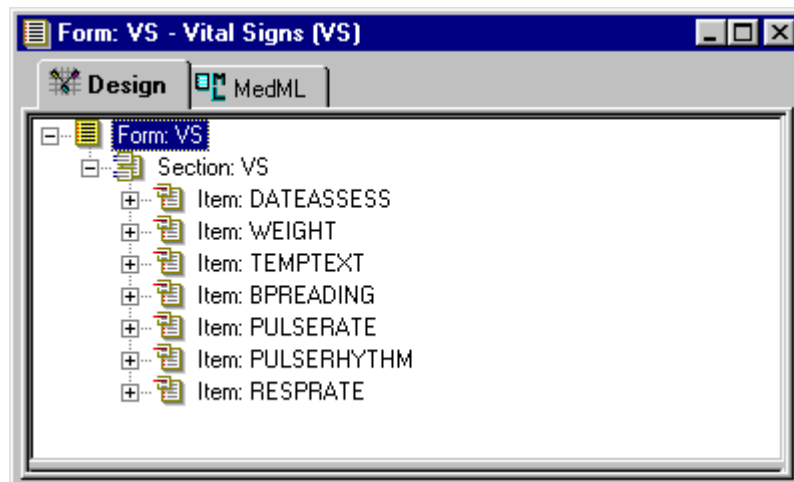
Dragging a rule onto a form

The simplest way to create a context is to drag the rule onto the appropriate location in the form window. To attach a rule to items on multiple different forms, perform the following steps for each applicable form and item:

- 1 Open the window for the form containing the item or itemset to which you are attaching the rule.

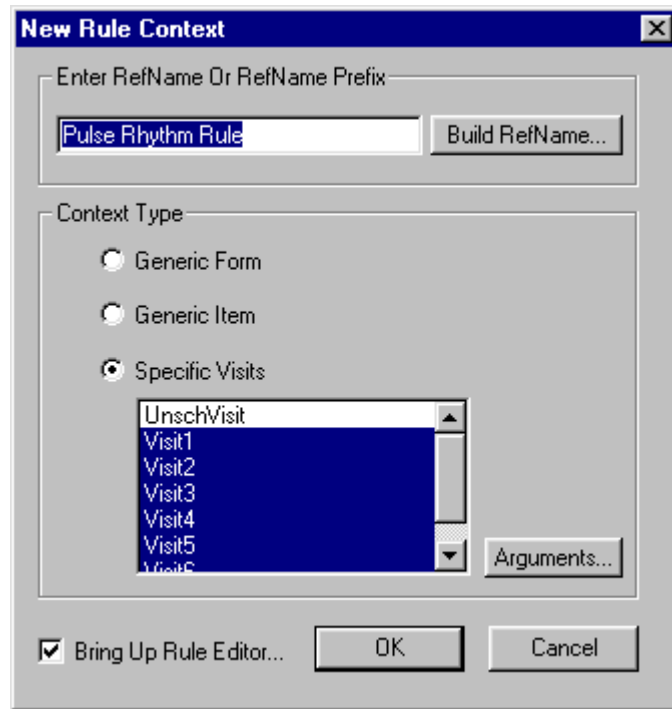
Note: You can attach a rule to an item or to an itemset. However, you cannot attach a rule to an item within an itemset.

- 2 Expand the section node so that the item to which you want to attach the rule is visible.



- 3 In the **Trial Objects** window, expand the **Rules** node so that the rule you are attaching is visible.
- 4 Drag the rule into the **Form** window and place it on the item or itemset.

The **New Rule Context** dialog box opens, enabling you to specify the **RefName** of the context and the type of context with which you want to associate the rule.

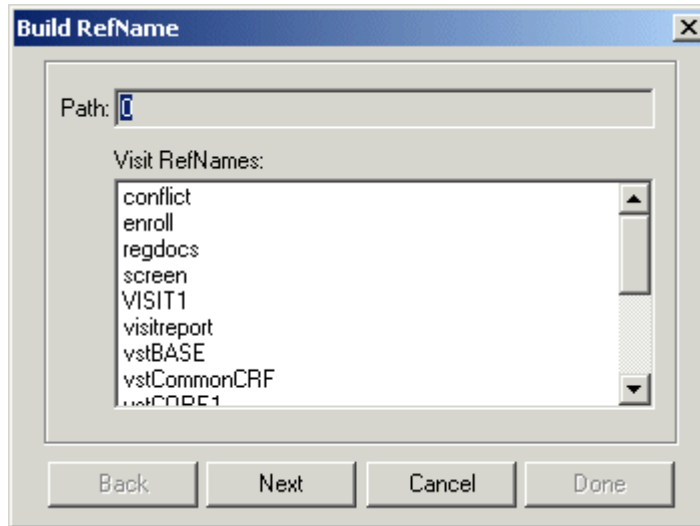


5. Verify the **RefName** or **RefName** prefix. By default, the InForm Architect application creates a Context RefName by appending one of the following to the RefName of the rule:
 - **Generic Form context**—0, to indicate all visits. For example, the default Context RefName when the PulseRhythmRule rule is attached as a Generic Form context is PulseRhythmRule_0.
 - **Generic Item context**—0, to indicate all visits, and a sequential number to indicate the order in which contexts are attached. For example, the default Context RefName when the PulseRhythmRule is attached as Generic Item context is PulseRhythmRule_0_1. If the same rule is attached as a Generic Item context to another item, the Context RefName for the second attachment is PulseRhythmRule_0_2.
 - **Specific Visit context**—Visit name. For example, the default Context RefName when the PulseRhythmRule is attached as a Specific Visit context to an item in Visit1 is PulseRhythmRule_Visit1.

To change the context RefName to a RefName path:

- a. Click the **Build RefName** button.

The Build RefName dialog box appears.



- b For each component of the RefName path, select the component from the list in the dialog box.
- c Specify whether you want to identify components of the RefName path with underscores or initial capital letters by selecting the **Underscored** or **Capitalized** option.
- d Click **Next**, or, to backtrack to a previous component, click **Back**.

The InForm Architect application builds the path from your selections and displays it in the **Path** field at the top of the dialog box.

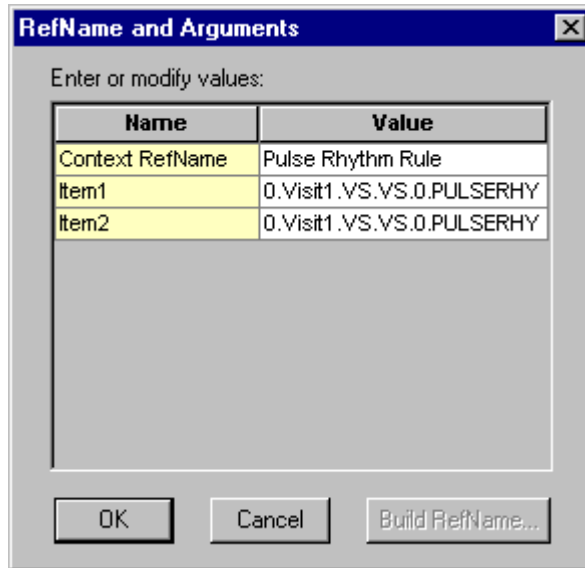
- e When the RefName path is complete, click **Done**.

The InForm Architect application redisplay the RefName and Arguments dialog box.

- 6 Select the context type with which you want to associate this rule. For more information on Context types, see *About rule contexts* (on page 284).

If you select **Specific Visits**, you need to select the visits on which you want to run the rule. Use the **Ctrl** key to select or unselect multiple visits.

- 7 To define arguments for the rule:
 - a Click the **Arguments** button. The **RefName and Arguments** dialog box opens, enabling you to specify the values of the context-specific arguments.



b If you want the arguments to have context-specific values, specify those values.

If the Argument Type is String, you can specify a RefName path as the value of an argument:

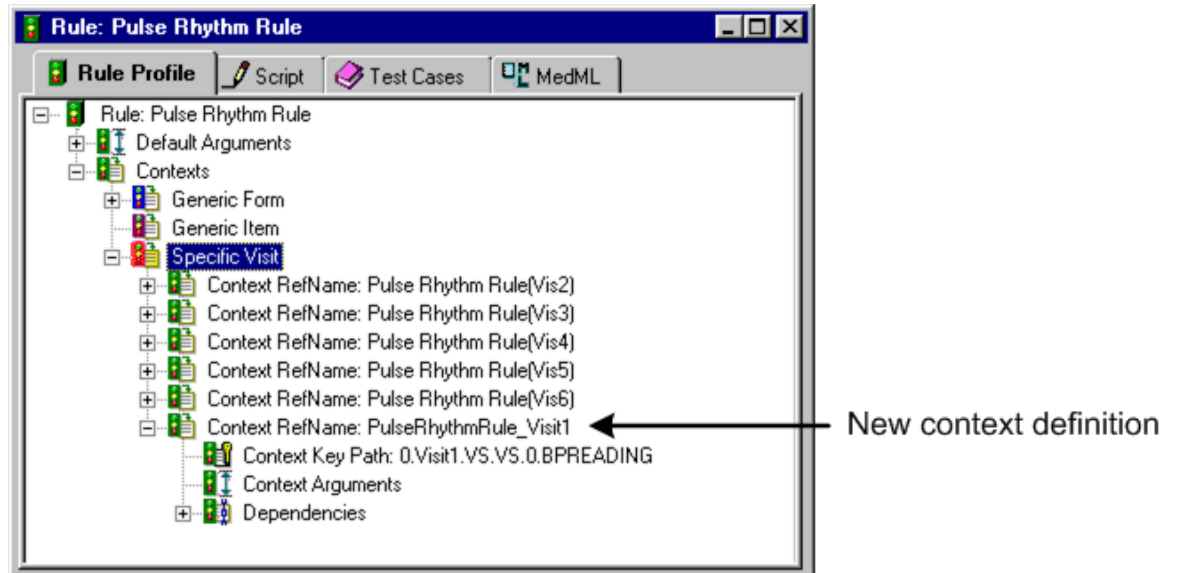
- c Select the **Value** cell of the argument.
- d Click the **Build RefName** button.
- e In the **Build RefName** dialog box, create a RefName path as described in *Using the Build RefName tool* (on page 316).
- f In the **Build RefName** dialog box, click **Done** to return to the **RefName and Arguments** dialog box.

Note: If you do not change the default value of an argument, the InForm Architect application does not add a new argument to the context, and the rule uses the default argument values when it runs in this context.

- g In the **RefName and Arguments** dialog box, click **OK** to return to the **New Rule Context** window.
- 8 Optionally, to display the **Rule** window when the attachment is created, select the **Bring Up Rule Editor** check box.
- 9 Click **OK**.

The **Form** window shows the attachment of the rule to the visit where you dragged it, and the **Rule** window, if you selected the **Bring Up Rule Editor** check box, shows the addition of the new context.





- 10 Define the remaining properties of the context by selecting the **Context RefName** node and editing its properties in the **Properties** window. The following table describes all of the properties that make up a context definition.

Property	Description
Context RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
Context Help Text	Description of the edit check being performed in this context. This text is displayed along with CRF Help for the item against which the rule runs. If you define context-specific help text, it overrides the default text specified in the Help Text property for the rule. OPTIONAL.
Context Event	RefName of the event that is triggered when the rule runs in this context. Select one event from the list of defined events. The event determines the default text of a query generated if the rule fails. An event defined for a context overrides the event specified as the default for the rule. OPTIONAL.

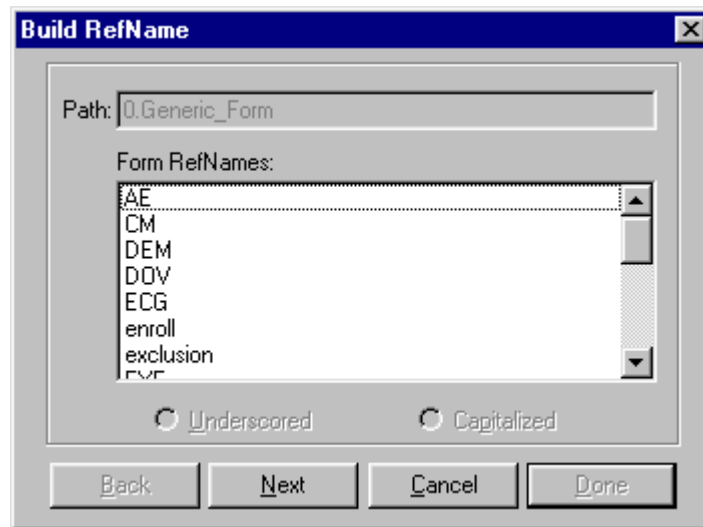
Right-clicking the Contexts node

To create a context by adding it in the Rules window:

- 1 In the **Rules** window, right-click one of the three **Contexts** nodes.
- 2 In the pop-up window, click **Add**.

The Build RefName dialog box appears. Note that when you open the dialog box, the Path is partially filled in. The contents of the **Path** box depend on the amount of information that is known based on the context type of the new context. Similarly, the contents of the list box vary based on the context type.

For example, the following figure illustrates the **Build RefName** dialog box for Generic Form context. Specify the path starting with the form name.



- 3 For each component of the RefName path, select the component from the list in the dialog box.
- 4 Click **Next**, or, to backtrack to a previous component, click **Back**.

The InForm Architect application builds the path from your selections and displays it in the **Path** field at the top of the dialog box.

- 5 When the RefName path is complete, click **Done**.

The InForm Architect application inserts the path into the component definition you are creating. The **RefName and Arguments** dialog box opens, enabling you to specify the RefName of the context and the values of the context-specific arguments.



- 6 In the **RefName and Arguments** dialog box, optionally give the context a new Context RefName.

Note: If you do not specify a new Context RefName, the InForm Architect application creates a Context RefName by appending the visit number to the RefName of the rule.

To create a Context RefName from the RefName path of the context item:

- a Select the **Value** cell of the **Context RefName**.
- b Click the **Build RefName** button.
- c In the **Build RefName** dialog box, create a RefName path as described in *Using the Build RefName tool* (on page 316). Specify whether you want to separate components of the RefName path with underscores or initial capital letters by selecting the **Underscored** or **Capitalized** option.

- 7 If you want the arguments to have context-specific values, specify those values.

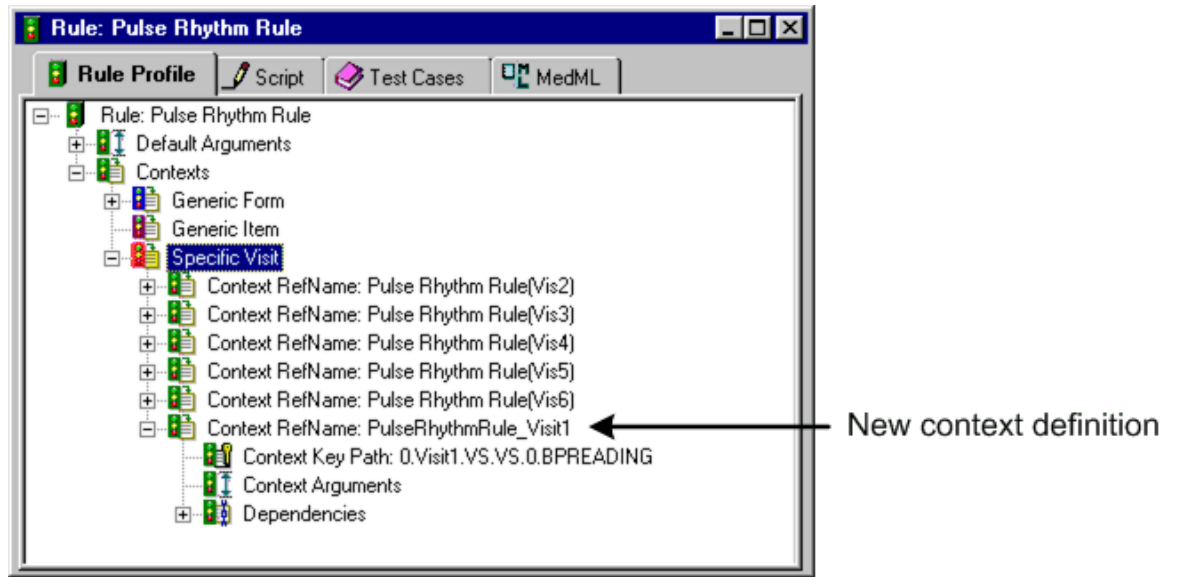
Note: If you do not change the default value of an argument, the InForm Architect application does not add a new argument to the context, and the rule uses the default argument values when it runs in this context.

Optionally, to specify a RefName path as the value of a string type argument:

- a Select the **Value** cell of the argument.
- b Click the **Build RefName** button.
- c In the **Build RefName** dialog box, create a RefName path as described in *Using the Build RefName tool* (on page 316).
- d When the RefName path is complete, click **Done**.

- 8 In the **RefName and Arguments** dialog box, click **OK**.

The **Rule** window shows the addition of the new context.



- 9 Define the remaining properties of the context by editing its properties in the **Properties** window. The following table describes all of the properties that make up a context definition.

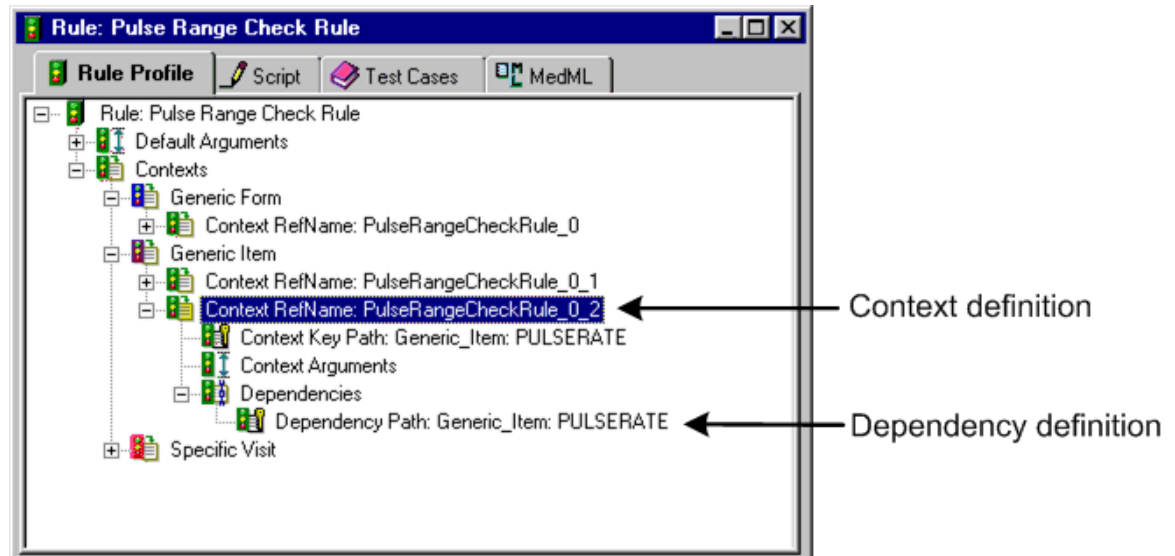
Property	Description
Context RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
Context Help Text	Description of the edit check being performed in this context. This text is displayed along with CRF Help for the item against which the rule runs. If you define context-specific help text, it overrides the default text specified in the Help Text property for the rule. OPTIONAL.
Context Event	RefName of the event that is triggered when the rule runs in this context. Select one event from the list of defined events. The event determines the default text of a query generated if the rule fails. An event defined for a context overrides the event specified as the default for the rule. OPTIONAL.

Automatic context and dependency creation

When you attach a rule to an item as described in the previous sections, the InForm Architect application creates, along with the context definition, an Applied dependency definition for the item to which you attached the rule:

- **If you create** a Generic Form context, the InForm Architect application creates a single dependency whose path is a RefName path with a 0 in the visit position.
- **If you create** a Generic Item context, the InForm Architect application creates a single dependency whose path contains the keyword `Generic_Item` and the RefName of the item to which it is attached.
- **If you create** a Specific Visit context for multiple visits, or if you attach the same rule to multiple items, the InForm Architect application creates contexts and dependencies for each applicable visit, form, section, and item combination.

These context and dependency definitions are visible in the Rule Profile tab of the Rule window. To see the dependency definitions for a context, expand the context definition by clicking the plus sign to the left of the context icon.



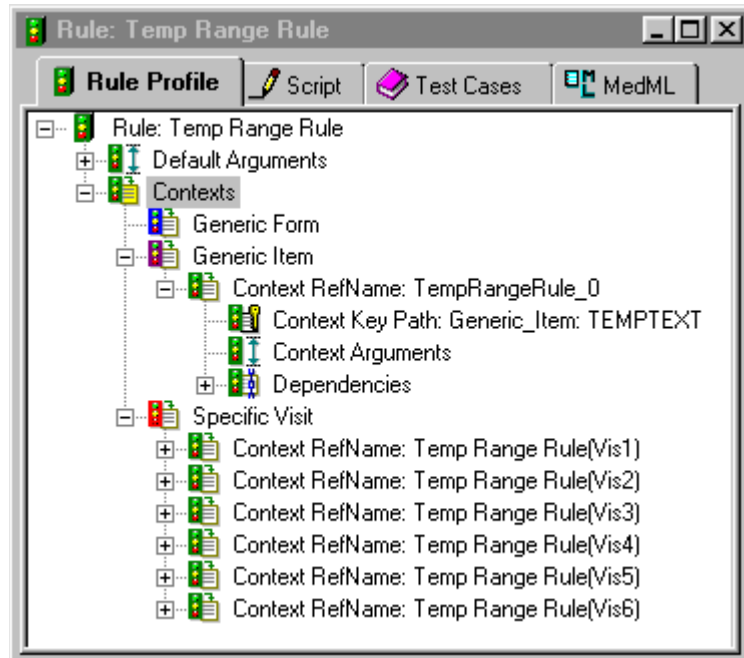
Viewing summarized context and dependency counts

When you create contexts for a rule, the InForm Architect application keeps track of how many contexts and how many dependencies of each type you are creating. You can view the following numbers:

- Number of contexts of each type (Generic Form, Generic Item, and Specific Visit) for a rule, along with total number of contexts for the rule
- Number of dependencies of each type (Applied, Dependency, and Trigger) for a specific context, along with total number of dependencies for the context

Viewing context type totals

To view a summary of context numbers, click the **Contexts** node in the **Rule** window.



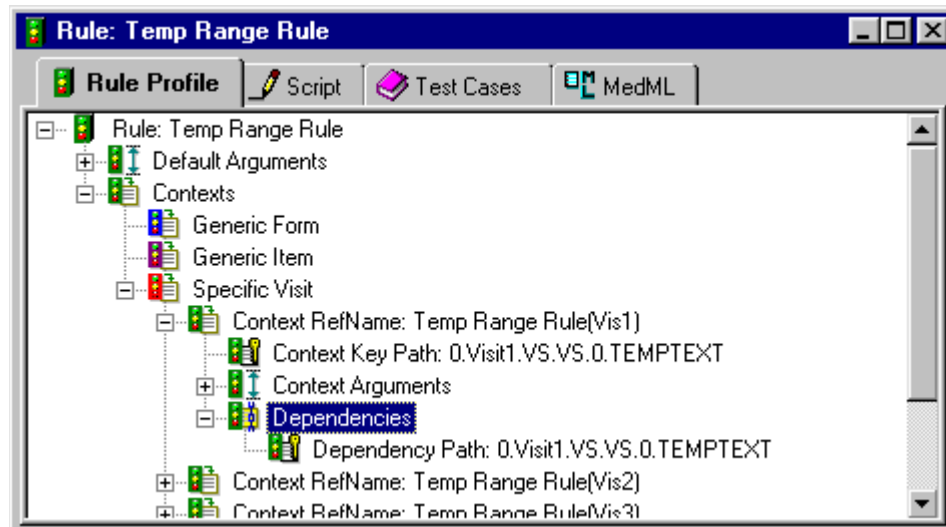
The **Properties** window displays the number of contexts of each type and the total number of contexts for the rule.

Rule: Temp Range Rule	
Property	Value
Generic Form	0
Generic Item	1
Specific Visit	6
Total Contexts	7

Properties

Viewing dependency totals for a specific context

To view a summary of dependency numbers for a specific context, click the **Dependencies** node of the context in the **Rule** window.



The **Properties** window displays the number of dependencies of each type and the total number of dependencies for the context.

Rule: Temp Range Rule	
Property	Value
Applied	1
Dependency	0
Trigger	0
Total Dependencies	1

Properties

Specifying context-specific arguments and values

When you create a generic rule script, you set up parameters for values such as RefNames, comparison values, and constants. The InForm application passes in the values of context-specific arguments to these parameters when a rule runs against a context and its dependencies.

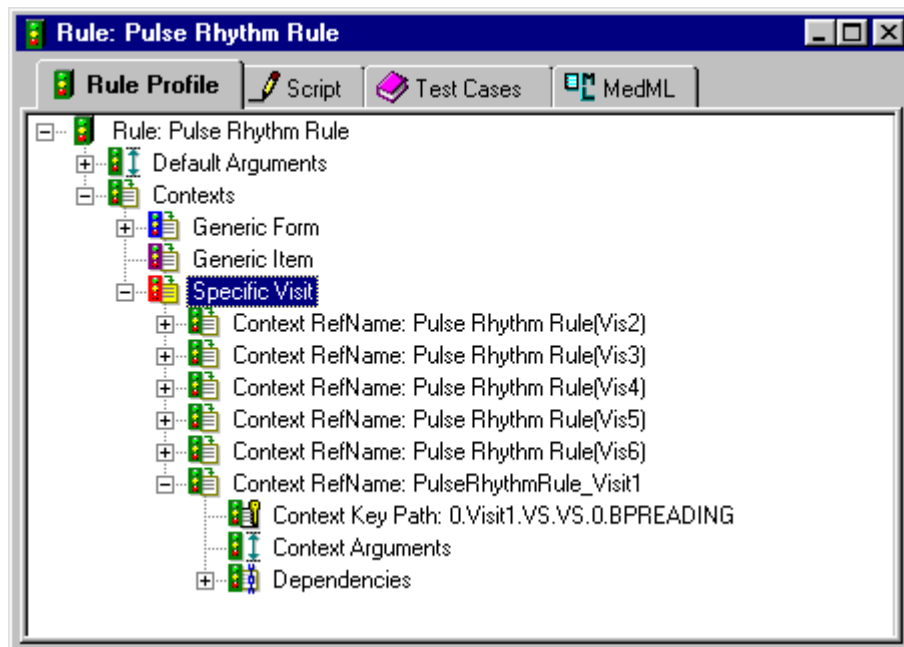
If you did not specify values for context-specific arguments when you defined the context, define them in this step. Any arguments defined for a specific context override the default arguments specified for the rule as a whole.

Note: When you define and update arguments for a specific context, you can only work with arguments that have previously been defined as default arguments. A context-specific argument must have the same Argument Name and Argument Type as its corresponding default argument. Therefore, the only context-specific argument property you can specify is the Argument Value.

Adding a context-specific argument

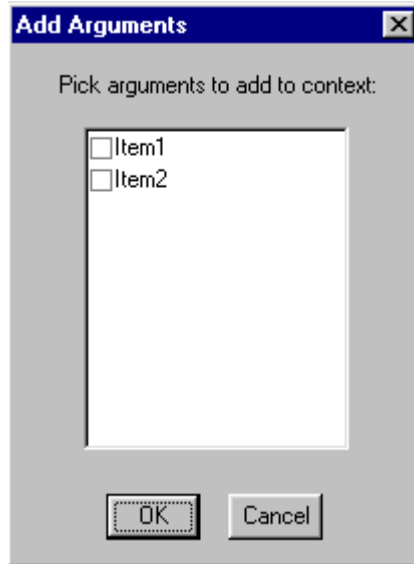
To add an argument for a specific context and its dependencies:

- 1 Click the **Rule Profile** tab of the **Rule** window.



- 2 Expand the context for which you want to specify arguments.
- 3 To add an argument, right-click the **Context Arguments** node and choose **Add** from the pop-up menu.

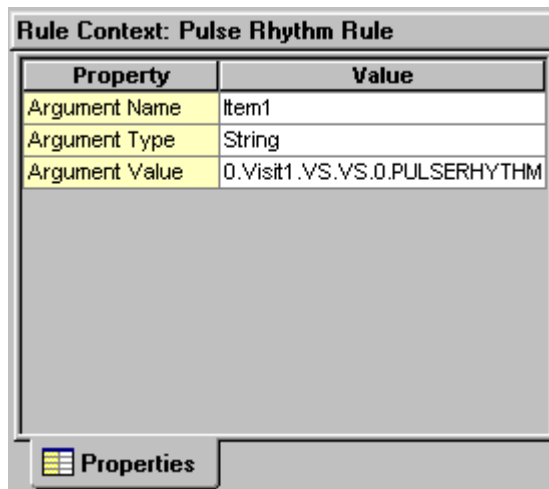
The **Add Arguments** dialog box opens, listing each default argument that has not been defined for the current context.



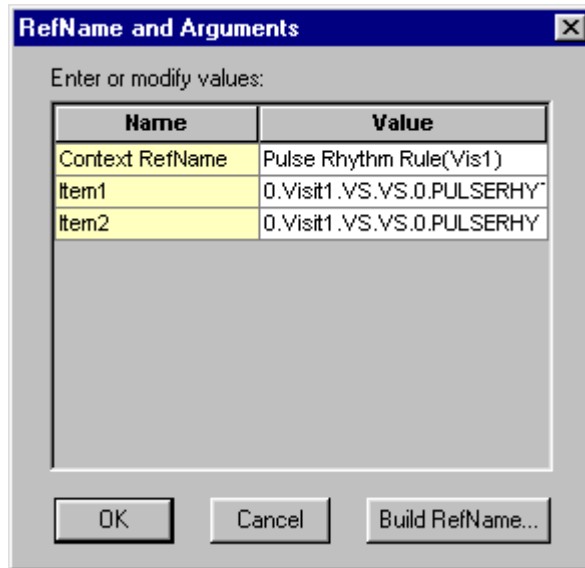
- 4 Select each default argument for which you want to specify a context-specific Argument Value.
- 5 Click **OK**.

For each selected argument, the InForm Architect application adds a new argument line under the **Context Arguments** node.

- 6 To specify the Argument Value of the new argument, do one of the following:
 - Select the argument. In the **Properties** window, specify the value of the argument in the **Value** column.



- Double-click the argument. In the **RefName and Arguments** dialog box, edit the **Value** cell for the argument and click **OK**.



Note: If you want the value of a string type argument to be a RefName path, you can click Build RefName to use the Build RefName tool, as described in *Using the Build RefName tool* (on page 316). You can not specify a RefName path for the value of a date, float or numeric.

Removing a context-specific argument

To remove an argument:

- 1 In the **Rule Profile** tab of the **Rule** editor, right-click the argument.
- 2 Choose **Delete** from the pop-up menu.

Defining additional rule dependencies

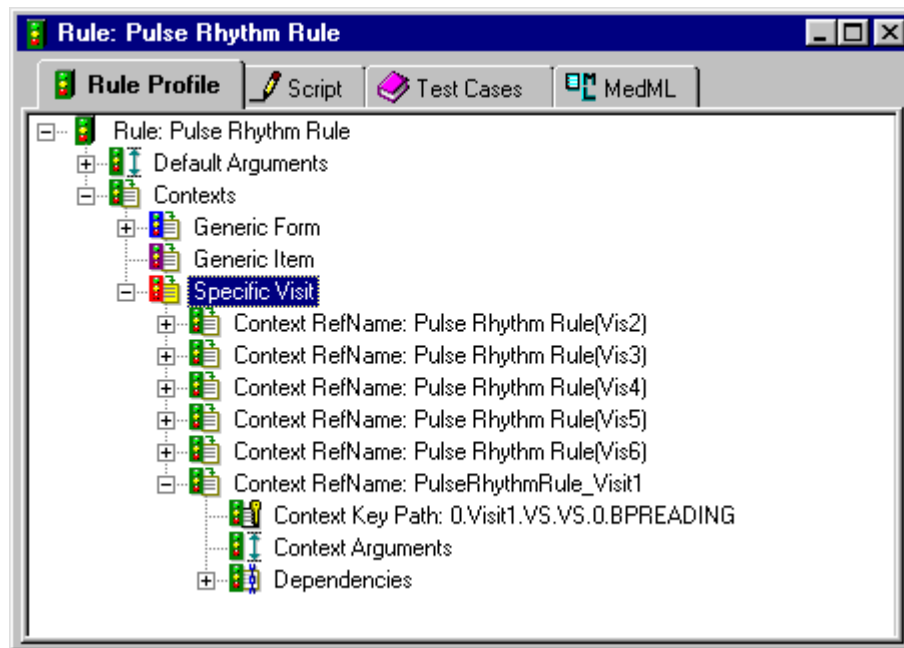
If a context has multiple dependencies—for example, if the rule is a calculation that requires data from another item before it can calculate the value of the item to which it is attached—you must now create the additional dependencies. This section describes how to create a dependency manually. Use these instructions also to modify an existing dependency.

Consider the following when planning additional rule dependencies:

- A Generic Item context cannot have additional dependencies.
- All dependencies associated with a context must have the same scope as the context. For example, if a context is defined as a Generic Form context, its dependencies must also be defined with Generic Forms.
- In a Generic Form context, additional dependencies can only be defined in the form specified in the key context.
- An association between two repeating forms can be a dependency of a rule context. When you define an association as a context dependency, the rule runs when a user selects or deselects the check box control that creates or dissolves an association between a specific instance of a repeating form and the other form in the association. A dependency defined for an association must be:
 - Attached to a Specific Visit context
 - Defined as a Trigger dependency

To define a rule dependency:

- 1 Open the rule definition by double-clicking it in the **Trial Objects** window.



- 2 Expand the **Contexts** node by clicking the plus icon next to the **Contexts** icon.
- 3 Expand the **Generic Form**, **Generic Item**, or **Specific Visit** context type node, as appropriate.

- 4 Expand the specific context for which you want to create or update a dependency.
- 5 Right-click the **Dependencies** icon and choose **Add** from the pop-up menu.

The Build RefName dialog box appears. Note that when you open the dialog box, the **Path** box is partially filled in. The contents of the **Path** box depend on the amount of information that is known based on the context type of the new context. Similarly, the contents of the list box vary based on the context type.

- 6 For each component of the RefName path, select the component from the list in the dialog box.
- 7 Click **Next**, or, to backtrack to a previous component, click **Back**.

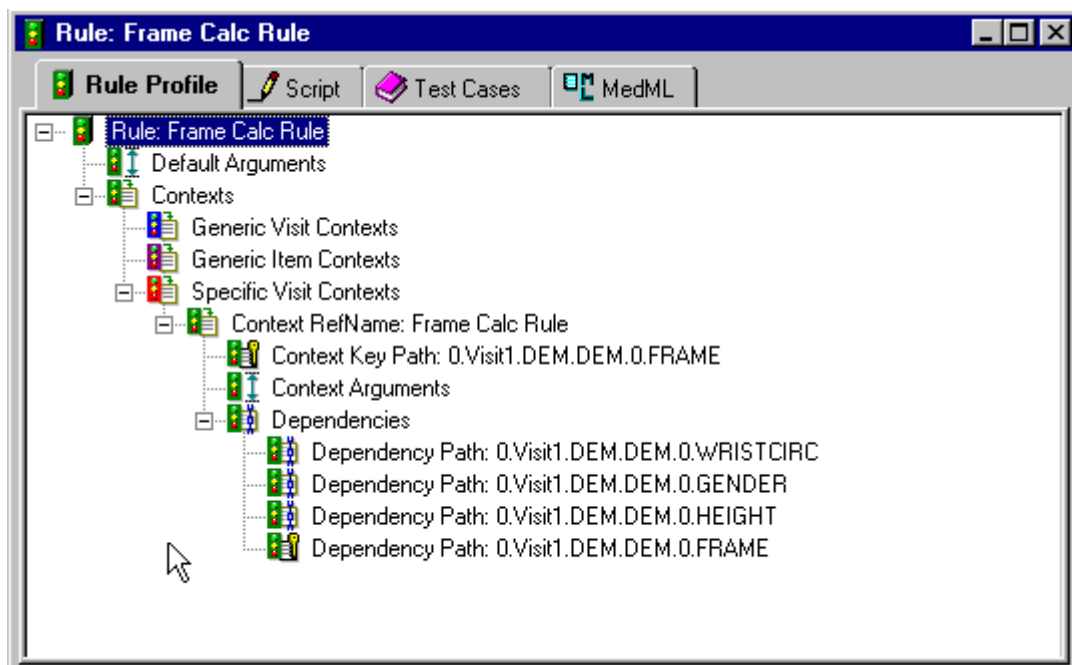
The InForm Architect application builds the path from your selections and displays it in the **Path** field at the top of the dialog box.

- 8 When the RefName path is complete, click **Done**.

The InForm Architect application inserts the path into the component definition you are creating.

- 9 Define the properties of the dependency by editing its properties in the **Properties** window. The following table describes the properties that you specify to create a dependency definition.

Property	Description
Dependency Path	<p>Item context with which the rule is associated. If necessary, update the path by using the Build RefName tool, as described in <i>Using the Build RefName tool</i> (on page 316).</p> <p>Note that this property is called Dependency Key Path when the selected dependency is the key dependency for the rule (the dependency created automatically when you add a context). You cannot change the Dependency Key Path. REQUIRED.</p>
Dependency Type	<ul style="list-style-type: none"> • Applied—The item identified by the set of RefNames in the dependency definition receives the result of a calculation rule or receives a query if a form rule fails. The rule fires only if the Applied item and all Dependency items have data. • Dependency—The value of the item identified by the set of RefNames in the dependency definition is used by the rule script in evaluating or calculating the Applied item. The rule fires only if the Applied item and all Dependency items have data. • Trigger—The rule script can use the value entered for this item in performing its calculations or edit checks. However, the rule does not require a value to be in this item in order to run. If you define an association as an additional dependency, the InForm Architect application enforces Trigger as the only allowed Dependency Type.



Removing a dependency

To remove a dependency definition:

- 1 In the **Rule editor** window, right-click the dependency to remove.
- 2 Choose **Delete** from the pop-up menu.

Note: You can only remove additional dependencies. You cannot remove the key dependency for the context.

Using the Build RefName tool

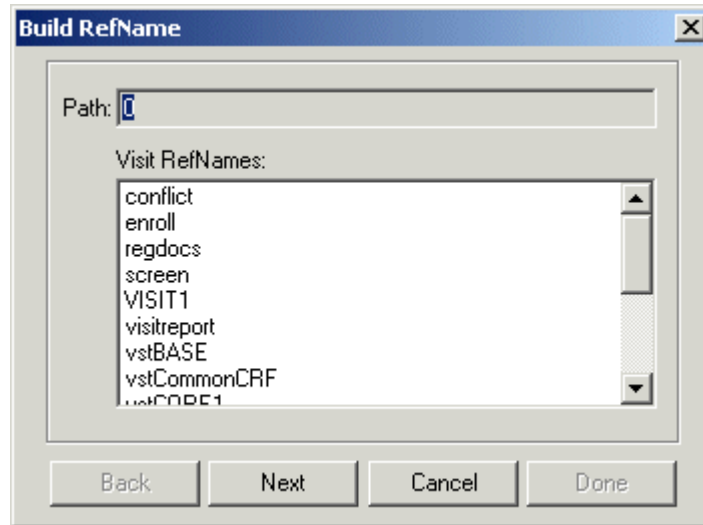
When you create a rule script or when you define rule contexts and dependencies, you need to specify the RefName paths of each item the rule references. The Build RefName tool enables you to build a RefName path by choosing its components from a series of lists. Additionally, you can use the Build RefName tool to modify context and dependency RefNames and to specify a RefName path as the value of a default or context-specific string type argument.

Note: To ensure that the Build RefName tool appears correctly, do not change the appearance of your Microsoft Windows desktop to use a nonstandard font. Keep the desktop appearance scheme set to “Windows Standard.”

General instructions for using the Build RefName tool

- 1 Display the **Build RefName** dialog box.

Note that the InForm Architect application partially fills in the **Path** field if it can obtain information from the location from which you display the dialog box. The label for the list of RefNames and the RefNames in the list change according to the context. For example, after you select a visit RefName, the label and list prompt you to select an itemset or item name.



- 2 For each component of the RefName path, select the component from the list in the dialog box.
- 3 Click **Next**, or, to backtrack to a previous component, click **Back**.

The InForm Architect application builds the path from your selections and displays it in the **Path** field at the top of the dialog box.

- 4 When the RefName path is complete, click **Done**.

The InForm Architect application inserts the path into the component definition you are creating.

Displaying the Build RefName dialog box

How you invoke the **Build RefName** dialog box depends on what you are doing in the InForm Architect application when you need to build a **RefName** path, as described in the following table:

To display the Build RefName dialog while...	Do this...
Creating or editing a script	<ol style="list-style-type: none"> 1. In the Script tab of the Rule window, place the cursor where you want to insert a RefName. 2. Right-click in the Rule editor window. 3. Choose Build RefName from the pop-up menu.
Updating the properties of a context	In the Rule window, double-click the Context RefName of the context you want to update.
Changing the RefName path of a dependency to associate the dependency with a different item in the same form and section	In the Rule window, double-click the Dependency Path of the dependency you want to update.
<p>Note: You cannot change the RefName path of the key dependency (the dependency created automatically when you add a context).</p>	
Defining an argument	<ol style="list-style-type: none"> 1. In the Rule window, double-click the argument for which you want to specify a RefName path. The RefName and Arguments dialog box appears. 2. Select the Value cell for the argument for which you want to specify a RefName path, and click the Build RefName button.

Rule objects and methods

Each type of rule has access to one or more objects and their methods. You use these objects and methods to obtain entered values and states and to pass values between rule, event, and execution plan processing. For information on the rule objects and methods that are available with the InForm application, see the MedML online help.

Form rule and calculation script examples

Calculation rule

The following calculation script determines the relative size of patients' skeletal frames by measuring the circumference of their wrists:

```
HT=Cdbl(Patient.GetValueRF
("0.Visit1.DEM.DEM.0.HEIGHT.PFHT_TC", "", 0, 0, 0))
WC=Cdbl(Patient.GetValueRF
("0.Visit1.DEM.DEM.0.WRISTCIRC.PFWC_TC", "", 0, 0, 0))
framesize="Unknown"
if (HT <> 0 and WC <> 0) Then
HTCM = HT * 2.54
WCCM = WC * 2.54
Radius = HTCM/WCCM
if (Patient.GetValueRF
("0.Visit1.DEM.DEM.0.GENDER.GENDERRADIO", "", 0, 0, 0) = 1)
Then
if Radius > 10.4 Then framesize = "Small Frame"
if (Radius >= 9.6 and Radius <= 10.4) Then
framesize = "Medium Frame"
if (Radius < 9.6 ) Then framesize = "Large Frame"
Else
if Radius > 11.0 Then framesize = "Small Frame"
if (Radius >= 10.1 and Radius <= 11.0) Then
framesize = "Medium Frame"
if (Radius < 10.1 ) Then framesize = "Large Frame"
End If
End If
Result.Result = framesize
Patient.SetValue "0.Visit1.DEM.DEM.0.FRAME.FRAME_CC", "",
0, 0, 0, framesize
```

Form rule with dynamic query text

The following form rule script checks whether a systolic component of a blood pressure measurement is less than the diastolic component. If the rule fails, the rule calls the AddNamedValue method to issue a query whose text overrides the default query text specified in the Event definition for the rule. This script uses the predefined QUERYTEXT and QUERYSTATE named value pairs:

```
MeasureItem1 = "0.BPREADINGGROUP.SYSTEXT"
MeasureItem2 = "0.BPREADINGGROUP.DIASTEXT"
Measure1=CInt(Patient.GetValueRF(MeasureItem1, "", 0, 0, 0))
Measure2=CInt(Patient.GetValueRF(MeasureItem2, "", 0, 0, 0))
if (Measure1<Measure2) Then
Result.RulePassed=0
    Patient.AddNamedValue "QUERYTEXT", "Systolic value
    must be higher than diastolic"
    Patient.AddNamedValue "QUERYSTATE", "Open"
Else
Result.RulePassed=1
End If
```

Rule with arguments

The following form rule script checks the value of three arguments—ItemName, Min, and Max. ItemName is a specific form control. Min and Max are range limits. This script can be attached to multiple items and tested with different controls and range-checking values passed in as arguments.

```
val = Patient.GetValueRF(Patient.GetArgument("ItemName"), "", 0,0,0)
If ( val > Patient.GetArgument("Min") AND val < Patient.GetArgument("Max") )
Then
Result.RulePassed=1
Else
Result.RulePassed=0
End If
```

Checking for date/time control completion

This example illustrates a way to check whether a date/time control is complete. The script uses the VALIDDATEMAP attribute of the GetValueRF method on the Patient object. VALIDDATEMAP returns an integer that contains a bitmask indicating which elements of the date/time control are complete:

YEAR	1
MONTH	2
DAY	4
HOUR	8
MINUTE	16
SECOND	32

```
strItem = "0.Visit1.DEM.DEM.0.DEMDOB"
PFYEAR=1
PFMONTH=2
PFDAY=4
PFCOMPLETEDATE = PFYEAR Or PFMONTH Or PFDAY
'PFCOMPLETEDATE=7
testDateMap = Patient.GetValueRF(strItem, "VALIDDATEMAP",
0, 0, 0)
If testDateMap=PFCOMPLETEDATE Then
Result.RulePassed = 1
Else
Result.RulePassed = 0
End If
```

Randomization rule scripts

Simple Central randomization

The following example illustrates a randomization rule for a Simple Central (Type 1) randomization scheme.

```
Randomization.SiteID = Patient.GetSiteID
Randomization.Type = 1
Randomization.Source = "SimpleCentral"
Randomization.PatientID = Patient.GetID
Randomization.Revision = 0
SeqNum = Randomization.GetNextKit(KitInfo)
Result.Result= SeqNum + " / " + KitInfo
```

Central Stratified randomization

The following example illustrates a randomization rule for a Central Stratified (Type 2) randomization scheme. In this example, the stratification is based on the patient's weight, as entered in the Demographics form in Visit 1. Based on the patient's weight, the InForm application gets the next sequence number from either the CS_WT150 or the CS_WT275 randomization source list.

```
Function GetRndSourceList()
wt = Patient.GetValueRF("0.Visit1.DEM.DEM.0.WEIGHT", "", 0,0,0)

IF (wt > 90 AND wt <= 150) THEN
  GetRndSourceList = "CS_WT150"
ELSEIF (wt > 150 AND wt < 275) THEN
  GetRndSourceList = "CS_WT275"
ELSE
  GetRndSourceList = ""
END IF
End Function

'set properties
Randomization.SiteID = Patient.GetSiteID
Randomization.Type = 2
Randomization.Source = GetRndSourceList
Randomization.PatientID = Patient.GetID
Randomization.Revision = 0
'randomize the patient and return result
SeqNum = Randomization.GetNextKit(KitInfo)
Result.Result= SeqNum + " / " + KitInfo
```

Simple Site randomization

The following example illustrates a randomization rule for a Simple Site (Type 3) randomization scheme. In this example, the patient's site determines the randomization source list from which the InForm application gets the next sequence number.

```
Function GetRndSourceList()
szSite = Patient.GetSiteName();

IF (szSite = "PhaseForward") THEN
  GetRndSourceList = "SS_PF"
ELSEIF (szSite = "Beth Israel") THEN
  GetRndSourceList = "SS_BID"
ELSE
  GetRndSourceList = ""
END IF
End Function

'set properties
Randomization.SiteID = Patient.GetSiteID
Randomization.Type = 3
Randomization.Source = GetRndSourceList
Randomization.PatientID = Patient.GetID
Randomization.Revision = 0
'run the randomization and return result
SeqNum = Randomization.GetNextKit(KitInfo)
Result.Result= SeqNum + " / " + KitInfo
```

Stratified by Site randomization

The following example illustrates a randomization rule for a Stratified by Site (Type 4) randomization scheme. In this example, the patient's site and height determine the randomization source list from which the InForm application gets the next sequence number. This example uses the Oracle randomization source lists SR_PF_HT45 and SR_PF_HT75 and the Beth Israel randomization source lists SR_BID_HT45 and SR_BID_HT75.

```
Function GetRndSourceList()
szSite = Patient.GetSiteName();
ht = Patient.GetValueRF("0.Visit1.DEM.DEM.0.HEIGHT", "", 0,0,0)
SELECT CASE szSite
CASE "PhaseForward"
IF (ht > 30 AND ht <= 45) THEN
    GetRndSourceList = "SR_PF_HT45"
ELSEIF (ht > 45 AND < 75) THEN
    GetRndSourceList = "SR_PF_HT75"
ELSE
    GetRndSourceList = ""
END IF

CASE "Beth Israel"
IF (ht > 30 AND ht <= 45) THEN
    GetRndSourceList = "SR_BID_HT45"
ELSEIF (ht > 45 AND < 75) THEN
    GetRndSourceList = "SR_BID_HT75"
ELSE
    GetRndSourceList = ""
END IF

CASE ELSE
    GetRndSourceList = ""
END SELECT
End Function

'setup randomization object properties
Randomization.SiteID = Patient.GetSiteID
Randomization.Type = 4
Randomization.Source = GetRndSourceList
Randomization.PatientID = Patient.GetID
Randomization.Revision = 0
'randomize the patient and return result
SeqNum = Randomization.GetNextKit(KitInfo)
Result.Result= SeqNum + " / " + KitInfo
```

Testing rules

The testing component of the InForm Architect application enables you to define test cases and to run rule scripts against those test cases. Default test cases run with the default arguments defined for the rule.

Additionally, you can test rules against actual patient data that has been submitted in the trial through the InForm application.

Creating test data

To create a test case:

- 1 Open the **Rule** window, and click the **Test Cases** tab.
- 2 In the **Expected Result** column do one of the following:
 - If the rule is a form rule, select **Pass** or **Fail** to indicate whether the rule should return a passing or failing value when it processes the accompanying test data.
 - If the rule is a calculation or conversion rule, enter the expected result of the calculation.
- 3 In the **Input Value 1** column, specify the value of the first argument referenced in the rule script. In subsequent **Input Value** columns, specify the values of each additional argument referenced by the rule script.
- 4 Repeat these steps to create as many as ten default test cases.
- 5 Save the rule with the **Install MedML When Saving** option on.

Note: The testing component of the InForm Architect application treats input values as strings. To test nonstring data, you must use VBScript casting functions in your test rule script to cast the input values appropriately. For example, if your script tests an item's input value against minimum and maximum ranges of integer values, cast each obtained value as an integer by using the CInt function:

```
val = CInt(Patient.GetValueRF (Patient.GetCurPath(), "", 0,0,0))
If ( val > CInt(Patient.GetArgument("Min")) AND
    val < CInt(Patient.GetArgument("Max")) ) Then
    Result.RulePassed=1
Else
    Result.RulePassed=0
End If
```


For a list of VBScript casting functions recognized by the InForm Architect application, see the VBS.ini file in the \bin directory of your InForm application installation.

Running rule scripts against test data

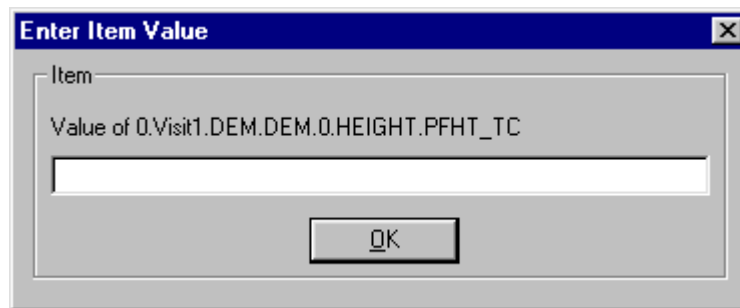
The InForm Architect application enables you to run rule scripts interactively or in batch mode.

Running interactively

To run a rule script interactively:

- 1 Open the **Rule** window.
- 2 In the **Rules** toolbar, click the **Run Script** () button.

In the **Get Item Value** dialog box, InForm Architect application prompts you to specify the value of the first argument specified in the script:



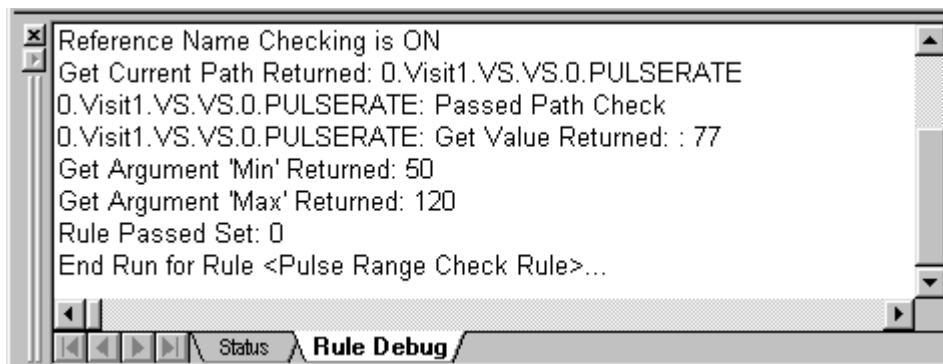
- 3 Specify the requested value, and click **OK**.

The InForm Architect application prompts you for the next value required by the script.

- 4 As each prompt appears in the **Get Item Value** dialog box, continue to provide values and click **OK**.

Note: It may be helpful to open the **Form** window or windows containing the data items referenced by the rule, so you can see which items correspond to the values prompted for in the **Get Item Value** dialog box.

As the required values are entered, the InForm Architect application runs the script against the specified values, and reports the progress and results in the **Rule Debug** tab of the **Output** window.



Running selected test cases

To run a rule script against selected test cases:

- 1 Open the **Rule** window.
- 2 Click the **Test Cases** tab.
- 3 Select the cases you want to test by clicking the appropriate numbered rows on the left. To select multiple adjacent cases, hold down the **Shift** key and click the first and last case you want. To select multiple nonadjacent cases, hold down the **Ctrl** key and click each case you want.


	Expected Result	Input Value 1	Input Value 2	In
1	Pass	0.Visit1.VS.VS.BPREADING.BPREADINGGROUP.SYSTEXT	120	80
2	Fail	0.Visit1.VS.VS.BPREADING.BPREADINGGROUP.SYSTEXT	79	80
3		0.Visit1.VS.VS.BPREADING.BPREADINGGROUP.SYSTEXT	301	80
4				
5				
6				
7				

- 4 In the **Rule** toolbar, click the **Run Selected Testcases** () button.

The InForm Architect application runs the script against the specified values, and reports the progress and results in the **Rule Debug** tab of the **Output** window. Cases with missing data are bypassed.

Running all test cases

To run a rule script against all test cases:

- 1 Open the **Rule** window.
- 2 In the **Rules** toolbar, click the **Run All Testcases** () button.

The InForm Architect application runs the script against the specified values, and reports the progress and results in the **Rule Debug** tab of the **Output** window. Cases with missing data are bypassed.

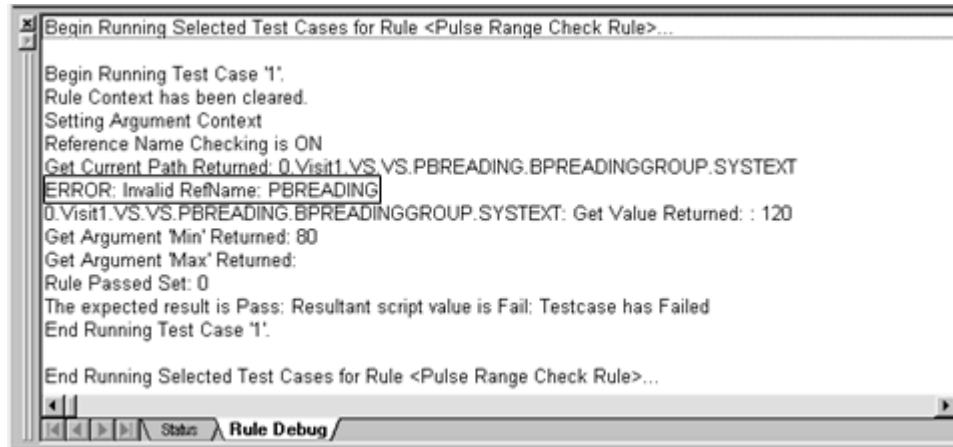
Using the RefName checker

The RefName checker is a tool that enables you to check the validity of RefName paths specified explicitly in a script or entered as arguments. When you run a script while the RefName checker is on, the InForm Architect application reports invalid RefName paths in the Output window.

To use the RefName checker:



- 1 Open the **Rule** window.
- 2 Click the **Toggle RefName Check** () button.
- 3 Set up and run test cases as described in *Running interactively* (on page 324).

The InForm Architect application runs the script against the specified values, and reports the progress and results in the **Rule Debug** tab of the **Output** window. Invalid RefNames are reported as errors.



Running rule scripts against actual patient data

To run a rule script against patient data submitted in the current trial:

- 1 Open the **Rule** window for the rule script you want to run.
- 2 In the **Rules** toolbar, make the following selections:
 - In the patient data list, select the patient against whose data you want to run the rule script.
 - Optionally, in the context list, select the context in which you want to apply the rule. The InForm Architect application uses the RefName paths and arguments specified for this context when processing the rule script.
- 3 Optionally, specify that you want the InForm Architect application to check RefNames by clicking the **Toggle RefName Check** () button in the **Rules** toolbar.
- 4 In the **Rules** toolbar, click the **Run Script** () button.

The InForm Architect application runs the script against the submitted values for the selected patient, and reports the progress and results in the **Rule Debug** tab of the **Output** window. If you specified that you want RefName checking to be performed, invalid RefNames are reported as errors.

Note: If you run a rule script containing the `GetCurrentISRowNum` method, that method always returns 0.

Debugging rule scripts

As described in the previous sections, the InForm Architect application supports debugging rule scripts by presenting messages in the Rule Debug tab of the Output window when you run scripts against test data. Specifically, the Output window displays:

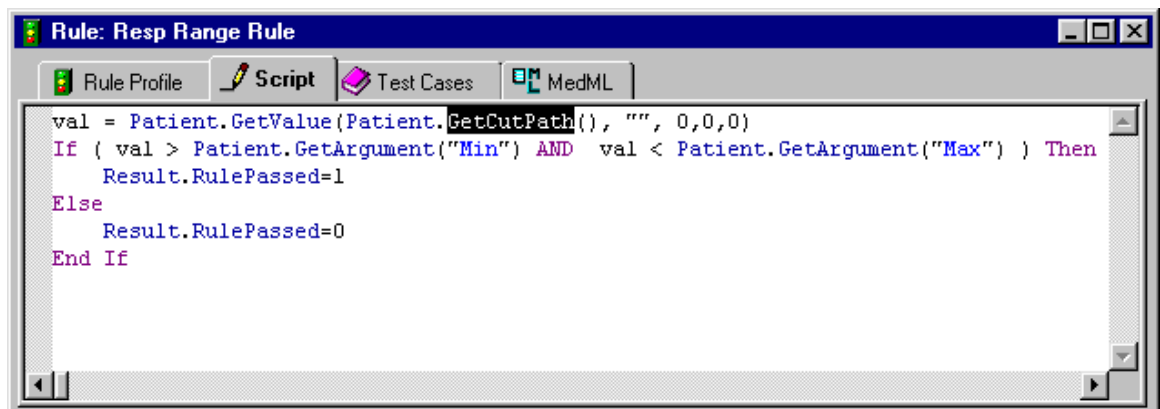
- VBScript syntax errors.
- Invalid RefName paths, if the RefName checker is on. For information on the RefName checker, see *Using the RefName checker* (on page 325).
- Values of arguments defined for the selected context.
- Values of test variables.
- Rule result:
 - If the rule is a form rule, the output indicates whether the rule would pass or fail with the submitted test values.
 - If the rule is a calculation, the output indicates the result of the calculation.
- Trace values specified with the Patient.OutputDebug method, as described in *Using the Patient.OutputDebug method* (on page 329).

Rule script debugging examples

This section provides examples of the types of output you can receive from the rule debugger.

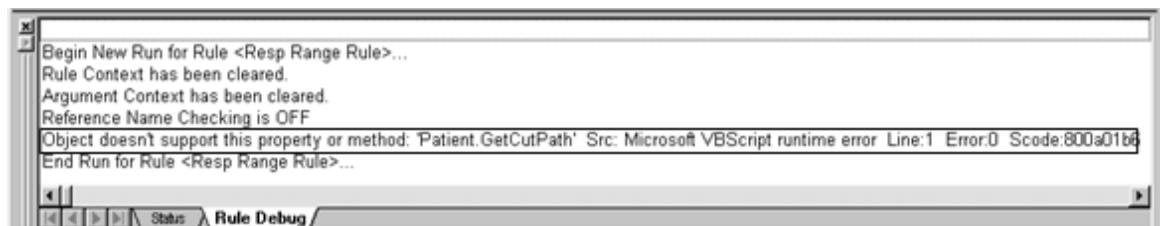
Syntax error

The following script has a typographical error in the first line. It should reference the GetCurPath method of the Patient object. The rule debugger reports this mistake as a syntax error.



```

Rule: Resp Range Rule
Script
val = Patient.GetValue(Patient.GetCutPath(), "", 0,0,0)
If ( val > Patient.GetArgument("Min") AND val < Patient.GetArgument("Max") ) Then
    Result.RulePassed=1
Else
    Result.RulePassed=0
End If
  
```



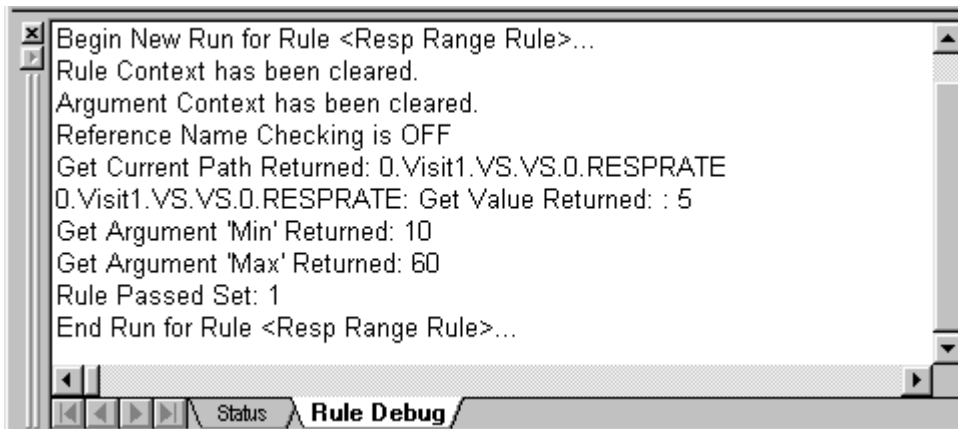
```

Begin New Run for Rule <Resp Range Rule>...
Rule Context has been cleared.
Argument Context has been cleared.
Reference Name Checking is OFF
Object doesn't support this property or method: 'Patient.GetCutPath' Src: Microsoft VBScript runtime error Line:1 Error:0 Scode:800a01b8
End Run for Rule <Resp Range Rule>...
  
```

Test argument and variable values

When you run the following script against a context with Min and Max arguments defined as 10 and 60, respectively, and enter a respiration rate of 5, the debugger displays the defined argument values and the entered test value. As the rule would fail with this test value, the output reports the rule failure.

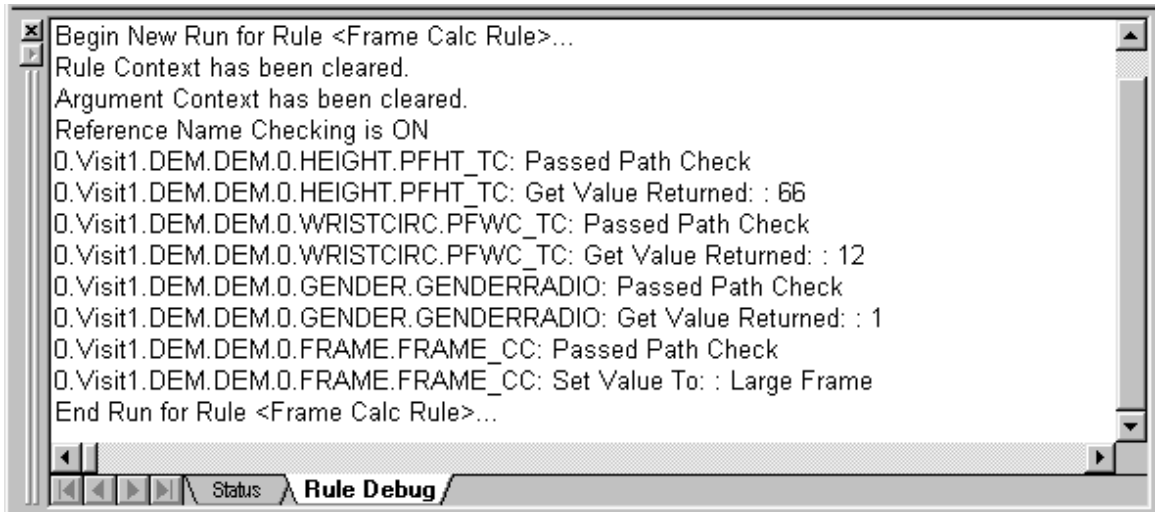
```
val = Patient.GetValueRF(Patient.GetCurPath(), "", 0,0,0)
If ( val > Patient.GetArgument("Min") AND
    val < Patient.GetArgument("Max") ) Then
Result.RulePassed=1
Else
Result.RulePassed=0
End If
```



RefName checker

The following script illustrates a calculation that requires input of three values for which the RefName path is specified in the script. When you run the rule with the RefName checker on, the output shows that the rule debugger has checked each RefName path. Additionally, because this is a calculation rule, the rule debugger displays the result of the calculation.

```
HT = Cdbl(Patient.GetValueRF("0.Visit1.DEM.DEM.0.HEIGHT.PFHT_TC", "", 0,0,0))
WC = Cdbl(Patient.GetValueRF("0.Visit1.DEM.DEM.0.WRISTCIRC.PFWC_TC", "", 0,0,0))
framesize = "Unknown"
if (HT <> 0 and WC <> 0) Then
HTCM = HT * 2.54
WCCM = WC * 2.54
Radius = HTCM/WCCM
if (Patient.GetValueRF("0.Visit1.DEM.DEM.0.GENDER.GENDERRADIO", "", 0,0,0) = 1)
Then
if Radius > 10.4 Then framesize = "Small Frame"
if (Radius >= 9.6 and Radius <= 10.4) Then framesize = "Medium Frame"
if (Radius < 9.6 ) Then framesize = "Large Frame"
Else
if Radius > 11.0 Then framesize = "Small Frame"
if (Radius >= 10.1 and Radius <= 11.0) Then framesize = "Medium Frame"
if (Radius < 10.1 ) Then framesize = "Large Frame"
End If
End If
Result.Result = framesize
Patient.SetValue "0.Visit1.DEM.DEM.0.FRAME.FRAME_CC", "", 0, 0, 0, framesize
```



Using the Patient.OutputDebug method

The InForm Architect application enables you to specify trace values in a rule script by using the OutputDebug method of the Patient object. When you use this method, the rule debugger reports the value of test variables that you define.

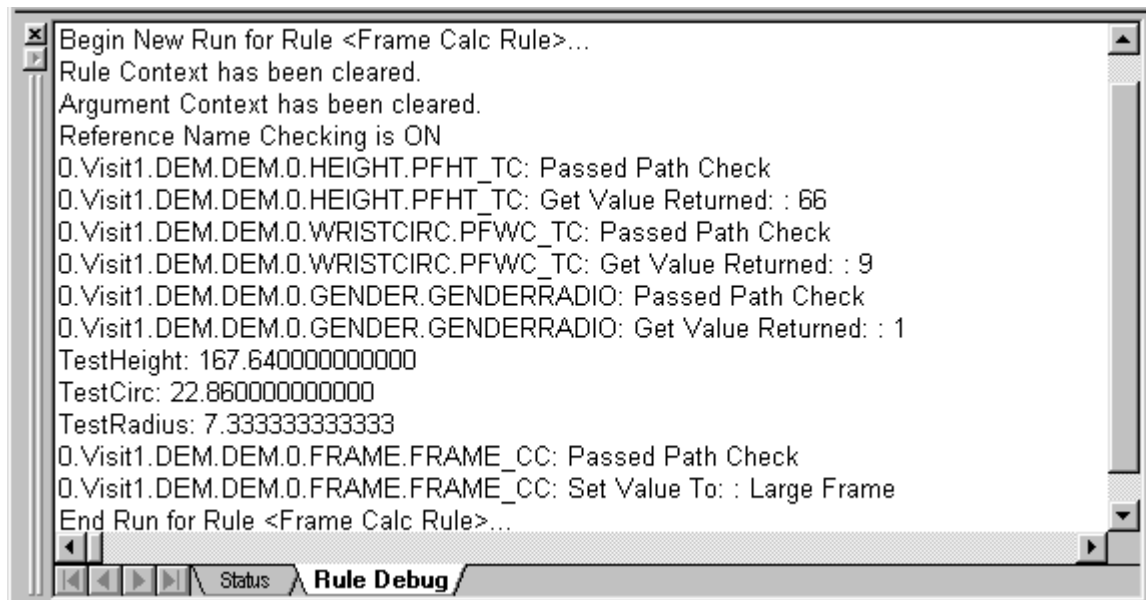
The OutputDebug method has the following arguments:

- **Print string**—String that identifies the value you want to trace.
- **Variable**—Name of the variable for which you want the rule debugger to display the value.

To use the method, insert it in your rule script at the point where you want the rule debugger to display the specified variable value.

The following example shows a portion of the frame size script illustrated in *RefName checker* (on page 328), with trace variables inserted. The resulting output shows the values of each variable.

```
if (Patient.GetValueRF("0.Visit1.DEM.DEM.0.GENDER.GENDERRADIO", "", 0, 0, 0) = 1)
Then
Patient.OutputDebug "TestHeight", HTCMT
Patient.OutputDebug "TestCirc", WCCM
Patient.OutputDebug "TestRadius", Radius
if Radius > 10.4 Then framesize = "Small Frame"
if (Radius >= 9.6 and Radius <= 10.4) Then framesize = "Medium Frame"
if (Radius < 9.6 ) Then framesize = "Large Frame"
```



CHAPTER 8

Defining events and execution plans

In this chapter

About events.....	332
Creating events	333
Creating an execution plan	335
Execution plan objects and methods.....	337
Debugging execution plans	338

About events

Events enable the InForm application to use the results returned by a rule script to generate any of the following actions:

- Issuing or closing a candidate or open query against an item on a form.
- Sending email, or by using an email gateway, sending a fax or activating a beeper.
- Creating an entry in the Windows log file.

For a description of the rule and event processing flow that occurs when an item is submitted or updated, see [Rules and events](#).

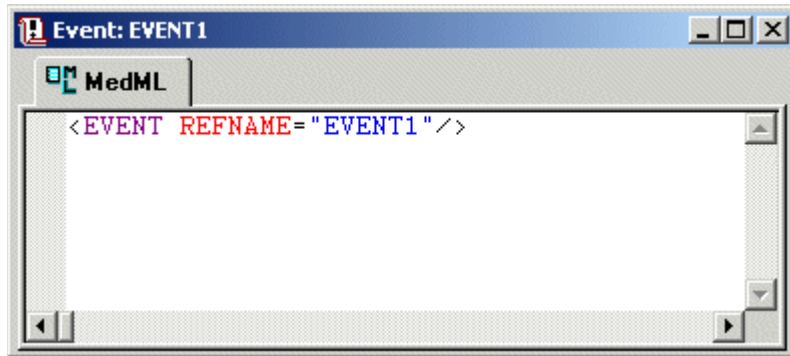
In addition to events that fire during rule processing, the InForm application includes predefined system events that fire when state information changes; for example, when a CRF is frozen or unfrozen. This chapter describes the events associated with rules.

Creating events

This section describes how to create an event and associate it with a rule definition.

Creating a new event

To create a new event, select **File > New > Event**. The **Event** window opens in the Trial Design workspace. This window consists of the **MedML** tab, which displays the MedML tags that define the event.

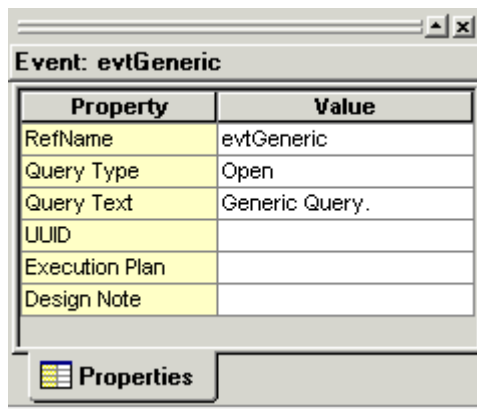


Specifying event properties

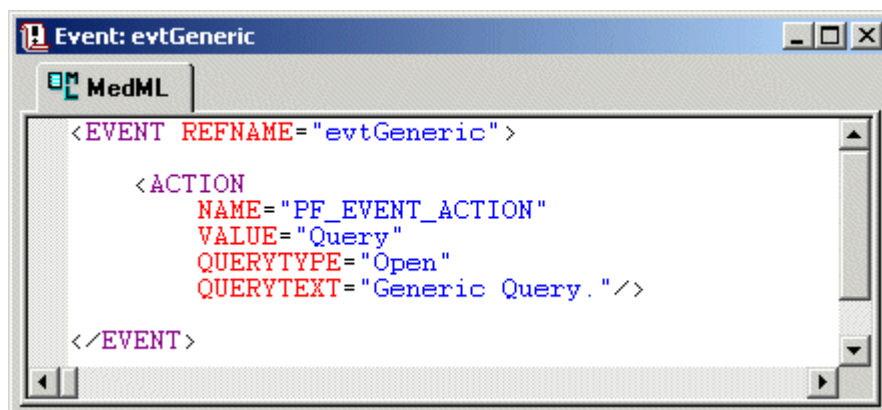
When you open a new Event window, the Properties window displays the properties for defining an event. The following table specifies event properties:

Property	Description
RefName	RefName of the component. REQUIRED. For rules about the use of RefNames, see <i>RefNames</i> (on page 89)
Query Type	Type of query to generate when the event fires: Candidate or Open. Candidate queries are invisible to sites and must be reviewed and explicitly opened. Open queries are immediately visible to sites and available for response. Both Query Type and Query Text must be present for InForm Architect application to generate MedML for the event description.
Query Text	Specify the default text of the query that the event generates. If you specify dynamic query text in the definition of the rule with which the event is associated, the InForm application overrides the default text specified as the event property. For information on generating dynamic query text, see <i>Form rule with dynamic query text</i> (on page 319). Both Query Type and Query Text must be present for InForm Architect application to generate MedML for the event description.

Property	Description
UUID	String that uniquely identifies the event within a trial. It may also be unique across all databases, trials, and machines, depending upon how you set up your trials. OPTIONAL. Note that the InForm Architect application automatically capitalizes UUID strings that contain lower-case characters.
Execution Plan	RefName of the execution plan that the event fires, if any. OPTIONAL.
Design Note	Free-form text, with a maximum of 255 characters, containing any information you want to capture about the design of the component. This information is for documentation only. OPTIONAL.



As you specify event properties, InForm Architect application updates the Event window with the appropriate MedML tags.



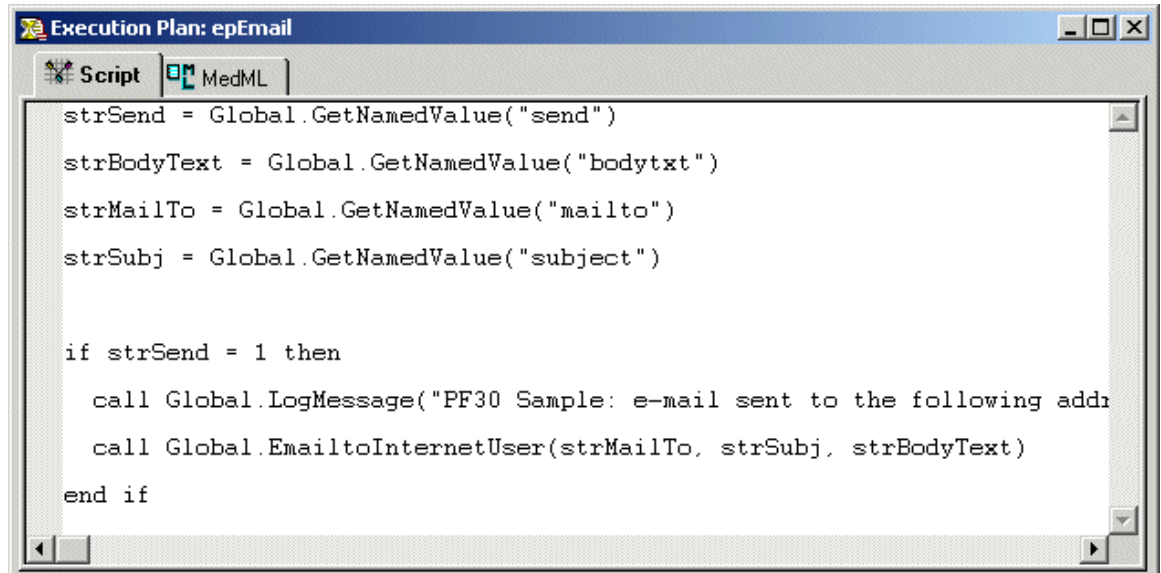
Associating an existing event with a rule

To associate an existing event with a rule, open the rule definition and select the event in the **Properties** window. For details, see *Specifying rule properties* (on page 294).

Creating an execution plan

To create a new execution plan:

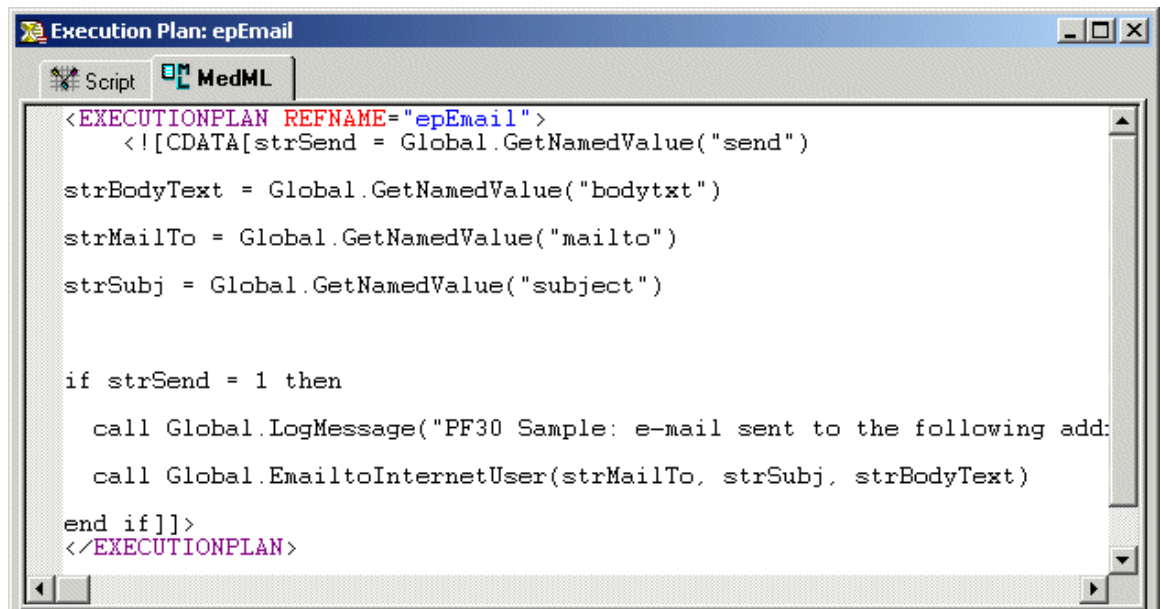
1. Select **File > New > Execution Plan**. The **Execution Plan** window opens in the Trial Design workspace. This window consists of the following tabs:
 - **Script** — Editor window that enables you to create the script that runs when the associated event fires.
 - **MedML** — Read-only tab that displays the MedML tags that define the execution plan.



```

Execution Plan: epEmail
Script
strSend = Global.GetNamedValue("send")
strBodyText = Global.GetNamedValue("bodytxt")
strMailTo = Global.GetNamedValue("mailto")
strSubj = Global.GetNamedValue("subject")

if strSend = 1 then
  call Global.LogMessage("PF30 Sample: e-mail sent to the following address: " & strMailTo)
  call Global.EmailtoInternetUser(strMailTo, strSubj, strBodyText)
end if
  
```



```

Execution Plan: epEmail
MedML
<EXECUTIONPLAN REFNAME="epEmail">
  <![CDATA[strSend = Global.GetNamedValue("send")
strBodyText = Global.GetNamedValue("bodytxt")
strMailTo = Global.GetNamedValue("mailto")
strSubj = Global.GetNamedValue("subject")

if strSend = 1 then
  call Global.LogMessage("PF30 Sample: e-mail sent to the following address: " & strMailTo)
  call Global.EmailtoInternetUser(strMailTo, strSubj, strBodyText)
end if]]>
</EXECUTIONPLAN>
  
```

- 2 In the **Script** window, create a script, using VBScript. You can use an execution plan script to accomplish any of the following:

- Log an event to the Windows event log.
- Pass a context string from a rule.
- Send email.

When creating an execution plan script, you can use the objects described in the MedML online help. The examples shown in *Execution plan objects and methods* (on page 337) illustrate scripts for each of the activities you can perform with execution plans.

- 3 In the **Properties** window, specify the RefName of the execution plan in the **Reference Name** row.
- 4 Optionally, specify a **Design Note** describing the execution plan for documentation purposes.

Execution plan objects and methods

When you create an execution plan script, you can use the objects and methods described in this section.

Execution plans have access to several objects and their methods. You use these objects and methods to retrieve entered values and to hold values passed from the Patient or Result rule objects. For information on the execution plan objects and methods that are available with the InForm application, see the MedML online Help.

Examples of scripts that use execution plan objects

Sending email to an InForm application user

```
Global.EMailToInformUser "sm", "CRFUnlock",
"Sending InForm User email when Unlockl-CRF"
```

Sending email to an internet user

```
Global.EMailToInternetUser "user@phaseforward.com", "CRFUnlock", "Sending
external User email when Unlockl-CRF"
```

Getting a set of named values from the Patient object

```
On error resume next
aeserval_m = Global.GetNamedValue("aeserval")
if aeserval_m = "Y" then
ptinit_m = Global.GetNamedValue("ptinit")
aedate_m = Global.GetNamedValue("aedate")
aeevent_m = Global.GetNamedValue("aeevent")
site_m = Global.GetNamedValue("sitenm")
emailtxt="An SAE has occurred at Site: "&site_m&" for Patient: "&ptinit_m &" on
AE Date: "&aedate_m&". The event is : "&aeevent_m&".
call Global.EmailToInternetUser "test@phaseforward.com", "SAE
Occurrence",emailtxt
end if
```

Sending a message to the Windows log

```
Global.LogMessage "Query generated"
```

Sending an email when a User is deactivated

This script, named **EmailInactiveUser.vbs** will send an email when a User is deactivated:

```
<?xml version="1.0" ?>
- <MEDMLDATA xmlns="PhaseForward-MedML-Inform4">
<EXECUTIONPLAN REFNAME="EMAILINACTIVE" SCRIPTTYPE="SERVERRULE"
SCRIPTFILE="EmailInactiveUser.vbs" />
- <EVENT REFNAME="DeActivate User" UUID="PF_EVENT_UUID_DEACTIVATE_USER">
<EXECUTIONPLANREF REFNAME="EMAILINACTIVE" />
</EVENT>
</MEDMLDATA>
```

This is the execution plan with the refname of EMAILINACTIVE:

```
Dim strMsg
strMsg = "User: " & Global.GetNamedValue("PF_EVENT_USERNAME")
strMsg = strMsg & " has been inactivated."
Global.EMailToInternetUser "user@phaseforward.com", strMsg, strMsg
```

Debugging execution plans

The InForm Architect application supports debugging of execution plan scripts in the same way that it supports debugging of rule scripts—by presenting messages in the Rule Debug tab of the Output window when you run the scripts. Specifically, the Output window displays:

- VBScript syntax errors.
- Trace values specified with the `Global.OutputDebug` method, as described in *Using the `Global.OutputDebug` method* (on page 338).

To test an execution plan script:

- 1 Open the **Execution Plan** window containing the script.
- 2 Click the **Run Script** button in the **Rules** toolbar.

The **Output** window displays messages that indicate the processing of the execution plan script.

Using the `Global.OutputDebug` method

The InForm Architect application enables you to specify trace values in a rule script by using the `OutputDebug` method of the `Global` object. When you use this method, the rule debugger reports the value of test variables that you define.

The `OutputDebug` method has the following arguments:

- **Print string**—String that identifies the value you want to trace.
- **Variable**—Name of the variable for which you want the rule debugger to display the value.

To use the method, insert it in your rule script at the point where you want the rule debugger to display the specified variable value.

CHAPTER 9

Enabling specialized InForm application features

In this chapter

Overview	340
Alternate versions of forms.....	341
Associations	343
Coding dictionaries	346
Common forms.....	348
Links to CRF Help	349
Date display format	352
Dynamic forms.....	353
Dynamic visits	355
Monitoring forms.....	357
Patient navigation mode	359
Patient ID information	360
Patient record transfer affidavit.....	364
Randomization	366
Repeating forms	377
Reports.....	378
Screening and enrollment forms	381
Forms requiring signature.....	384
Study Completion form	393
Unscheduled or repeating visits.....	401
UUIDs	402

Overview

This section describes:

- How to implement specialized functionality in the InForm application by using the InForm Architect application and MedML.
- Specific UUIDs that are required for specialized InForm application features.

Alternate versions of forms

From time to time during a trial, the contents of a form may change to accommodate a protocol revision or a need for clarification. In general, if a patient has started a CRF when a new CRF version is implemented, the original CRF is retained. Any patients who have not started the CRF use the new form. This is a logical approach when the data collected on a CRF depend on measurements and observations taken at a specific visit.

However, there may be occasions when it is desirable to collect additional data retrospectively, when the data are not specifically visit dependent. For example, the sponsor may decide to collect additional family history data after a patient has completed the intake visit where that data might most logically be recorded. In such a case, you can create an alternate version of a form for the additional data to be collected, and include the form in the visit where the data would have been collected originally. For patients who have started the original version of the form, both the original and the alternate form are presented, and the new data can be collected on the alternate version of the form.

Implementing an alternate form

To implement an alternate version of a form:

- 1 Use the appropriate form definition components to create, save, and install a form that includes the new items to collect.

Note: You can do this step by using the InForm Architect application. For the remaining steps, you must manually create an XML file that defines a revised trial definition.

- 2 Include the alternate form in a StudyVersion definition by using the ALTREFNAME attribute of the FORMREF tag that associated the original form definition with a VISIT FormSet.
- 3 Update the StudyVersion by changing the VERSIONDESCRIPTION attribute.
- 4 Load the new StudyVersion with the MedML Installer tool.

Example of implementing an alternate form

In this example, the original StudyVersion definition includes a VISIT FORMSET that contains the DEM form. In the updated StudyVersion FORMSET tag, the FORMREF tag that includes the DEM form has an ALTREFNAME attribute identifying the RefName of the alternate form, **ALTD**EM.

Original:

```
<STUDYVERSION VERSION="1" VERSIONDESCRIPTION="Demo Study"
  STUDYNAME="Hypertension Study" PROTOCOL="PF-105">
  </FORMSET>
  <FORMSET REFNAME="Visit1" TITLE="Baseline"
    MNEMONIC="Baseline"
    UUID="03B0D5D8-7F2C-11D2-A728-00A0C977C64B"
    LANGUAGE="English" TYPE="Visit"
    STARTHOUR="0" SCHEDULED="TRUE" ORDER="1">
    <FORMREF REFNAME="patientidentifier"/>
    <FORMREF REFNAME="DEM"/>
  </FORMSET>
</STUDYVERSION>
```

Updated:

```
<STUDYVERSION VERSION="2" VERSIONDESCRIPTION="Demo Study2"
  STUDYNAME="Hypertension Study" PROTOCOL="PF-105">
  </FORMSET>
  <FORMSET REFNAME="Visit1" TITLE="Baseline"
    MNEMONIC="Baseline"
    UUID="03B0D5D8-7F2C-11D2-A728-00A0C977C64B"
    LANGUAGE="English" TYPE="Visit"
    STARTHOUR="0" SCHEDULED="TRUE" ORDER="1">
    <FORMREF REFNAME="patientidentifier"/>
    <FORMREF REFNAME="DEM" ALTREFNAME="ALTD
```

EM"/>
 </FORMSET>
</STUDYVERSION>

Associations

Form associations provide cross-references and simplified navigation between instances of forms containing data that users often refer to together. In this way, when users display the details of one associated form, they can easily view the details of any instances of the other associated form.

Overview

To associate forms:

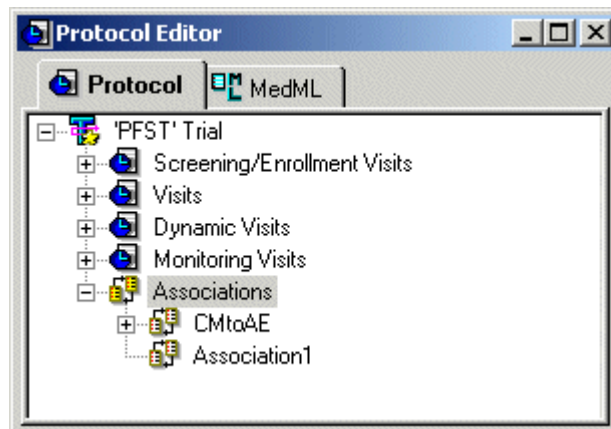
- 1 Create two repeating forms that you want to be in an association with each other. For more information, see *Repeating forms* (on page 377).
- 2 *Define an association* (on page 343) and include the two repeating forms in it.

Defining an association

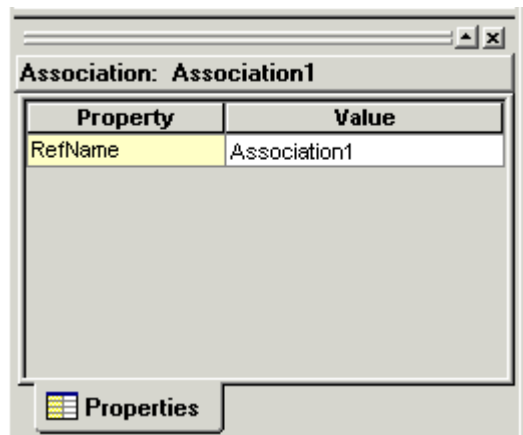
To define an association:

- 1 Open the **Protocol Editor** window.
- 2 Right-click the **Associations** icon and select **Add Association** from the pop-up menu.

The InForm Architect application adds a new association to the **Associations** node.



- 3 Select the new association and edit its **RefName** in the **Properties** window.



Including repeating forms in an association

To include two forms in an association, do one of the following:

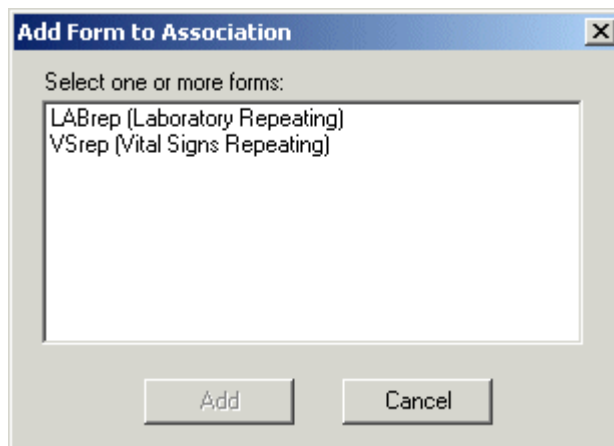
- *Right-click the association* (on page 344) in the **Properties** window.
- *Drag the forms* (on page 345) from the **Trial Objects** window onto the association in the **Properties** window.

Right-clicking the association

To add repeating forms to an association by right-clicking:

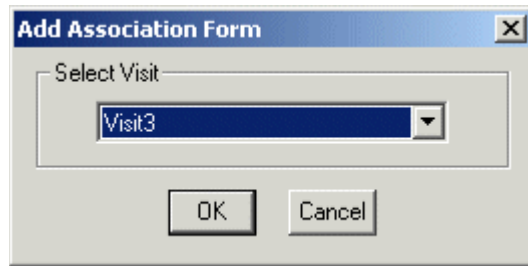
- 1 Right-click the association and select **Add Form** from the popup menu.

The **Add Form to Association** dialog box opens and displays the repeating forms that have been defined for the trial.



- 2 Select one or both of the forms that you want to include in the association.

The **Add Association Form** dialog box opens.



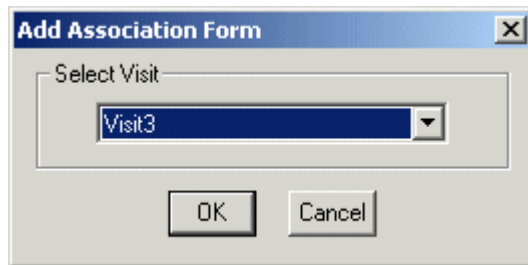
- 3 Select the visit for the first form in the association, and click **OK**.
- 4 Do one of the following:
 - If you selected both forms in the **Add Form to Association** dialog box, the **Add Association Form** dialog box reappears. Select the visit for the second form in the association and click **OK**.
 - If you selected only one form in the **Add Form to Association** dialog box, repeat the entire procedure to add the second form to the association.

Dragging and dropping forms

To add repeating forms to an association by dragging and dropping forms:

- 1 From the **Trial Objects** window, select, drag, and drop a desired form into the new form relationship.

The **Add Association Form** dialog box opens.



- 2 Select the visit for the form, and click **OK**.

Note: Once you have set up an association and installed it in a study version, you cannot change or remove it.

Coding dictionaries

When used with the InForm Adapter and Central Coding, the InForm application supports automated coding, using the following standard coding dictionaries:

- **Medical Dictionary for Drug Regulatory Activities**—Standard thesaurus containing terminology applicable to all phases of drug development and to the health effects of devices.
- **World Health Organization Drug Dictionary**—Standard thesaurus containing terminology about drugs.

To use Central Coding with an InForm application trial, you must install the XML definitions of the applicable standard dictionaries in the trial database by using the MedML Installer. The InForm application includes an XML file containing sample definitions of the MedDRA and WHODD dictionaries. This file, called `core_Dicts.xml`, is located in the XMLBase folder of the InForm application installation.

Coding dictionary XML

The following table lists the MedML components that identify a coding dictionary:

MedML component	Description	Attributes
DICTIONARY	Identifies the coding dictionary. Each DICTIONARY tag must have at least one CODETARGET child element and can have one or more CONTEXTITEM and VERBATIMTYPE child elements	<ul style="list-style-type: none"> • TYPE—Type of dictionary, for example, MedDRA or WHODD. • VERSION—Dictionary version, for example, 8.1, 05Q4. • CULTURE—Language and culture, for example, en-US. <p>All are REQUIRED. and together uniquely identify a dictionary.</p>
CODETARGET	Identifies a specific code target defined in the dictionary.	<p>NAME—Name of the code target, for example, ATC 1.CODE.</p> <p>REQUIRED..</p>
CONTEXTITEM	Identifies a specific context item defined in the dictionary.	<p>NAME—Name of the context item, for example, Route Of Administration, Indication.</p> <p>REQUIRED..</p>
VERBATIMTYPE	Identifies a specific type of verbatim defined in the dictionary. VERBATIMTYPE corresponds to item type in Central Coding.	<p>NAME—Type of verbatim item, for example, AE, DISEASE, LABDATA, MEDPROD. AE, DISEASE, and LABDATA are valid verbatim types for the MedDRA dictionary. MEDPROD is a valid verbatim type for the WHODD dictionary.</p> <p>REQUIRED..</p>

Dictionary XML restrictions

When you install a new or updated definition of a coding dictionary in the InForm application database, the MedML Installer checks that:

- Each target, context item, and verbatim type name is unique within the dictionary.
- At least one target is defined.

Dictionary XML updates

When you install a valid updated definition of a coding dictionary in the InForm application database, the InForm application:

- Creates a new revision of the dictionary definition if a dictionary with the same the type, version, and culture exists in the database.
- Checks whether any code targets or context items in the current dictionary definition do not exist in the new definition. If so:
 - The InForm application checks all coding mappings for references to the obsolete code targets or context items and removes code target and context item control paths that are no longer part of the dictionary.
 - For each removed code target or context item control path, the MedML Installer displays a warning indicating the removal.

Common forms

Regular forms contain data for a specific visit. Even if the same form appears in another visit, the form starts over in the new visit and data do not carry forward from the previous visit in which the form occurs.

Common forms are CRFs that contain cumulative data; for example, a Concomitant Medication or Adverse Event form. When a common form appears in a subsequent visit, it contains the data from all visits in which the form occurs.

Note: Be careful not to include the Date of Visit field in a common form when creating trial definition data.

Defining a common form

To define a common CRF:

- 1 Create a form definition for capturing cumulative data by using the InForm Architect application. The following formats work well for common CRFs:
 - **Itemset format**, in which data are displayed in rows consisting of associated, repeated items.
 - **Repeating form format**, in which multiple instances of the same data points can be collected in the same form in the same visit.
- 2 In the form definition, specify **True** as the value of the **Common Form** property.

When entry is not required

When you create a form definition for a common form in which entry is not required, either of the following design or procedural steps is necessary to ensure that the InForm application correctly records the form as completed. Use either of the following methods:

- **Define a regular section** in addition to the repeating section for the itemset if the form contains an itemset. In the regular section, include a required item that asks whether there is any data for the itemset.
- **Create a standard operating procedure** that requires sites to mark an optional common form that will not be completed as Not Applicable. To do this, a user defines a form-level comment and indicates why the form is being left blank. This process is described in the InForm application online Help.

The bookmark is followed by the text of the CRF Help item description:

```
<ul>
<li><a name="Item1">Male or Female must be checked.</a>
</li>
</ul>
```

This bookmark and item description appear online in the Document window as follows:

Item 1 Gender

- Male or Female must be checked.

Defining a HelpLink component

To define a HelpLink component:

- 1 Use the <HELPLINK/> tag.
- 2 Use the DOCREFNAME tag to specify the RefName of the Document component representing CRF Help.
- 3 Use the BODYREFNAME tag to specify the RefName of the DocBody component that represents the CRF Help file where the bookmark occurs.
- 4 Optionally, use the BOOKMARK tag to specify the bookmark you want displayed at the top of the window when a user clicks a CRF link in an item. If you do not specify a BOOKMARK tag, the CRF Help file opens to the top of the file, and users must scroll to the specific item definitions they want.

Including a HelpLink component in a CRF Help document

You do not define a HelpLink as a stand-alone component. Instead, define a HelpLink and include it in an item or form definition in one step.

Example of a HelpLink to an CRF item help topic

The following tags define an item and the HelpLink that points to the bookmark labeled **Item1**:

```
<ITEM REFNAME="GENDER" QUESTION="Gender: ">
  <CONTROLREF REFNAME="GENDERRADIO" />
  <HELPLINK DOCREFNAME="Study" BODYREFNAME="DEMHELP"
    BOOKMARK="Item1" />
</ITEM>
```

1.	<u>Gender:</u>	<input type="radio"/> Male <input type="radio"/> Female	
----	----------------	--	---

Example of a HelpLink to a CRF help topic

The following tags define a CRF and the HelpLink that points to the CRF Help topic about the CRF:

```
<FORM REFNAME="DEM" TITLE="Demographics" MNEMONIC="DEM"  
  QUESTIONWIDTH="35">  
  <SECTIONREF REFNAME="DEM" />  
  <SECTIONREF REFNAME="SH" />  
  <HELPLINK DOCREFNAME="Study" BODYREFNAME="DEMHELP" />  
</FORM>
```

Date display format

The InForm application enables you to customize the form in which dates are displayed in the user interface. The following date formats are available:

- Month/Day/Year
- Day/Month/Year
- Year/Month/Day

You can specify the date format at the trial, site, or user level. The date format specified for a site overrides the format specified for the trial, and the date format specified for a user overrides the formats specified for the trial and the site.

If a user is assigned to more than one site, the date format defaults to the assigned trial date format.

The following table shows where to specify the date format for the trial, site, and user:

Level	Where to specify formatting
Trial	Specify system configuration option on the System Configuration screen in the InForm application.
Site	Update the Date Format item on the Site screen by using the InForm application or the InForm Architect application.
User	Update the Date Format item on the User screen by using the InForm application or the InForm Architect application.

Dynamic forms

Dynamic forms appear conditionally in a visit based on the result of a calculation defined on another form. For example, if you want to collect additional data from a female patient who is pregnant, you can define a dynamic form that appears in the trial only if the response to a question determining whether the patient is pregnant is **yes**.

Overview - Implementing dynamic forms

To implement a dynamic form:

- 1 *Define the form and add it to the visit* (on page 353) where it will appear.
- 2 *Create a calculation* (on page 353) that tests the data item on which the appearance of the form depends. Attach the calculation to that data item.

Note: To do these tasks you can use the InForm Architect application.

Creating a dynamic form in a visit

To create a dynamic form definition in a specific visit:

- 1 Create the form that will appear in the visit.
- 2 Add the form to the visit in the **Protocol Editor**.
- 3 In the **Protocol Editor**, select the form.
- 4 In the **Properties** window that is visible when you select the form in the **Protocol Editor**, set the value of the **Dynamic** property to **True**.

Note: The dynamic aspect of a form is a property of association with a visit, not of the form itself. Therefore, you can use the same form as a regular CRF in one visit and a dynamic CRF in another.

Creating a dynamic form calculation

To make a dynamic form display in a trial, create a calculation rule script that tests the value of the item on which the form appearance is based. The script uses the ShowDynamicForm method on the Patient or Result object.

ShowDynamicForm has the following parameters:

- **RefName** of the visit in which to schedule the appearance of the form
- **RefName** of the form to schedule
- **Visit Index**. This is a number that specifies an index into a repeating visit. Values are:
 - **0**—Indicates that the visit is not repeating or, if the visit is repeating, references the current instance of the visit.
 - **Any number greater than 0**—Explicitly indexes the specified occurrence of the visit.

Example of dynamic form calculation rule

In this example, the patient's gender determines whether a pregnancy test form appears in the visit.

```
MALE=1
FEMALE=2

Gender=Patient.getvalueRF("0.visit1.DEM.DEM.0.GENDER", "", 0, 0, 0)
If Gender=FEMALE Then
    Patient.ShowDynamicForm "visit1" , "Preg", 0
Else
    Patient.RemoveNotStartedDynamicForm "visit1", "Preg", 0
End If
```

The rule is attached to the GENDER item on the DEM form as a Special Visit context.

Removing a dynamic form

If no data has been entered into a dynamic form, you can remove it from the patient's Case Book by using the `RemoveNotStartedDynamicForm` method on the Patient or Result object.

`RemoveNotStartedDynamicForm` has the following parameters:

- **RefName** of the visit in which to the form appears or may appear.
- **RefName** of the dynamic form to remove from the schedule.
- **Visit Index.** This is a number that specifies an index into a repeating visit. Values are:
 - **0**—Indicates that the visit is not repeating or, if the visit is repeating, references the current instance of the visit.
 - **Any number greater than 0**—Explicitly indexes the specified occurrence of the visit.

Dynamic visits

Dynamic visits enable you to dynamically schedule a set of visits for a patient based on the value of a data item. For example, if you want to create a different sequence of visits for patients who are randomly assigned to receive different dosages of a drug in the third visit of a trial, you can do this by setting up dynamic visits.

Overview - Implementing dynamic visits

To implement a dynamic visit:

- 1 **Create a dynamic visit definition** (on page 355) and include it in the study version of the trial.
- 2 **Create a rule that tests the data item** (on page 355) on which the choice of visits is determined and assigns the patient to the appropriate dynamic visit.

You can do these tasks by using the InForm Architect application.

Creating a dynamic visit definition

To create a dynamic visit definition:

- 1 Create a new visit in the **Dynamic** node of the **Protocol Editor**.
- 2 Set the value of the **Dynamic** property to **True**.

Creating a rule for selecting a dynamic visit schedule

To assign a patient to a dynamic visit schedule, create a calculation rule script that tests the value of the item on which the visit assignment is based and makes the assignment. Use the `SetDynamicVisitStart` method on the `Patient` or `Result` object.

`SetDynamicVisitStart` has the following parameters:

- **RefName of the dynamic visit** to schedule.
- **RefName of the visit preceding** the dynamic visit.
- **Number of hours** before the dynamic visit starts. If you specify a preceding visit `RefName`, this is the number of hours from that visit. If you do not specify a preceding visit `RefName`, this is the number of hours from the beginning of the trial.

Example of a rule for a dynamic visit

In this example, the patient's gender determines whether the patient is scheduled for the `Dose100` or the `Dose250` visit, both of which start 720 hours after the patient's first visit.

```
MALE=1
FEMALE=2
Gender = Patient.GetValueRF("0.Visit1.DEM.DEM.0.GENDER", "", 0, 0, 0)

If Gender = MALE Then
    DVisit="Dose100"
Else
    DVisit="Dose250"
End If
```

```
Patient.SetDynamicVisitStart DVisit, "", 720
```

Example of a rule for a dynamic visit date

In this example the patient's gender determines whether the patient starts the Visit4Gen dynamic visit ten days or two weeks after the previous visit, Visit3.

```
MALE=1
FEMALE=2
Gender = Patient.GetValueRF("0.Visit1.DEM.DEM.0.GENDER", "", 0, 0, 0)

If Gender = MALE Then
    StartAt=240
Else
    StartAt=336
End If

Patient.SetDynamicVisitStart "Visit4Gen", "Visit3", StartAt
```

Removing a dynamic visit

If no patient data has been entered in any form in a dynamic visit, you can remove the visit by calling the RemoveEmptyDynamicVisit method. If you attempt to remove a dynamic visit that has any data entered in any of its forms, the InForm application issues an error message indicating that you cannot remove the visit.

RemoveEmptyDynamicVisit has a single parameter: the RefName of the dynamic visit.

Example of rule for removing a dynamic visit

In this example a patient is removed from an erroneously scheduled dynamic visit and rescheduled into the appropriate visit for the patient's gender.

```
MALE=1
FEMALE=2
MALEVISIT="Dose100"
FEMALEVISIT="Dose200"
Gender = Patient.GetValueRF("0.Visit1.DEM.DEM.0.GENDER", "", 0, 0, 0)

If Gender = MALE Then
    SVisit="MALEVISIT"
    RVisit="FEMALEVISIT"
Else
    SVisit="FEMALEVISIT"
    RVisit="MALEVISIT"
End If

Patient.RemoveEmptyDynamicVisit RVisit
Patient.SetDynamicVisitStart SVisit, "", 720
```


Monitoring forms

Monitoring forms are custom forms that are available in the InForm application when a user clicks the Monitor navigation button. MedML supports the creation of the following types of Monitoring forms:

- **Visit Report**—A form that enables a CRA to report on a visit to a specific site. To accommodate the reporting of multiple visits, you can set up a Visit Report as a repeating Form.
- **Regulatory Document Checklist**—A form that enables a CRA to record a site’s compliance with documentation requirements such as the availability of current trial documents and regulatory documents such as the trial protocol, IRB approvals, and electronic signature reporting documentation.

Note: Even if you do not plan to collect visit report or regulatory documentation data, you should create blank forms as placeholders.

Creating a Visit Report

To create a Visit Report:

- 1 Using the appropriate MedML components, define a form that captures the data you want to include in the Visit Report. The form must have the following characteristics:
 - The form must contain a section definition consisting of a single Item containing a date/time control definition that includes the month, day, year, hour, and minute of the visit, of which at least the month and year must be required.
 - The Visit report component definitions must include the following specific required UUIDs:

Component	UUID
Visit Report visit	PF_UUID_VISITREPORT_FORMSET
Visit Report form	PF_UUID_VISITREPORT_FORM
Visit Report section	BD991BBE-B0A4-11D2-80E3-00A0C9AF7674
Visit Report item	BD991BBF-B0A4-11D2-80E3-00A0C9AF7674
Visit Report date/time control	BD991BC0-B0A4-11D2-80E3-00A0C9AF7674

- 2 In the form definition, specify **VISITREPORT** as the value of the TYPE attribute.
- 3 Include the form in a visit definition that has the following characteristics:
 - The value of the visit TYPE attribute is **MONITOR**.
 - The value of the UNSCHEDULED attribute is **True**.
 - The value of the REPEATING attribute is **True**.

Creating a Regulatory Document Checklist

To create a Regulatory Document Checklist:

- 1 Using the appropriate MedML components, define a form that captures the data you want to include in the Regulatory Document Checklist.
- 2 In the form definition, specify **REGDOC** as the value of the TYPE attribute.
- 3 Include the form in a visit definition that has the following characteristics:
 - The value of the visit TYPE attribute is **MONITOR**.
 - The value of the UNSCHEDULED attribute is **True**.

Note: These instructions describe how to set up a Regulatory Document Checklist that is not repeating. To create a repeating Regulatory Document Checklist, follow the instructions for *creating a Visit Report* (on page 357), substituting REGDOC as the value of the Form TYPE attribute.

Patient navigation mode

In addition to the primary mode of patient navigation, in which you enter each patient's Case Book from the Patients list, the InForm application enables users to choose the following alternative methods of navigating through patient forms:

- **Navigate by visit**—Navigate to the same visit for each patient by clicking a navigation button.
- **Navigate by form**—Navigate to the same form in the current visit for each patient by clicking a navigation button.

Additionally, users can reorder the list of patients.

Enabling patient navigation modes

To enable patient navigation modes:

- 1 Set the value of the **Navigation Mode** system configuration parameter to **Enabled**, by using the **System Configuration** screen in the InForm application.
- 2 Add the **Navigation** rights to the appropriate rights groups by using the **Rights Group** screen in the InForm application or the InForm Architect application. The **Navigation** rights are:
 - Navigate by Visit
 - Navigate by Form
 - Reordering of Patients

Patient ID information

This section describes:

- How to enable users to create and edit patient numbers.
- How to allow users to change the current Patient ID after enrollment.
- How patient identification information interacts with the patient record transfer activity.

If you do not create the definitions described in this section, the InForm application automatically generates patient numbers by using the patient number Sequence definition loaded into the database as part of the Base trial.

Patient Initials data entry item

To enable users to create and edit patient numbers, set up the Patient Initials and Patient Number data entry items on the Screening and Enrollment forms.

The Patient Initials data entry item is a required item on the Screening form. When you define the Patient Initials item, the following requirements apply:

- Attach the calculation Rule that determines screening success or failure to the Patient Initials item.
- If you are mapping Screening form data items to another form by using ITEM MAP tags in the StudyVersion definition, do not map the Patient Initials item.
- Use the following UUIDs to define the Patient Initials item:

Component	UUID
Patient Initials item	AEB64F16-127C-11D2-A41C-00A0C963E0AC
Patient Initials textbox control	AEB64F17-127C-11D2-A41C-00A0C963E0AC

Patient Number data entry item

To enable users to create and edit patient numbers, set up the Patient Initials and Patient Number data entry items on the Screening and Enrollment forms.

The Patient Number data entry item is an optional item on the Enrollment form. If you do not create a Patient Number item, or if a user leaves the item blank and overrides the enrollment failure, the InForm application automatically assigns a patient number based on the default or specified sequence number format for patient numbers.

If you create a Patient Number data entry item, the following requirements apply:

- Do not attach the calculation Rule that determines enrollment success or failure to the Patient Number item.

- Ensure that the Patient Number item is in the same visit, form, and section of the Enrollment form as the item to which you attach the calculation Rule.
- If you are mapping Enrollment form data items to another form by using ITEM MAP tags in the StudyVersion definition, do not map the Patient Number item.
- Use the following UUIDs to define the Patient Number item:

Component	UUID
Patient Number item	3D25EE4C-7F1B-11D2-A728-00A0C977C64B
Patient number textbox control	433DAFF6-7F1C-11D2-A728-00A0C977C64B

Editable patient ID

To enable users to change the Patient ID after enrollment is complete, set up a special Patient ID form that has the following required UUIDs:

Component	UUID
Visit containing Patient ID form	03B0D5D8-7F2C-11D2-A728-00A0C977C64B
Patient ID form	06702B62-7ED6-11D2-A728-00A0C977C64B
Patient ID section	3D25EE4B-7F1B-11D2-A728-00A0C977C64B
The Patient Number item used on the Enrollment form	3D25EE4C-7F1B-11D2-A728-00A0C977C64B
Patient Number text box control, if the section contains the Patient Number item used on the Enrollment form	433DAFF6-7F1C-11D2-A728-00A0C977C64B
A Change Initials item	D959FE72-7F1C-11D2-A728-00A0C977C64B
Change Initials text box control, if the section contains a Change Initials item	433DAFF7-7F1C-11D2-A728-00A0C977C64B

Recommendation for note in patient ID section

When you create a Patient Initials or Patient Number data item on a CRF by using the special UUIDs for these elements, a user can change the patient initials or number that was assigned during the screening or enrollment process. This change actually updates the initials or number by which a patient is identified throughout the InForm application, although the Screening and Enrollment forms continue to reflect the values that applied at the time of enrollment.

If a user clears the data in either of these data items, the data item is blank, but the patient initials or number do not change in the database. It is strongly recommended that you provide a note that explains the consequences of editing or clearing these data items. To create such a note, use the NOTE property in the Section definition where these items occur. Suggested text:

Note: If you change the value of the Patient Initials or Patient Number data item, the InForm application uses the new values throughout the system to identify the patient. If you clear the value of the Patient Initials or Patient Number data item, the InForm application changes the data item to blank but continues to use the old values throughout the system to identify the patient.

Patient identification and patient record transfer

When a user transfers a patient to a different site by using the patient record transfer feature, the InForm application checks the patient initials and patient number for potential duplicates at the destination site. If you plan to allow for the transfer of patient records in your trial, be aware of the following interactions with system configuration settings.

Patient initials

The system configuration setting UniqueIntlDOBSwch (Require unique patient initials and date of birth in the InForm application Admin user interface) enables you to specify that the InForm application requires all combinations of patient initials and date of birth to be unique across a trial or across a site. The default setting is not to require unique initials and dates of birth.

If unique patient initials and dates of birth are required at the trial or site level, the InForm application does not process the transfer of patients whose combined initials and dates of birth are duplicated at the destination site. To prevent the possibility of conflict, you can set this switch to **None** in a trial where patient transfer is anticipated.

Patient numbers

The system configuration setting UniquePatIDSwch (Require Unique Patient ID in the InForm application Admin user interface) enables you to specify that the InForm application requires all patient numbers to be unique across a trial or across a site. The default setting is not to require unique patient numbers.

In a trial where patient record transfer occurs, the InForm application detects a duplicate patient number at the destination site and allows users to change the patient number to one that is not in conflict. To prevent the possibility of conflict and the necessity of changing patient numbers, you can set the UniquePatIDSwch configuration variable to require unique patient numbers across the trial.

Trial designers can set up patient numbers to include the site mnemonic as a portion of the patient number and can write rules that use this mnemonic. If a patient is transferred to a new site, the site information embedded in the patient number is no longer accurate. Therefore, this practice is not recommended for trials in which patient transfer is anticipated.

Rules on patient identification components

When a patient transfers to a new location, the InForm application does not automatically run rules. Therefore, any rules that are attached to patient identification items (patient initials, date of birth, or number) do not automatically run against the patient in the destination site.

Patient record transfer affidavit

The patient record transfer feature of the InForm application enables users to transfer a patient's clinical record to a new site. This feature is useful in situations where:

- A patient moves to a different location.
- A patient moves seasonally to another address.
- An investigator leaves a trial and closes down a site.

When a user transfers a patient to another site, the InForm application displays an affidavit that provides guidance on the responsibility of the administrative user regarding moving a patient from one site to another. The Patient Record Transfer page in the Base trial provides default affidavit text.

Customizing the patient record transfer affidavit

To customize the patient record transfer affidavit:

- 1 Copy the `sysform_PatientSiteChange.xml` file in the `\InForm\XMLBase` folder of the InForm application installation to the `\Custom` folder for your trial. Do not make changes directly in the `system_PatientSiteChange.xml` file.
- 2 Edit the affidavit portion of the `<FORM>` tag in the copy of the `sysform_PatientSiteChange.xml` file. Within the affidavit text, you can use HTML characters for formatting and highlighting.
- 3 Add the edited copy of the file to the `.rsp` file you use to load your trial metadata into the trial database.

Patient transfer affidavit XML

The `<FORM>` tag for the Patient Record Transfer page follows with the default affidavit text highlighted.

Caution: Do not change anything in the form definition other than the highlighted text.


```

<FORM REFNAME="PSC_FORM" UUID="3389a7c6-cba5-46f2-b48f-8e034a28a349"
NOTE="<TABLE BORDER="0"CELLPADDING="4" CELLSPACING="0" WIDTH="100%"
ID="Table1"> <TR> <TD CLASS="ttl"><B>Patient Record Transfer Affidavit:</B></TD>
</TR> <TR> <TD><TABLE CLASS="tbl" BORDER="1" CELLPADDING="4"
CELLSPACING="0" WIDTH="100%" BORDERCOLOR="#003366" ID="Table2"> <TR>
<TD><TABLE BORDER="0" CELLPADDING="4" CELLSPACING="0" WIDTH="100%"
ID="Table3"> <TR> <TD><P><B>The sponsor takes responsibility for and has clear
processes/procedures in place around the handling of patient transfers and coordinating across
sites to ensure that:</P> <P>1. The originating site produces or receives a certified copy of the
current CRF data created for all transferred patients prior to the transfer.</P> <P>2. The target site
accepts responsibility for the management of any outstanding queries and data cleaning following
the patient record transfer, since the originating site will no longer have modification access.</P>
<P>3. Trial version differences due to protocol amendments or country specific data requirements are
acknowledged and the sponsor has appropriate regulatory approvals in place to support patient
record transfer.</P> <P>4. The patient(s) being transferred are fully enrolled and a Case Book has
been created for each patient.</P></TD> </TR> </TABLE> </TABLE> </TABLE>"
TITLE="Patient Record Transfer Form" MNEMONIC="PSC_FORM" QUESTIONWIDTH="30"
CONTROLWIDTH="70" TYPE="CUSTOMTRIAL">

<SECTIONREF REFNAME="PSC_SECT2" />

<SECTIONREF REFNAME="PSC_SECT" />

</FORM>

```

Randomization

The randomization feature of the InForm application enables users to assign a drug kit to a patient based on a randomization scheme that has been chosen for the trial. When randomization configuration is complete, one of the trial's forms contains a Drug Kit section. When a user clicks the Randomize button, the InForm application returns a drug kit number, along with associated information about the drug kit, in the Drug Kit section of the form. Before a user can use the randomization feature, you must perform the following configuration activities:

- 1 **Create a randomization sequence** (on page 367) for each different list of drug kits used in the trial. The number of drug kit lists depends on the randomization method chosen:
 - **Simple Central (Type 1)**—The trial uses one list of drug kits. Each new patient is assigned the next sequential drug kit number on the list.
 - **Central Stratified (Type 2)**—The trial has multiple lists of drug kits. Each new patient is assigned to a drug kit list based on entered patient data. Then, the patient is assigned the next sequential drug kit number on that list.
 - **Simple Site (Type 3)**—Each site has a different drug kit list. Each new patient is assigned the next sequential drug kit number on the list for the patient's site.
 - **Stratified by Site (Type 4)**—Each site has multiple lists of drug kits. Each new patient is first assigned to the set of lists for the patient's site. Then, the patient is assigned to one of the site's drug kit lists based on entered patient data. Finally, the patient is assigned the next sequential drug kit number on that list.
- 2 **Add the Drug Kit section and randomization control** (on page 368) to the form from which users will randomize patients.
- 3 **Write a randomization rule** (on page 368) that determines the drug kit list from which to assign a drug kit number. This rule fires when a user clicks the Randomize button.
- 4 **Configure the randomization data source manager** (on page 369) (COM object) to be used.
- 5 **Configure the format of each randomization sequence** (on page 370) to be used.
- 6 **Set up a randomization source database** (on page 370) for each trial.
- 7 **Create an ODBC connection** (on page 374) for each randomization source database. You can do this manually or by using the PFAdmin utility.
- 8 **Configure each trial** (on page 376) to use the randomization data source defined for it. This step is necessary only if you create the ODBC connection manually. If you perform that step by using PFAdmin, the utility performs both the DSN creation and configuration automatically.

Randomization sequences

A trial that uses randomization requires the following sequence definitions:

- One SequenceType definition for the Randomization sequence. This definition is included in the Base trial and is in the following format:

```
<SEQUENCETYPE SEQUENCETYPENAME="Randomization" />
```

- A Sequence definition for each randomization sequence needed for the trial. You need to define one randomization Sequence element for each randomization source. For example:
 - If your trial uses the Simple Central randomization method, you need one Sequence definition for the single list of randomization kits.
 - If your trial has ten sites and it randomizes by using the Simple Site randomization method, you need ten Sequence definitions, one for each site.
 - If your ten-site trial uses the Stratified by Site randomization method, and you categorize patients into one of three stratifications, you need thirty Sequence definitions.

Each Sequence definition requires the following attributes:

- **SEQUENCENAME**—Name used when referring to the Sequence in the randomization rule and in the randomization source database.
- **SEQUENCETYPENAME**—"Randomization," indicating that the sequence is of the Randomization SequenceType.
- **UUID**—Universally Unique Identifier: a string that identifies the element uniquely across all trials, trial databases, and machines.

Note: The format of the UUID must follow the format used in the sample UUIDs provided in the Base trial. A suggested method for generating UUIDs for Sequence definitions is to copy the UUID of the randomization Sequence definition provided in the core_DefinitionSequences.xml file and change one character for each Sequence definition you create. This ensures that the format is correct and each string is unique. Alphabetic characters in UUIDs should be uppercase.

Example of a randomization sequence

The following sample .xml file illustrates the definition of the Randomization SequenceType and several randomization sequences. The first is for a Simple Central randomization scheme, and the other two define sequences for sample randomization sources used in a Central Stratified randomization scheme.

```
<?xml version="1.0"?>
<MEDMLDATA xmlns="PhaseForward-MedML-Inform4"><!--SequenceType definition -->
<SEQUENCETYPE SEQUENCETYPENAME="Randomization" />
<!-- Sample Randomization Sequences -->
<!-- Type 1 randomization sequence -->
<SEQUENCE SEQUENCENAME="SimpleCentral"
  SEQUENCETYPENAME="Randomization"
  UUID="6DFF68EE-759C-11D2-938C-00A0C9769A13" />
<!-- Type 2 randomization sequences -->
<!-- Sequence _WT150 for Weight 90 up to 150 -->
<SEQUENCE SEQUENCENAME="CS_WT150"
```

```

SEQUENCETYPENAME="Randomization"
UUID="C31355CE-7598-11D2-938C-00A0C9769A13" />

<!-- Sequence _WT275 for Weight above 150 up to 275 -->
<SEQUENCE SEQUENCENAME="CS_WT275"
SEQUENCETYPENAME="Randomization"
UUID="D5270A30-7598-11D2-938C-00A0C9769A13" />

</MEDMLDATA>

```

Drug Kit section and randomization control

Include the predefined DRUGKITSECTION section in the definition of the form used for randomizing patients. This section includes a drug kit item containing a calculated control in which the InForm application inserts the drug kit information. The InForm application adds a Randomization button to the form that includes the DRUGKITSECTION section.

You can revise the DRUGKITSECTION definition by including it in a different Form definition or by changing the text of the calculated control or item. However, do not change the UUIDs associated with these definitions. The following table shows the required UUIDs for the Drug Kit section and randomization item and control:

Component	UUID
Randomization calculated control	DC2EB0BF-4F12-11D2-9319-00A0C9769A13
DRUGKIT item	52AF1207-4F13-11D2-9319-00A0C9769A13
DRUGKITSECTION section	C0482B37-4F13-11D2-9319-00A0C9769A13

Randomization rule

Use VBScript to create a rule script that gets the next sequence number for a patient being randomized. You can use form rule and calculation script objects as well as the Randomization object. If the rule is for a stratified randomization (Type 2 or Type 4), include a test of each patient data item on which it depends.

Example of a Simple Central randomization script

The following example illustrates a randomization rule for a Simple Central (Type 1) randomization scheme.

```

Randomization.SiteID = Patient.GetSiteID
Randomization.Type = 1
Randomization.Source = "SimpleCentral"
Randomization.PatientID = Patient.GetID
Randomization.Revision = 0
SeqNum = Randomization.GetNextKit(KitInfo)
Result.Result= SeqNum + " / " + KitInfo

```

Example of a Central Stratified randomization script

The following example illustrates a randomization rule for a Central Stratified (Type 2) randomization scheme. In this example, the stratification is based on the patient's weight, as entered in the Demographics form in Visit 1. Based on the patient's weight, the InForm application gets the next sequence number from either the CS_WT150 or the CS_WT275 randomization source list.

```
Function GetRndSourceList()
wt = Patient.GetValueRF("0.Visit1.DEM.DEM.0.WEIGHT", "", 0,0,0)
IF (wt > 90 AND wt <= 150) THEN
GetRndSourceList = "CS_WT150"
ELSEIF (wt > 150 AND wt < 275) THEN
GetRndSourceList = "CS_WT275"
ELSE
GetRndSourceList = ""
END IF
End Function

'set properties
Randomization.SiteID = Patient.GetSiteID
Randomization.Type = 2
Randomization.Source = GetRndSourceList
Randomization.PatientID = Patient.GetID
Randomization.Revision = 0
'randomize the patient and return result
SeqNum = Randomization.GetNextKit(KitInfo)
Result.Result= SeqNum + " / " + KitInfo
```

Using the InForm Architect application, create a context that attaches the rule to the DRUGKIT item in the DRUGKITSECTION section. If the rule is for a stratified randomization, specify the patient data items that it checks as dependent items for the rule.

Notes:

- Special configuration may be required if you are using the Central Stratified randomization type. For more information, see *Setting up Central Stratified randomization* (on page 374).
- A randomization rule can have only one rule context.
- If any items you have specified as dependent items do not have data entered or have queries in Candidate or Opened state, the InForm application does not assign the randomization sequence number.

Randomization source manager configuration

The randomization data source is a Customer-Defined Database that contains the drug kit information for each list of drug kits in use in the trial. The randomization source manager is implemented as a COM object that accesses the randomization data source database. The randomization source manager provided with the InForm application accesses the database through ODBC.

The randomization source manager for the InForm application supports all four randomization methods and Microsoft Access, Microsoft SQL Server, or Oracle 9i randomization source databases. To use this randomization source manager, you must add the configuration name and COM ProgID of the randomization source manager to the system configuration table by using the following statement in the .xml file used to set values in the system configuration table:

```
<SYSCONFIG CONFIGNAME="RandomizationSrc"
TYPE="0" VALUE="InForm.PFRandomization.1" />
```

This statement appears in the core_SystemConfig.xml file in the Base trial.

Randomization sequence format configuration

In the InForm application, you can configure the format of screening numbers, patient numbers, and randomization numbers. To specify the randomization sequence number format, include a SYSCONFIG component for the sequence number format in the .xml file used to set configuration variable values. The following default format appears in the core_SystemConfig.xml file in the Base trial:

```
<SYSCONFIG CONFIGNAME="RandSimpleCentral"  
  TYPE="0"  
  VALUE="SC:RND-%q" />
```

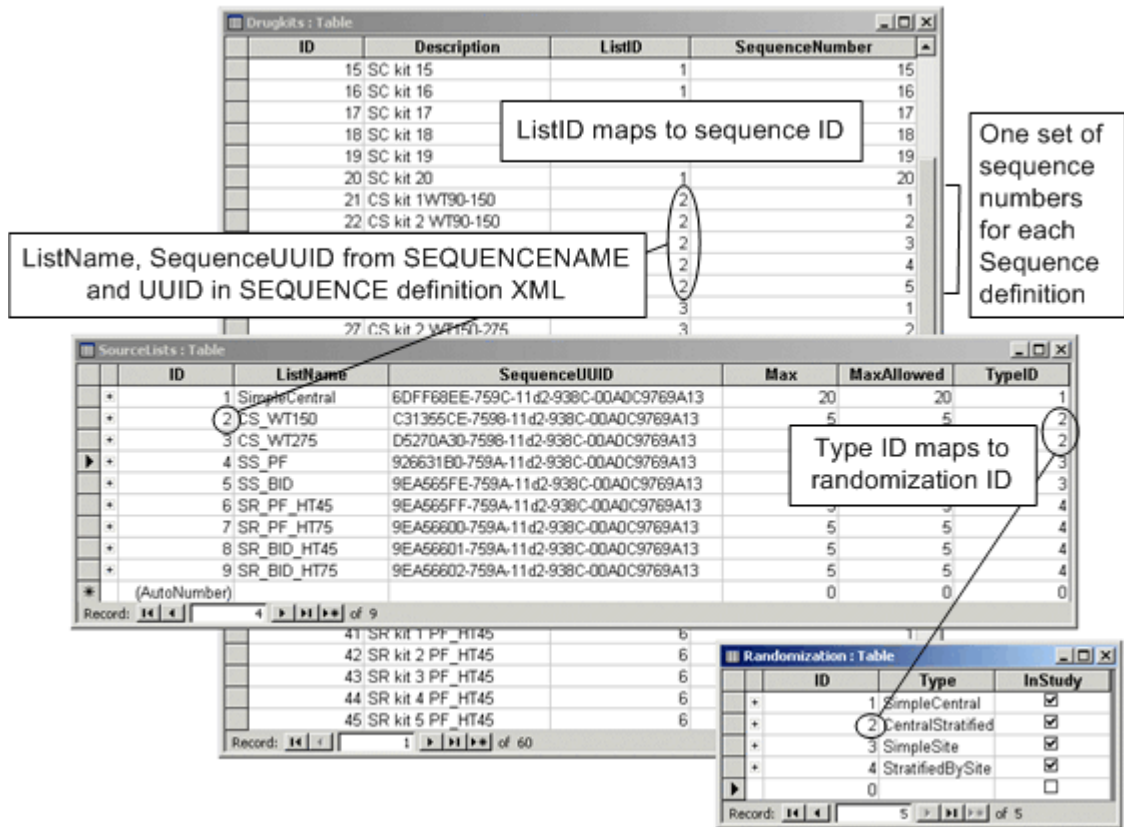
Randomization number configuration is required. For more information, see *Managing numbering* (on page 452).

Randomization source database setup

The randomization data source is a custom database that contains the drug kit information for each list of drug kits in use in the trial. The InForm application supports randomization source databases implemented with Microsoft Access or Oracle 9i. The database has the following tables:

- RANDOMIZATION
- SOURCELISTS
- DRUGKITS

Note: A sample Microsoft Access database is included in the sample trial. To use an Oracle randomization database, create an Oracle schema with a table structure that is identical to the sample Microsoft Access database.



RANDOMIZATION table

The **RANDOMIZATION** table indicates which randomization types are in use in the trial. This table has the following columns:

Column name	Data type	Description
ID	Number, primary key	Number that corresponds to the randomization type number: <ul style="list-style-type: none"> • 1—Simple Central • 2—Central Stratified • 3—Simple Site • 4—Stratified by Site
Type	Text	Name of the randomization type; one of the following: <ul style="list-style-type: none"> • SimpleCentral • CentralStratified • SimpleSite • StratifiedBySite

Column name	Data type	Description
InStudy	Boolean	If true, indicates that the corresponding randomization type is being used in the current trial.

SOURCELISTS table

The **SOURCELISTS** table corresponds to the randomization SequenceTypes entered into the InForm application database with XML and the Trial Set-up Tool. This table has one row for each randomization list used in the trial. The table has the following columns:

Column name	Data type	Description
ID	Autonumber, primary key	Sequential list identification number that maps to the ListID in the DrugKits table.
ListName	Text	Name of the drug kit list. The randomization rule must populate the Randomization.Source property with this name.
SequenceUUID	Text	UUID of the SequenceType to which the drug kit list corresponds in the InForm application database. Note that Alphabetic characters in required UUIDs must be uppercase.
Max	Number	Maximum number of drug kits.
MaxAllowed	Number	Maximum number of drug kits that can be used during the trial.
TypeID	Number	Identification code of the randomization type. This value maps to the ID column in the Randomization table: <ul style="list-style-type: none"> • 1—Simple Central • 2—Central Stratified • 3—Simple Site • 4—Stratified by Site

DRUGKITS table

The **DRUGKITS** table holds drug kit information for each randomization source list. This table has the following columns.

Note: Special configuration may be required if you are using the Central Stratified randomization type. For more information, see *Setting up Central Stratified randomization* (on page 374).

Column name	Data type	Description
ID	Autonumber	Sequential drug kit identification number.
Description	Text	Drug kit information. If the randomization rule so specifies, this information is appended to the randomization number that the InForm application generates.
ListID	Number	Number that maps to the ID in the SourceLists table and enables the randomization source data manager to get the correct drug kit string.
SequenceNumber	Number	Randomization sequence number in the list identified by the ListID number.

Populating randomization source database tables

For each trial, you must populate the tables of the randomization source database with the information about each drug kit that will be available for use in the trial, as follows:

- **In the Randomization table**, specify the type of randomization to be used in the trial.
- **In the SourceLists table**, enter the source name and sequence UUID, copied from the .xml file used to enter SequenceTypes into the InForm application database, for each drug kit list.
- **In the DrugKits table**, enter information about each drug kit to be used.

Note: As a suggested convention, include the name of the trial in the name of the randomization source database, and store the database in the trial directory in the InForm application installation tree.

The sample Microsoft Access database PFST_Rnd.mdb distributed with InForm application provides a template for the randomization source database.

Note: The randomization source database must physically reside on the same machine as the InForm application Server; you cannot map it to the InForm application Server machine by using a network drive.

Setting up Central Stratified randomization

For the Central Stratified type of randomization, special configuration is necessary if you want the sequence numbers in each drug kit list to start with a number other than 1. In the Central Stratified type of randomization, the trial has multiple lists of drug kits. Each new patient is assigned to a drug kit list based on entered patient data. Then, the patient is assigned the next sequential drug kit number on that list.

Under Central Stratified randomization, unless each set of sequence numbers in the DRUGKITS table in the randomization database starts at 1, randomization fails when you try to randomize a trial patient.

If you want your sequence numbers to start with a number other than 1:

- 1 Set up the DRUGKITS table so that:
 - Each set of sequence numbers in the SEQUENCENUMBER column starts with 1.
 - The sequence numbers you want to use are in the DESCRIPTION column.
- 2 In the rule that assigns the sequence number when a patient is randomized, use the GetNextKit method so that the values in the DESCRIPTION column are included in return value provided by the KitInfo variable.

ODBC connection for the randomization database

To configure an ODBC connection for a Microsoft Access or Oracle 9i randomization data source, you can do either of the following:

- Create a randomization DSN manually, by using the ODBC manager of Microsoft Windows.
- Let the PFAAdmin utility create the randomization DSN automatically.

You must create a separate DSN for each trial's randomization database.

Creating an ODBC connection manually

To create a randomization DSN by using the Microsoft Windows ODBC Manager:

- 1 Select **Start > Programs > Administrative Tools > Data Sources (ODBC)**.
- 2 Click the **System DSN** tab.
- 3 Click **Add**.
- 4 In the **Create New Data Source** window, select **Microsoft Access Driver**.
- 5 Click **Finish**.
- 6 *Configure the data source* (on page 375).

Configuring a randomization data source

After creating a randomization source with the Windows ODBC manager, Use this procedure to configure a randomization data source:

- 1 In the **ODBC Text Setup** dialog box, enter the DSN of the randomization database as the **Data Source Name**. As a suggested convention, include the name of the trial in the name of the randomization source database; for example, if the trial name is PF304, you can name the randomization DSN PF304RND.
- 2 In the **Database** section of the dialog box, click **Select Directory**, and select the randomization source database from the directory where it is stored.
- 3 Click **OK** in each dialog box.

Creating an ODBC connection with PFAdmin

The *PFAdmin utility* is a command-line tool included in the InForm application installation to help with setting up a trial and the InForm application server. To use PFAdmin to create a DSN for the randomization database automatically:

- 1 Open a Windows command window.
- 2 Change to the \bin directory in the InForm application installation.
- 3 Enter the following command:

```
PFAdmin config trial <TrialName> /Rnd [<MDBFilePath>
[UID] [PID]]
```

where:

- *TrialName* is the name of the trial.
- *MDBFilePath* is the full pathname of the Microsoft Access randomization source database.
- *UID* is username of the trial schema owner.
- *PID* is the password of the trial schema owner.

For example:

```
PFAdmin config trial PF304 /Rnd
F:\InForm2\Trials\PF304\Template\RndPF304.mdb
pf304db pf304db
```

Configuring the trial to use the randomization DSN

After manually creating a randomization source DSN for a trial, you must configure the trial to use the DSN.

Note: This step is necessary only if you use the Control Panel ODBC manager to create the randomization DSN. If you use the PFAdmin utility in the previous step, it automatically configures the trial to use the randomization DSN.

Configure each trial separately. Use PFAdmin to do this:

- 1 Open a Windows command window.
- 2 Change to the \bin directory in the InForm application installation.
- 3 Enter the following command:

```
PFAdmin CONFIG Trial <TrialName> /RndDSN DSN  
[UID] [PID]
```

where:

- *TrialName* is the name of the trial.
- *DSN* is the name of the randomization source DSN. For more information, see *Creating an ODBC connection for the randomization database* (on page 374).
- *UID* is username of the trial schema owner.
- *PID* is the password of the trial schema owner.

Note: Use alphabetic or alphanumeric characters for the UID and PID, and begin them with a letter. Do not use all numeric characters.

For example:

```
PFAdmin config trial PF304 /RndDSN PF304RND  
pf304db pf304db
```

Repeating forms

A repeating form is a type of form that allows multiple instances of the form within the visits where it occurs. To the end user, repeating forms are visible in a summary list from which the user can navigate to individual detailed instances of the form.

As a designer, consider whether to implement multiple collections of the same data in a visit as an itemset or as a repeating form. Both itemsets and repeating forms present users with summary and detailed views of data. Both allow you to address specific instances of the data and specific items within an instance with rules. The principal advantage of repeating forms is that pairs of related forms can be associated. For more information, see *Associations* (on page 343). In an association, users can navigate easily between forms with related information.

Consider the following restrictions concerning repeating forms:

- A date of visit form, containing the DOV control installed with the Base trial, cannot be repeating.
- A repeating form cannot have an alternative version. For more information, see *Alternate versions of forms* (on page 341).
- After a repeating form definition has been installed in the InForm application database, you cannot change its Repeating property.

Creating a repeating form

To create a repeating form:

- 1 Select **File > New > Form**. A new Form window appears in the application workspace.
- 2 In the **Properties** window, select **True** as the **Repeating Form** property.
- 3 Follow the rest of the procedure for creating a regular form.

Enhancing repeating form navigation

In a repeating form, you can specify that the values of certain items appear in the pulldown list used to navigate through form instances. These items are called key items. For more information, see *Using key items* (on page 113).

Reports

To ensure that data is reported appropriately in standard and ad hoc InForm application reports, observe the trial design considerations described in this section.

Calculations

Calculations enable you to set the value of a data item by performing a calculation that uses the value entered in one or more related items.

Consider this example:

A form might contain items for Wrist Circumference, Weight, and Frame Size. Wrist Circumference and Weight are regular items for which users enter data. Frame Size is a calculated control; the data that populates this item is the result of a calculation performed with values entered in the Wrist Circumference and Weight items.

When designing calculations, avoid changing the data type of the data that populates the target item. When a calculation changes the data type, reports that contain the target item do not return data on that item.

Date of Visit

Form aging and cycle time reports show the amount of time it takes for data to be captured throughout various steps of the electronic data entry process. These calculations begin with the date of visit (DOV).

In trial design code, you can design DOV controls in a number of different ways. Oracle recommends designing DOV controls in a particular way. In general, the DOV control should:

- Have the following Universal Unique Identifier (UUID):
BD991BC0-B0A4-11D2-80E3-00A0C9AF7674
- Be included in an item that is identified with the following UUID:
BD991BBF-B0A4-11D2-80E3-00A0C9AF7674
- Appear within a form section that is identified with the following UUID:
BD991BBE-B0A4-11D2-80E3-00A0C9AF7674
- Not appear on a common form.

Although not required, the following are also good practices in DOV design:

- Include the Date of Visit section on the first form of every visit. This Date of Visit section is the same as the Date of Visit section used in *Unscheduled Visits*.
- Configure InForm application to enforce visit date reporting by setting the value of the `EnforceVisitDate` configuration parameter to 1 (yes). Use the MedML `SYSCONFIG` tag. For more information, see *Defining system configuration settings* (on page 456).

The Reporting and Analysis retrieves DOV data from the database assuming that your trial has been built in this way. The DOV used in reporting is a date entered by an InForm application user into a DOV control designed with specific UUIDs mentioned above.

What if no date of visit is entered?

If you have designed your trial using the special DOV UUIDs, InForm application users may still leave the DOV blank. If this is the case, DOV columns and columns containing calculations that rely on a DOV will be blank.

What if the data of visit entered is later than the data entry date?

If InForm application users enter a DOV that is later than the data entry date, resulting reports show negative values in cycle time columns.

Itemset column headers

When defining an item, you can specify a label in the Itemset Column Header property. If the item is used in an itemset, this label appears as the column heading in the row and column representation of the itemset.

In reporting, the Itemset Column Header property is also used as the default short name for the field in the clinical reporting model, whether or not the item is used in an itemset. You can override this name when customizing reports.

Note: You should ensure that Itemset Column Header definitions are unique within a trial. If the Cognos reporting system finds duplicate Itemset Column Header values, it combines the items in reports.

Required visits and items

The CRF and Case Book Status reports determine the status of CRFs and Case Books based on what data is required and what data has been entered. A CRF or Case Book is complete for the purpose of this report only when all items marked as required have data entered. For this reason it is important to set items and visits as required, or not required, according to trial needs.

Study completion and dropout reason

For the Patient Status and Patient Dropout Reports to report completion and dropout status correctly, the Case Book for the trial must include a Study Completion form that contains certain required trial components. Additionally, to specify report column headings associated with the noncompletion reasons in the Study Completion form, you must create a text resource for each noncompletion reason on which you want to report.

For information about the required trial components in a Study Completion form, and the text resources required for dropout reporting, see *Required Study Completion form components* (on page 393).

Study version acceptance dates

Report designers can report on the following site dates:

- **Site Activation Date**—The projected date for the activation of the site. This date is specified in the definition of the site and reflects an estimate of when the activation might occur, not the actual activation date.

You can specify site activation date by using the STARTDATE attribute of the <SITE> tag in the trial metadata. (You cannot specify this value by using the InForm application or InForm Architect application Admin user interface.)

- **Current Site Acceptance Date**—The date and time that a site accepts the current study version. This date is one of the following:
 - The specified value of the ACCEPTDATE attribute of the <STUDYVERSIONSITE> tag defined in the trial metadata. The <STUDYVERSIONSITE> tag associates a site with a particular study version.
 - The system date and time when a user selects the current study version on the Site detail page of the InForm application Admin user interface.
- **Sponsor Date**—The date that appears in the annotated view of the trial. In the InForm Architect application, you can specify this date and time (in GMT) by using the Sponsor Date property of the trial definition. The default is the date and time when you generate the trial XML by saving the trial components. In the MedML definition of the study version, this date is specified in the SPONSORDATE attribute of the <STUDYVERSION> tag.

Subject states

To capture patient screening, enrollment, and randomization states in reports, you must use the screening, enrollment, and randomization features of the InForm application. Otherwise, the reports that show patient status only show patients as enrolled.

Visit start hour

The standard CRF Performance, Aging, and Cycle Time reports, and any ad hoc reports that deal with performance, aging, and cycle times calculate the number of CRFs expected to be completed by a certain date. This calculation uses the Start Hour property of the visit trial component to set the time and date on which the visit is expected to occur.

Although the Start Hour property is optional, to ensure that the calculation of expected CRFs is accurate in reports, you should include this value in your visit definitions.

For more information about the calculation of expected CRFs in reports, see the *Reporting and Analysis Guide*.

Screening and enrollment forms

Each trial is required to have a Screening form and Enrollment form. To enable the special system processing that occurs when a user submits data for these forms, the forms have several implementation requirements. When you create forms, you define item, section, form, and visit components. Screening and Enrollment forms have the following requirements for these components:

- The Screening form must be the only form in its visit, which is defined by a screening visit.
- The Enrollment form must be the only form in its visit, which is defined by an enrollment visit.
- When you define Screening and Enrollment visits, forms, sections, and items, you must include specific UUID attributes. For more information, see *Screening UUIDs* (on page 381) and *Enrollment UUIDs* (on page 382).
- The Screening form must include the Patient Initials item, which requires specific UUIDs. For more information, see *Patient Initials data entry item* (on page 360).
- The Enrollment form can include an editable Patient Number item, which requires specific UUIDs. For more information, see *Patient Number data entry item* (on page 360).
- At least one item is required on each form.

Screening UUIDs

The following table lists the required components and UUIDs for a screening form:

Component	UUID
Screening visit	D882CE38-0F42-11D2-A419-00A0C963E0AC
Screening form	D882CE39-0F42-11D2-A419-00A0C963E0AC
Screening form section that contains the Patient Initials data entry item	96CAE354-126C-11D2-A41C-00A0C963E0AC

Enrollment UUIDs

When you create an enrollment visit and form, well-known UUIDs are required in the visit and form definitions. The required UUIDs appear in the table that follows. The InForm Architect application automatically provides the UUID properties in the visit and form. When you create an enrollment form section that contains the Patient Number data entry item, you must include the UUID shown for the section.

Note: A trial can have only one Enrollment form.

Component	UUID
Enrollment visit	D882CE3A-0F42-11D2-A419-00A0C963E0AC
Enrollment form	D882CE3B-0F42-11D2-A419-00A0C963E0AC
Enrollment form section that contains the Patient Number data entry item	ABCFA388-223A-11D2-A426-00A0C963E0AC

The Enrollment form can include an editable Patient Number item. The Patient Number item is optional. If you do not include a Patient Number item, the patient number is assigned automatically by the InForm software.

If the Enrollment form includes an editable Patient Number Item, that item requires the following UUIDs.

Component	UUID
Patient Number item	3D25EE4C-7F1B-11D2-A728-00A0C977C64B
Patient number textbox control	433DAFF6-7F1C-11D2-A728-00A0C977C64B

Screening or enrollment copy calculation rule

To map data from a Screening or Enrollment form to another form in the trial, use a specialized calculation rule that copies the desired data when the Screening or Enrollment form is submitted. This rule must include the following attributes:

```
UUID="PF_ENROLLMENT_CALCULATION"
SCRIPTTYPE="SERVERCALCULATION"
```

For example, this is the definition of the rule that copies the patient date of birth, race, and screening date values from the SYSTEM SCREENING form to the DEMOGRAPHICS form in the sample PFST trial:

```
<RULE REFNAME="rulScrMap"
  DESCRIPTION="Maps data from Screening form"
  ENABLED="TRUE"
  SCRIPTTYPE="SERVERCALCULATION"
  UUID="PF_ENROLLMENT_CALCULATION">
```

The script includes the following statements, which use the Patient.SetNewValueRF method to update controls in the DEMOGRAPHICS form:

```
'Mapping Date of Birth
strDOB = Patient.GetValueRF("0.screen.frmScr.sctScrInfo.0.itmScrDOB.dtmScrDOB",
"", 0, 0, 0)
Patient.SetNewValueRF "0.vstBASE.frmDem.sctDem.0.itmDemDOB.dtmDemDOB", "", 0, 0,
0, strDOB
Patient.SetNewValueRF "0.vstBASE.frmDem.sctDem.0.itmDemDOB.dtmDemDOB",
"VALIDDATEMAP", 0, 0, 0, 7
'Mapping Race radio control
strRaceRDC =
CStr(Patient.GetValueRF("0.screen.frmScr.sctScrInfo.0.mitmRace.mrdcRace", "", 0,
0, 0))
Patient.SetNewValueRF "0.vstBASE.frmDem.sctDem.0.mitmRace.mrdcRace", "", 0, 0,
0, strRaceRDC
'Mapping Screen Date
strScreenDt =
Patient.GetValueRF("0.screen.frmScr.sctScrInfo.0.itmScrDate.mdtmScrDate", "", 0,
0, 0)
Patient.SetNewValueRF
"0.vstBASE.frmDem.sctDem.0.itmDemScreenDate.mdtmGlobalDate", "", 0, 0, 0,
strScreenDt
Patient.SetNewValueRF
"0.vstBASE.frmDem.sctDem.0.itmDemScreenDate.mdtmGlobalDate", "VALIDDATEMAP", 0,
0, 0, 7>
```

Note: This is the only circumstance where you should use the SetNewValueRF method.

Using Patient Initials and Patient Number items

The Patient Initials item is required on the Screening form. Patient Number is an optional Item on the Enrollment form. To set up the Patient Initials and Patient Number data entry items on the Screening and Enrollment forms, use the instructions in the following sections:

- *Patient Initials data entry item* (on page 360)
- *Patient Number data entry item* (on page 360)

Forms requiring signature

To comply with regulations set forth in 21 CFR 11, CRFs and Case Books must be signed by users specifically authorized to handle these functions. The InForm application supports electronic signature by using signature groups and by using the SIGNCRF MedML tag.

Overview of implementing electronic signatures

To implement electronic signatures:

- 1 Decide which forms require signatures and which users will be authorized to sign.
- 2 Create the text of a signing affidavit. For more information, see *Creating the text of a signing affidavit* (on page 385).
- 3 Create one or more signature groups with signing affidavits by using the SIGNATUREGROUP tag. Each signature group must have a CRF affidavit, a Case Book affidavit, or both. For more information, see *Creating a signature group* (on page 386).
- 4 Assign users to those groups. For more information, see *Creating a signature group* (on page 386).
- 5 Use the SIGNCRF tag to associate each form that requires signature with the appropriate signature group. For more information, see *Associating a form with a signature group* (on page 389).
- 6 Install the XML definitions for signature groups and SIGNCRF tags in the InForm application database by using the MedML Installer.
- 7 Optionally, use the ViewCRFSignList configuration variable to specify whether you want a list of needed and supplied signatures to be displayed on each CRF for which a signature is required. For more information, see *Displaying a required signature list on a CRF* (on page 392).

Creating the text of a signing affidavit

To be in compliance with 21 CFR 11, electronic signatures must include:

- Full name of the signer.
- Date and time of the signature.
- Meaning of the signature, for example, review, approval, responsibility, authorship.

The InForm application captures the user name and password of the signer and the date and time of signature and records them in the signature history which is displayed on the Signature Details screen. To complete the compliance, make sure that:

- The First Name and Last Name of each person who will be signing CRFs are that person's full name. This insures that the signature history bears the signer's full name.
- The text of the signing affidavit includes the signer's full name. To do this, include two %s variables in the text, one for the first name of the user and the other for the last name.
- The text of the signing affidavit includes the meaning of the signature.

The signing affidavit is a text file that can contain HTML characters for formatting and emphasis. The text of a sample signing affidavit follows for both a CRF and a CRB.

Example

```
CRFTEXT="By my dated signature below, I, %s %s, verify that this case report form
accurately&#13;&#10;displays the results of the examinations, tests, evaluations and
treatments noted within.&lt;BR&gt;&lt;BR&gt;&#13;&#10;Pursuant to Section 11.100 of
Title 21 of the Code of Federal Regulations, this&#13;&#10;is to certify that I intend that
this electronic signature is to be the legally&#13;&#10;binding equivalent of my
handwritten signature.&lt;BR&gt;&lt;BR&gt;&#13;&#10;To this I do attest by supplying
my user name and password and clicking the button marked &lt;B&gt;Submit&lt;/B&gt;
below.&#13;&#10;"
```

```
CRBTEXT="By my dated signature below, I, %s %s, verify that all case report form
pages&#13;&#10;accurately display the results of the examinations, tests, evaluations and
treatments&#13;&#10;performed on this
patient.&lt;BR&gt;&lt;BR&gt;&#13;&#10;Pursuant to Section 11.100 of Title 21 of the
Code of Federal Regulations, this&#13;&#10;is to certify that I intend that this electronic
signature is to be the legally&#13;&#10;binding equivalent of my handwritten
signature.&lt;BR&gt;&lt;BR&gt;&#13;&#10;To this I do attest by supplying my user name
and password and clicking the button marked &lt;B&gt;Submit&lt;/B&gt;
below.&#13;&#10;"
```

Creating a signature group

A signature group identifies users with permission to sign CRFs or Case Books. To be fully signed, a CRF must be signed by a representative of each signature group with which it is associated.

Note: You associate CRFs and Case Books with a signature group by using the SIGNCRF MedML tag. For more information, see *Associating a form with a signature group* (on page 389).

To create a signature group, use the SIGNATUREGROUP tag.

```
<SIGNATUREGROUP
  GROUPNAME="name"
  [ GROUPDESCRIPTION="text" ]
  [ CRFTEXT="text" ]
  [ CRFFILE="file" ]
  [ CRFMEANING="text" ]
  [ CRBTEXT="text" ]
  [ CRBFILE="file" ]
  [ CRBMEANING="text" ]>
  <USERREF* attributes/>
</SIGNATUREGROUP>
```

Using one or more USERREF child elements, add each user you want in the group to the SIGNATUREGROUP definition. These users must have the right to sign a CRF or Case Book, as appropriate. The USERREF element has a single USERNAME attribute, which specifies the user name of the user whom you are assigning to the group.

The following table describes the attributes of the SIGNATUREGROUP tag:

Attribute	Description
GROUPNAME	Name of the signature group. REQUIRED..
GROUPDESCRIPTION	Text describing the signature group. OPTIONAL.
CRFTEXT	Text of the Electronic Signature Affidavit displayed to members of the signature group when signing a CRF associated with the signature group. OPTIONAL; if used, specify either the CRFTEXT or CRFFILE attribute. If you do not specify either attribute, MedML Installer uses the text provided in a default CRF signing text resource. This resource has the UUID PF_DEFAULT_CRFAFFADAVIT. The text must include the stated intention of the signer for the electronic signature to be the legal equivalent of a handwritten signature. The text should include two %s characters to represent the user's first and last names.

Attribute	Description
CRFFILE	<p>Path name of an HTML or text file containing the text of the Electronic Signature Affidavit displayed to members of the signature group when signing a CRF associated with the signature group. The path name must be relative to the directory from which you run the MedML Installer.</p> <p>OPTIONAL; if used, specify either the CRFTEXT or CRFFILE attribute. If you do not specify either attribute, MedML Installer uses the text provided in a default CRF signing text resource. This resource has the UUID PF_DEFAULT_CRFAFFADAVIT.</p>
CRFMEANING	<p>Text that summarizes the meaning of the signature on the CRF. This text is displayed on the Signature Details screen and in the list of completed and required signatures on the CRF.</p> <p>OPTIONAL. Required if either the CRFTEXT or CRFFILE attribute is specified. At least one combination of CRFTEXT or CRFFILE and CRFMEANING should be specified.</p>
CRBTEXT	<p>Text of the Electronic Signature Affidavit displayed to members of the signature group when signing a Case Book associated with the signature group.</p> <p>OPTIONAL; if used, specify either the CRBTEXT or CRBFILE attribute. If you do not specify either attribute, the MedML Installer uses the text provided in a default Case Book signing text resource. This resource has the UUID PF_DEFAULT_CRBAFFADAVIT.</p> <p>The text must include the stated intention of the signer for the electronic signature to be the legal equivalent of a handwritten signature.</p> <p>The text should include two %s characters to represent the user's first and last names.</p>
CRBFILE	<p>Path name of an HTML or text file containing the text of the Electronic Signature Affidavit displayed to members of the signature group when signing a Case Book associated with the signature group. The path name must be relative to the directory from which you run the MedML Installer.</p> <p>OPTIONAL; if used, specify either the CRBTEXT or CRBFILE attribute. If you do not specify either attribute, the MedML Installer uses the text provided in a default Case Book signing text resource. This resource has the UUID PF_DEFAULT_CRBAFFADAVIT.</p>
CRBMEANING	<p>Text that summarizes the meaning of the signature on the Case Book. This text is displayed on the Signature Details screen and in the list of completed and required signatures on the CRF used for signing a Case Book.</p> <p>OPTIONAL. Required if either the CRBTEXT or CRBFILE attribute is specified. At least one combination of CRBTEXT or CRBFILE and CRFMEANING should be specified.</p>

Example

```
<SIGNATUREGROUP
```

```
    GROUPNAME="Investigator Sigs"
```

```
    GROUPEDESCRIPTION="Investigator Sigs"
```

```
    CRFTEXT="By my dated signature below, I, %s %s, verify that this case report form accurately&#13;&#10;displays the results of the examinations, tests, evaluations and treatments noted within.&lt;BR&gt;&lt;BR&gt;&#13;&#10;Pursuant to Section 11.100 of Title 21 of the Code of Federal Regulations, this&#13;&#10;is to certify that I intend that this electronic signature is to be the legally&#13;&#10;binding equivalent of my handwritten signature.&lt;BR&gt;&lt;BR&gt;&#13;&#10;To this I do attest by supplying my user name and password and clicking the button marked &lt;B&gt;Submit&lt;/B&gt; below.&#13;&#10;"
```

```
    CRBTEXT="By my dated signature below, I, %s %s, verify that all case report form pages&#13;&#10;accurately display the results of the examinations, tests, evaluations and treatments&#13;&#10;performed on this patient.&lt;BR&gt;&lt;BR&gt;&#13;&#10;Pursuant to Section 11.100 of Title 21 of the Code of Federal Regulations, this&#13;&#10;is to certify that I intend that this electronic signature is to be the legally&#13;&#10;binding equivalent of my handwritten signature.&lt;BR&gt;&lt;BR&gt;&#13;&#10;To this I do attest by supplying my user name and password and clicking the button marked &lt;B&gt;Submit&lt;/B&gt; below.&#13;&#10;"
```

```
    CRFMEANING="Approval" CRBMEANING="Approval">
```

```
    <USERREF USERNAME="esnow"/>
```

```
    <USERREF USERNAME="alarkin"/>
```

```
</SIGNATUREGROUP>
```


Associating a form with a signature group

You can associate CRFs and Case Books with signature groups. To associate a CRF or Case Book with a signature group, create XML for each form that requires an authorized signature, using the SIGNCRF MedML tag.

One form represents the Case Book for signature purposes. To designate a CRF as the Case Book signing form, specify its RefName as the FORMREFNAME in the SIGNCRF tag, and specify **True** as the value of the FINALCRF attribute.

The SIGNCRF tag has the following syntax:

```
<SIGNCRF
  SIGNATUREGROUPNAME="name"
  FORMREFNAME="name"
  [ RESETFORMSTATE="TRUE | FALSE" ]
  [ INVALIDATIONLEVEL="USER | GROUP" ]
  [ FINALCRF="TRUE | FALSE" ] />
```

The following table describes the attributes of the SIGNCRF tag:

Attribute	Description
SIGNATUREGROUPNAME	Identifies the authorized signature group. The signature group must have already been defined by a SIGNATUREGROUP tag. REQUIRED..
FORMREFNAME	Specifies the RefName of the CRF. The form must have already been defined in the system. Note that screening and enrollment forms cannot be signed. REQUIRED..
RESETFORMSTATE	Specifies the policy for handling already-signed forms when a new signature group is added. Values are: <ul style="list-style-type: none"> • TRUE—Indicates that the form must be approved by a member of the current signature group. This resets the state of fully-signed forms specified in the FORMREFNAME attribute to not fully signed. When a form is reset, the original signatures remain valid. However, the form must now be signed by a member of the newly-associated signature group as well. • FALSE (default)—Indicates that the form need not be re-signed when a new signature group is added. OPTIONAL..

INVALIDATIONLEVEL	<p>Specifies whether a signature should be invalidated when a data item is imported through Autocode or Central Coding processing after the form has been signed. Values are:</p> <ul style="list-style-type: none">• USER—The form or Case Book signature is invalidated if the user who signed can view the item being imported.• GROUP—The form or Case Book signature is invalidated if at least one user in the signature group can view the item being imported. <p>OPTIONAL. If you do not specify the INVALIDATIONLEVEL attribute, the InForm application invalidates the signature whenever the form is edited, either directly or by import. For more information, see <i>Signature invalidation</i> (on page 391).</p>
FINALCRF	<p>Indicates whether the form specified in the FORMREFNAME attribute represents the entire Case Book for signing purposes. Note the following restrictions on the CRF designated as the Case Book signing form:</p> <ul style="list-style-type: none">• Only one form can be marked as FINALCRF• The final CRF cannot belong to more than one visit or to a repeating visit• The final CRF cannot be a repeating form <p>Values are:</p> <ul style="list-style-type: none">• TRUE—The form is the Case Book signature form.• FALSE (default)—The form is not the Case Book signature form. <p>OPTIONAL.</p>

Signature invalidation

When a form or case book is signed, it is expected that data entry is complete and source verification has been performed. However, later information can result in data entry changes after the form or case book has been signed.

When this happens, the InForm application invalidates the previous signature or signatures and updates the signature listings and each form to indicate that the form or case book must be signed again.

Data that has been exported from the InForm application by using AutoCode or Central Coding functionality and reimported after coding has special invalidation considerations. Coded data is often designed to be hidden on a form; if the import of coded data invalidates a signature, it might not be possible to see which item caused the invalidation. Therefore, trial designers can specify whether imported data invalidates a signature.

When coded data is imported to a signed CRF or Case Book, invalidation of the CRF or Case Book is determined by the following:

- Visibility of the imported data.
- Setting of the `INVALIDATIONLEVEL` attribute of the `SIGNCRF` MedML component that associates a form or a Case Book with a signature group.

If the signing user can view the imported item, or if the `INVALIDATIONLEVEL` property is not specified, the signature is always invalidated.

If the trial is designed to base invalidation on the value of the `INVALIDATIONLEVEL` property, the invalidation decision for a CRF or a Case Book is based on the following rules:

- If the `INVALIDATIONLEVEL` value is **USER**, the signature is invalidated only if the signing user can view the imported item.
- If the `INVALIDATIONLEVEL` value is **GROUP**, the signature is invalidated only if at least one user in the signature group can view the imported item.

Note: Invalidation processing for Case Book signatures is invoked if coded data is imported to any CRF in the Case Book, whether or not the CRF requires a signature.

Displaying a required signature list on a CRF

By default, the InForm application displays a list of required signatories on each CRF that requires signatures. This feature is an option that you set with a system configuration variable. To specify whether to display a required signature list:

- 1 Create an XML file containing the following tag:

```
<SYSCONFIG  
  CONFIGNAME="ViewCRFSignList"  
  TYPE="0"  
  VALUE="n"/>
```

where *n* is a code:

- **0**—Do not display required signature list on a CRF.
 - **1 (default)**—Display required signature list on a CRF.
- 2 Install the setting in the trial database using the MedML Installer utility. Note that you must stop and start the trial for a change in setting to take effect.

The current value of the system configuration variable is visible on the Configuration page of the InForm application Admin. See the View CRF signature list item.

Note: If you set up a common dynamic form to require a signature, it only appears on the Signature Listing page for the Common CRF visit. It does not appear on the Signature Listing page of the dynamic visit.

Study Completion form

The Study Completion form records the last time the patient was seen and whether the patient completed the trial. If the patient did not complete the trial, the Study Completion form records the reason for not completing.

The trial must contain a Study Completion form for standard and ad hoc patient status and patient dropout reports to show patient trial completion status.

Note: A trial design can call for a Study Completion form to be designated as the Case Book signature form; however, there is no InForm application requirement for this to be so.

Required Study Completion form components

Along with at least one section and one item, the Study Completion form must contain the design components described in this section.

Note: The UUIDs in the Study Completion form are independent of study versions and apply to all patients in a trial; therefore, you cannot change study completion metadata by creating a new study version.

Visit

The visit in which the Study Completion form is included must not be a repeating visit. It must have the following UUID:

F4699051-69E2-11D2-8FB5-00A0C977C66A

Form

The Study Completion form can appear only once in a trial, and it cannot appear as a repeating form. The Study Completion form must have the following UUID:

7314A6A5-316E-11D2-8F9A-00A0C977C66A

Controls

The following controls are required:

- **Completion status**—The value of this control specifies whether a patient has completed or has not completed the trial. Its required UUID is:

PF_SC_COMPLETECTL

- **Noncompletion reason**—The value of this control indicates the reason why a patient did not complete the trial. Its required UUID is:

PF_SC_REASONCTL

Elements

The following element components are required:

- An element indicating that the patient completed the trial. Its required UUID is:
PF_SC_STUDY_COMPLETE
- An element indicating that the patient did not complete the trial. Its required UUID is:
PF_SC_STUDY_INCOMPLETE

Text resources

For each dropout reason you want to appear as a column heading on standard or ad hoc dropout reports, define a resource with a type of TEXT. You must define resources by using MedML and the MedML Installer tool.

The resource specifies the internal value of the control defining the specific dropout reason and the text of the dropout report column header, using the following format:

```
<RESOURCE
  UUID="PF_SC_REASONCTL_<internal_reason_value">"
  [DESCRIPTION="<resource_description">]
  DATATYPE="TEXT"
  [TEXT="<column_heading_text">"]/>
```

Variable	Definition
<i>internal_reason_value</i>	<p>One of the following:</p> <ul style="list-style-type: none"> • The Value property defined for an element component in a simple or pull-down control included in the control specifying noncompletion reason • The Selection Value property defined for any other type of subordinate control included in the control specifying noncompletion reason • The default value for a subordinate control for which no Selection Value property has been defined. The default value is assigned by InForm application and is in the format <code>!pf!
<control_DBUID_path></code>.
<i>resource_description</i>	Description of the resource.
<i>column_heading_text</i>	Text of the column heading that represents the specified noncompletion reason in standard and ad hoc patient dropout reports.

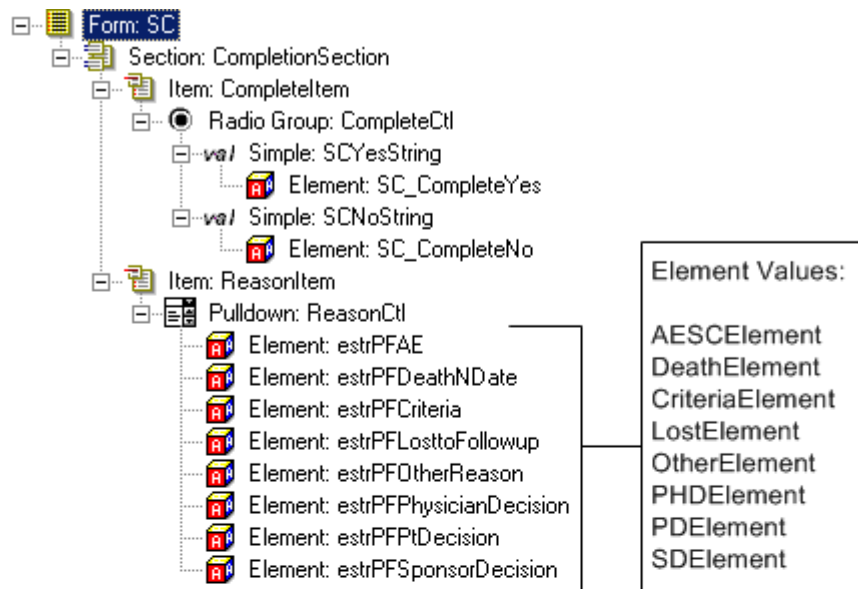
Form design examples

The following examples illustrate some ways that you can use the required design components to create a Study Completion form.

Separate study completion and noncompletion reason items

In this example:

- Study completion is captured in an item that uses a radio control with the PF_SC_COMPLETECTL UUID. The Yes and No choices are defined by element components with the PF_SC_STUDY_COMPLETE and PF_SC_STUDY_INCOMPLETE UUIDs.
- Incomplete reason is captured in an item that uses a pulldown list defined with the PF_SC_REASONCTL UUID. Each selection in the pulldown list is associated with an element component defined with a value.
- Each element component has a corresponding text resource in which the UUID includes the element's value.



STUDY COMPLETION (SC)	
Patient Completion Information	
1. Did the patient complete the study? *	<input type="radio"/> Yes <input type="radio"/> No
2. If patient did not complete study, select primary reason	<div style="border: 1px solid black; padding: 5px;"> Adverse experience Death Did not meet criteria Lost to follow-up Other Physician decision Patient decision Sponsor decision </div>

The following MedML file defines report column headings for noncompletion reasons:

```
<?xml version="1.0"?>
<MEDMLDATA xmlns="PhaseForward-MedML-Inform4">
<RESOURCE
  UUID="PF_SC_REASONCTL_AESCElement" DESCRIPTION="SC AE column heading"
  DATATYPE="TEXT" TEXT="Adverse Experience"/>
<RESOURCE
  UUID="PF_SC_REASONCTL_DeathElement" DESCRIPTION="SC Death column
  heading" DATATYPE="TEXT" TEXT="Death"/>
<RESOURCE
  UUID="PF_SC_REASONCTL_CriteriaElement" DESCRIPTION="SC Criteria
  column heading" DATATYPE="TEXT" TEXT="Did Not Meet Criteria"/>
<RESOURCE
  UUID="PF_SC_REASONCTL_LostElement" DESCRIPTION="SC Lost column
  heading" DATATYPE="TEXT" TEXT="Lost to Follow-Up"/>
<RESOURCE
  UUID="PF_SC_REASONCTL_OtherElement" DESCRIPTION="SC Other column
  heading" DATATYPE="TEXT" TEXT="Other"/>
<RESOURCE
  UUID="PF_SC_REASONCTL_PHDElement" DESCRIPTION="SC PHD column heading"
  DATATYPE="TEXT" TEXT="Physician Decision"/>
<RESOURCE
  UUID="PF_SC_REASONCTL_PDElement" DESCRIPTION="SC PD column heading"
  DATATYPE="TEXT" TEXT="Patient Decision"/>
<RESOURCE
  UUID="PF_SC_REASONCTL_SDElement" DESCRIPTION="SC SD column heading"
  DATATYPE="TEXT" TEXT="Sponsor Decision"/>
</MEDMLDATA>
```

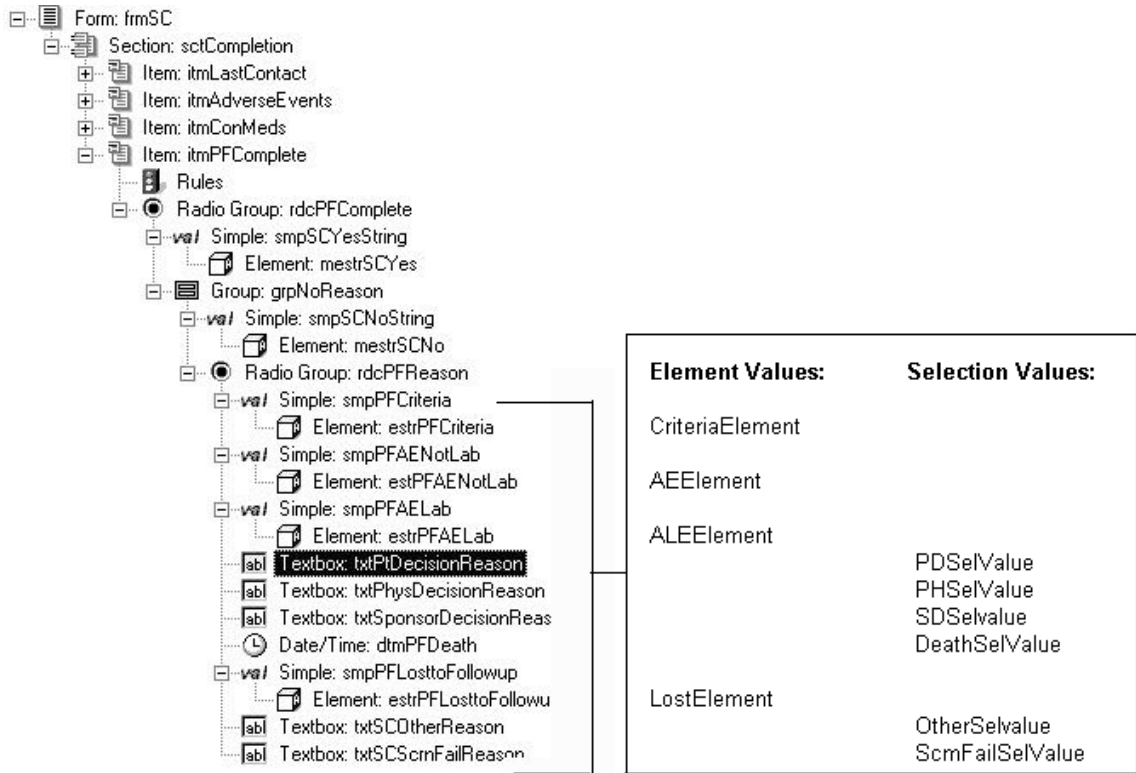
Same item for study completion and noncompletion reason

In this example:

- Study completion and noncompletion reason are captured in a single item that uses a radio control with the PF_SC_COMPLETECTL UUID.
- The **Yes** choice is defined by an element component with the PF_SC_STUDY_COMPLETE UUID. The **No** choice is defined by a radio control with the PF_SC_REASONCTL UUID. The choices in the radio control are defined by simple or group controls. By using group controls you can capture additional information along with each noncompletion reason.

Note: In this example the form does not use an element component with the PF_SC_STUDY_INCOMPLETE UUID. If you use the InForm application to save the trial by component, the application does not save the PF_SC_STUDY_INCOMPLETE element component because it is not used in the trial. Therefore, in this case you must load that element component into the trial database separately with MedML and the MedML Installer tool. The VALUE attribute of the study element component with the PF_SC_STUDY_INCOMPLETE UUID that you load with the MedML Installer tool must be the same as the Selection Value property of the radio control with the PF_SC_REASONCTL UUID.

- The UUID for each text resource that is associated with a noncompletion reason control reflects the element component Value property for each reason defined with a simple control or the Selection Value property for each reason defined with any other type of control.



STUDY COMPLETION Jan/1/2004 00:00		Patient: AAA/01-001
Subject Completion Information		
1.*	Date of last subject contact	<input type="text"/> / <input type="text"/> / <input type="text"/>
2.*	Did the subject experience any adverse events during the Study?	<input type="radio"/> No <input type="radio"/> Yes
3.*	Did the subject receive any concomitant medications during the Study?	<input type="radio"/> No <input type="radio"/> Yes
4.*	Did the subject complete the study?	<input type="radio"/> Yes <input type="radio"/> No, specify <input type="radio"/> Did not meet criteria <input type="radio"/> Adverse experience (other than adverse laboratory experience) <input type="radio"/> Adverse laboratory experience <input type="radio"/> Subject decision, Reason: <input type="text"/> <input type="radio"/> Physician decision, Reason: <input type="text"/> <input type="radio"/> Sponsor decision, Reason: <input type="text"/> <input type="radio"/> Death, specify Date of Death: <input type="text"/> / <input type="text"/> / <input type="text"/> <input type="radio"/> Lost to follow-up <input type="radio"/> Other, Specify: <input type="text"/> <input type="radio"/> Screen Failure, Specify: <input type="text"/>

The following MedML file defines report column headings for noncompletion reasons:

```
<?xml version="1.0"?>
<MEDMLDATA xmlns="PhaseForward-MedML-Inform4">

<RESOURCE UUID="PF_SC_REASONCTL_CriteriaElement"
DESCRIPTION="SC Criteria column heading" DATATYPE="TEXT" TEXT="Did Not Meet
Criteria"/>

<RESOURCE UUID="PF_SC_REASONCTL_AEElement"
DESCRIPTION="SC AENonLab column heading" DATATYPE="TEXT" TEXT="Adverse
Experience Non-Laboratory"/>

<RESOURCE UUID="PF_SC_REASONCTL_ALEElement"
DESCRIPTION="SC AELab column heading" DATATYPE="TEXT" TEXT="Adverse
LaboratoryExperience"/>

<RESOURCE UUID="PF_SC_REASONCTL_PDSelValue"
DESCRIPTION="SC PTD column heading" DATATYPE="TEXT" TEXT="Patient Decision"/>

<RESOURCE UUID="PF_SC_REASONCTL_PHDSelValue"
DESCRIPTION="SC PHD column heading" DATATYPE="TEXT" TEXT="Physician Decision"/>

<RESOURCE UUID="PF_SC_REASONCTL_SDSelValue"
DESCRIPTION="SC SD column heading" DATATYPE="TEXT" TEXT="Sponsor Decision"/>

<RESOURCE UUID="PF_SC_REASONCTL_DeathSelValue"
DESCRIPTION="SC Death column heading" DATATYPE="TEXT" TEXT="Death"/>

<RESOURCE UUID="PF_SC_REASONCTL_LostElement"
DESCRIPTION="SC Lost column heading" DATATYPE="TEXT" TEXT="Lost to Follow-Up"/>

<RESOURCE UUID="PF_SC_REASONCTL_OtherSelValue"
DESCRIPTION="SC Other column heading" DATATYPE="TEXT" TEXT="Other"/>

<RESOURCE UUID="PF_SC_REASONCTL_ScrnFailSelValue"
DESCRIPTION="SC Scrn Fail column heading" DATATYPE="TEXT" TEXT="Screen
Failure"/>
</MEDMLDATA>
```

Unscheduled or repeating visits

An unscheduled visit occurs without a specific schedule in relation to other visits. An unscheduled visit can occur one time or it can be a repeating visit, which occurs as often as required.

Repeating visits can be sets of CRFs used for unscheduled visits or can be monitoring forms such as site visit reports or regulatory documentation checklists.

To define an unscheduled visit, create a visit definition for the visit and specify **True** as the value of the UNSCHEDULED attribute. If the unscheduled visit repeats multiple times, include as well the specifications for defining a repeating visit.

To define a repeating visit, create a visit definition with the following specifications:

- The value of the REPEATING attribute is **True**.
- The first form in a repeating visit must not be a common CRF.
- The first form in a repeating visit must contain a Section definition consisting of a single Item containing a DateTimeControl definition that includes the month, day, year, hour, and minute of the visit, of which at least the month and year must be required.

Note: Creating a separate CRF that contains only this DateTimeControl section is recommended.

- The Section, Item, and DateTimeControl definitions must include the following specific required UUIDs:

Component	UUID
DateTimeControl	BD991BC0-B0A4-11D2-80E3-00A0C9AF7674
Item	BD991BBF-B0A4-11D2-80E3-00A0C9AF7674
Section	BD991BBE-B0A4-11D2-80E3-00A0C9AF7674

UUIDs

Universal Unique Identifiers (UUIDs) ensure the unique identification of components across all servers, databases, and trials.

In most cases, assignment of a UUID attribute is unnecessary. However, for certain purposes, the InForm application requires the use of specific, well-known UUIDs. For example, if you want to enable users to define patient numbers, rather than allowing the InForm application to generate them automatically, you must use well-known UUIDs for the visit, form, section, item, and text control definitions that make up the specification of the patient number data entry field.

Note: The InForm Architect application automatically capitalizes UUID strings that contain lowercase characters.

The following components require specific UUIDs:

- *Conflict visit and form* (on page 402)
- *Date Of Visit item* (on page 403)
- *Enrollment form* (on page 403)
- *Patient ID form* (on page 404)
- *Patient Initials item* (on page 404)
- *Patient Number item* (on page 405)
- *Randomization* (on page 405)
- *Regulatory Documents (Reg Docs) form* (on page 405)
- *Repeating visit and reporting DOV* (on page 406)
- *Screening form* (on page 406)
- *Sequences* (on page 407)
- *Study Completion form* (on page 408)
- *Visit Report form* (on page 409)

Additionally, the unit definitions that are included with the Base trial have predefined UUIDs that are not required but are listed in this section for reference.

Conflict visit

The Conflict visit is required if you use the InForm Unplugged software to synchronize data across multiple trial instances. This visit is activated if a study version conflict occurs when users on different instances of a trial start the same patient on different versions of the same CRF, and then synchronize. You must create the Conflict visit in the Dynamic visit node. Use the following UUID:

PF_UUID_CONFLICT_FORMSET

Date Of Visit item

The Date of Visit item appears on the first form of every visit. To set up a scheduled or unscheduled visit, you must use section, item, and date/time control components with the following UUIDs to capture the date and time of the visit. The InForm Architect application supplies these components in the Base trial. When creating the first form of a visit, you can drag the components from the Trial Objects window to the Form window.

Component	UUID
DOV Date/time control	BD991BC0-B0A4-11D2-80E3-00A0C9AF7674
DOV item	BD991BBF-B0A4-11D2-80E3-00A0C9AF7674
DOV section	BD991BBE-B0A4-11D2-80E3-00A0C9AF7674

Enrollment form

When you create an enrollment visit and form, well-known UUIDs are required in the visit and form definitions. The required UUIDs appear in the table that follows. The InForm Architect application automatically provides the UUID properties in the visit and form. When you create an enrollment form section that contains the Patient Number data entry item, you must include the UUID shown for the section.

Note: A trial can have only one Enrollment form.

Component	UUID
Enrollment visit	D882CE3A-0F42-11D2-A419-00A0C963E0AC
Enrollment form	D882CE3B-0F42-11D2-A419-00A0C963E0AC
Enrollment form section that contains the Patient Number data entry item	ABCFA388-223A-11D2-A426-00A0C963E0AC

The Enrollment form can include an editable Patient Number item. The Patient Number item is optional. If you do not include a Patient Number item, the patient number is assigned automatically by the InForm software.

If the Enrollment form includes an editable Patient Number Item, that item requires the following UUIDs.

Component	UUID
Patient Number item	3D25EE4C-7F1B-11D2-A728-00A0C977C64B
Patient number textbox control	433DAFF6-7F1C-11D2-A728-00A0C977C64B

Patient ID form

The Patient ID form enables users to modify the patient number and patient initials that appear in the top right corner of each form. To set up this form, use the following UUIDs:

Component	UUID
Visit containing Patient ID form	03B0D5D8-7F2C-11D2-A728-00A0C977C64B
Patient ID form	06702B62-7ED6-11D2-A728-00A0C977C64B
Patient ID section	3D25EE4B-7F1B-11D2-A728-00A0C977C64B
The Patient Number item used on the Enrollment form	3D25EE4C-7F1B-11D2-A728-00A0C977C64B
Patient Number text box control, if the section contains the Patient Number item used on the Enrollment form	433DAFF6-7F1C-11D2-A728-00A0C977C64B
A Change Initials item	D959FE72-7F1C-11D2-A728-00A0C977C64B
Change Initials text box control, if the section contains a Change Initials item	433DAFF7-7F1C-11D2-A728-00A0C977C64B

Patient Initials item

The Patient Initials Item definition is required on the Screening form, along with the following related items and controls. Use the following UUIDs:

Component	UUID
Patient Initials item	AEB64F16-127C-11D2-A41C-00A0C963E0AC
Patient Initials textbox control	AEB64F17-127C-11D2-A41C-00A0C963E0AC

Patient Number item

The Patient Number Item definition is an optional item on the Enrollment form. If you allow users to edit the Patient Number, use the following required UUIDs to define the Patient Number item:

Component	UUID
Patient Number item	3D25EE4C-7F1B-11D2-A728-00A0C977C64B
Patient number textbox control	433DAFF6-7F1C-11D2-A728-00A0C977C64B

Randomization

When setting up the InForm application to generate randomization numbers, you must include a well-known control, item, and section on the form where the randomization number appears. These components are included in the Base trial and use the following UUIDs:

Component	UUID
Randomization calculated control	DC2EB0BF-4F12-11D2-9319-00A0C9769A13
DRUGKIT item	52AF1207-4F13-11D2-9319-00A0C9769A13
DRUGKITSECTION section	C0482B37-4F13-11D2-9319-00A0C9769A13

Regulatory Documents (Reg Docs) form

The Regulatory Documents (Reg Docs) form is optional in a trial. If you include a Reg Docs form, it must:

- Occur only once in the trial.
- Use required UUIDs for visit, form, and section.
- Include at least one item.

The InForm Architect application automatically provides the UUID properties in the visit and form. UUIDs for Reg Docs visit, form, and section appear in the following table:

Component	UUID
Reg Docs visit	PF_UUID_REGDOCS_FORMSET
Reg Docs form	PF_UUID_REGDOCS_FORM

Component	UUID
Reg docs section	PF_UUID_REGDOCS_SECTION

Repeating visit and reporting DOV

To set up a repeating visit, you must use the following UUIDs to create a section, item, and date/time control in which to capture the date and time of the visit (DOV).

These specific components are also required for accurate reporting in the form aging and cycle time reports, as well as the legacy ASP CRF Performance Report.

For more information, see *Date of Visit* (on page 378). The following table shows the required DOV UUIDs for repeating visits and for reporting:

Component	UUID
DateTimeControl	BD991BC0-B0A4-11D2-80E3-00A0C9AF7674
Item	BD991BBF-B0A4-11D2-80E3-00A0C9AF7674
Section	BD991BBE-B0A4-11D2-80E3-00A0C9AF7674

Screening form

The definition of a Screening form requires the UUIDs shown in the following table. The InForm Architect application automatically provides the UUID properties in the visit and form. A trial can have only one Screening form.

Component	UUID
Screening visit	D882CE38-0F42-11D2-A419-00A0C963E0AC
Screening form	D882CE39-0F42-11D2-A419-00A0C963E0AC

If you design a Screening form to include Patient Initials, additional UUIDs are required for the section, item, control, and related items. For more information, see *Screening section with Patient Initials* (on page 407).

Screening section with Patient Initials

When you define the properties for the section of the screening form that contains the Patient Initials and related items, you must include the following UUIDs:

- Screening form section:

Component	UUID
Screening form section that contains the Patient Initials, Date of Birth, and Date Screened items	96CAE354-126C-11D2-A41C-00A0C963E0AC

- Patient Initials item and control:

Component	UUID
Patient Initials item	AEB64F16-127C-11D2-A41C-00A0C963E0AC
Patient Initials textbox control	AEB64F17-127C-11D2-A41C-00A0C963E0AC

- Related items and controls:

Component	UUID
Date of Birth item	96CAE359-126C-11D2-A41C-00A0C963E0AC
Date of Birth control	40AEE712-217C-11D2-A425-00A0C963E0AC
Date Screened item	96CAE356-126C-11D2-A41C-00A0C963E0AC
Date Screened control	96CAE357-126C-11D2-A41C-00A0C963E0AC

Sequences

The InForm application automatically generates enrollment, query, randomization, and screening numbers as a trial progresses. These numbers are generated according to a sequence established in a Sequence definition and loaded into the database in the Base trial. The required UUIDs define the enrollment, query, and screening number sequences:

Component	UUID
Query	5B7D4EB4-0465-11D2-A414-00A0C963E0AC
Enrollment	EB75B898-078B-11D2-A417-00A0C963E0AC
Screening	F7F1B3B8-0B5C-11D2-A418-00A0C963E0AC

Study Completion form

The Study Completion form records the last time the patient was seen and whether the patient completed the trial. If the patient did not complete the trial, the Study Completion form records the reason for not completing. Use the following UUIDs in the definition of a Study Completion form.

Note: The trial must contain a Study Completion form for the standard Patient Status and Patient Dropout reports to show patient status correctly.

The UUIDs in the Study Completion form are independent of study versions and apply to all patients in a trial; therefore, you cannot change study completion metadata by creating a new study version.

The following table shows the required components for defining a Study Completion form:

Component	UUID
Visit in which the Study Completion form is included	F4699051-69E2-11D2-8FB5-00A0C977C66A
Study Completion form	7314A6A5-316E-11D2-8F9A-00A0C977C66A
Control to indicate completion status	PF_SC_COMPLETECTL
Element objects defining the values and display text for study completion or noncompletion	<ul style="list-style-type: none"> • PF_SC_STUDY_COMPLETE — Indicates that the patient completed the trial. • PF_SC_STUDY_INCOMPLETE — Indicates that the patient did not complete the trial.
Control to indicate the reason the patient dropped out of the trial	PF_SC_REASONCTL
Text resources mapping the internal values of noncompletion reasons to the text of column headings on the standard and ad hoc patient dropout reports	<p>PF_SC_REASONCTL_ <i>internal_reason_value</i></p> <p><i>Internal_reason_value</i> is one of the following:</p> <ul style="list-style-type: none"> • The Value property defined for an element object in a simple or pull-down control included in the control specifying noncompletion reason. • The Selection Value property defined for any other type of subordinate control included in the control specifying noncompletion reason. • The default value for a subordinate control for which no Selection Value property has been defined. The default value is assigned by InForm application and is in the format !pf! <i>control_DBUID_path</i>.

Visit Report form

Required UUIDs for Visit Report components appear in the following table:

Component	UUID
Visit Report visit	PF_UUID_VISITREPORT_FORMSET
Visit Report form	PF_UUID_VISITREPORT_FORM
Visit Report section	BD991BBE-B0A4-11D2-80E3-00A0C9AF7674
Visit Report item	BD991BBF-B0A4-11D2-80E3-00A0C9AF7674
Visit Report date/time control	BD991BC0-B0A4-11D2-80E3-00A0C9AF7674

Unit UUIDs

The Base trial includes unit definitions that include the UUIDs listed in the following table. These UUIDs are not required but are provided for future product enhancements:

Unit RefName	UUID
BPDIAS	PF_BP_DIAS
BPSYS	PF_BP_SYS
Celsius	498100FF-E9DF-11D1-9E60-00A0C9769A33
Centimeter	498100F5-E9DF-11D1-9E60-00A0C9769A33
Fahrenheit	498100FE-E9DF-11D1-9E60-00A0C9769A33
Inches	498100F2-E9DF-11D1-9E60-00A0C9769A33
Kilogram	498100F8-E9DF-11D1-9E60-00A0C9769A33
Pound	498100FC-E9DF-11D1-9E60-00A0C9769A33

CHAPTER 10

Customizing predefined lists

In this chapter

Overview of customizing predefined lists	412
Customizing predefined lists.....	413

Overview of customizing predefined lists

Several predefined lists are included with every InForm installation. These lists include default text for things such as rights, comments, query responses, and so on.

Most predefined lists **must not** be manipulated directly, but the InForm application does allow you to make changes to the default text displayed in pulldown controls for:

- Reasons for change.
- Query responses (**not** query states).
- Reasons for clearing a CRF.
- Reasons for editing an item.

Customizing predefined lists

The only predefined lists that you can edit are in the following locations of your InForm application installation directory:

- XMLBase\sysform_ClearCRF.xml
- XMLBase\sysform_EditItem.xml
- XMLBase\sysform_ItemEditReasons.xml
- Resources\pfscrip.js

Do not change any other files in these directories.

Note: Take care when making changes to predefined lists, and be sure to make changes only to the files discussed in this chapter. Changing other files can result in serious problems in the trial data.

Changing reason text for clearing a CRF or editing an item

You can add new text values to predefined lists, and you can remove existing values or replace them with new values by changing the definitions that InForm application references. You cannot physically delete an existing text definition in a predefined list (in other words, **do not** delete the PFELEMENT definitions of the existing values).

To change a reason text value:

- 1 Open the appropriate file for the list you want to edit.
 - **sysform_ClearCRF.xml**—Contains the list of reasons for clearing a CRF
 - **sysform_EditItem.xml**—Contains the list of reasons for editing a CRF item
- 2 Create a new PFELEMENT definition for the new text you want to add. For example,

```
<PFELEMENT REFNAME="CLEARCRF_SEVTRANSCRERROR"
LABEL="Severe Transcription Error"
TYPE="STRING"
VALUE="Severe Transcription Error"/>
```

- 3 Edit the list of ELEMENTREF definitions in the appropriate pulldown control definition:
 - To add a new text value, add a new ELEMENTREF definition that uses the REFNAME value you defined in step 2. Be sure to specify the order in which you want the new text to appear in the pulldown control. For example, add a new selection for a Severe Transcription Error in the second position in the pulldown control:

```
<PULLDOWNCONTROL REFNAME="CLEARCRF_CHANGEREASONPULLDOWN"
UUID="730C92AD-33AE-11D3-8D93-00902757C687"
NAME="CLEARCRF_CHANGEREASONPULLDOWN">
<ELEMENTREF REFNAME="CLEARCRF_TRANSCRERROR" ORDER="1"/>
<ELEMENTREF REFNAME="CLEARCRF_SEVTRANSCRERROR"
ORDER="2"/>
<ELEMENTREF REFNAME="CLEARCRF_NEWINFO" ORDER="3"/>
<ELEMENTREF REFNAME="CLEARCRF_CHANGEDINFO" ORDER="4"/>
</PULLDOWNCONTROL>
```

- To replace an existing text value with a new one, edit the ELEMENTREF definition that refers to the old value by changing the REFNAME attribute to refer to the new value you defined in step 2. For example, to substitute the new text for a Severe Transcription Error in place of the original “Transcription Error,” you would change the XML to look like this:

```
<PULLDOWNCONTROL REFNAME="CLEARCRF_CHANGEREASONPULLDOWN"
UUID="730C92AD-33AE-11D3-8D93-00902757C687"
NAME="CLEARCRF_CHANGEREASONPULLDOWN">
<ELEMENTREF REFNAME="CLEARCRF_SEVTRANSCRERROR"
ORDER="1"/>
<ELEMENTREF REFNAME="CLEARCRF_NEWINFO" ORDER="2"/>
<ELEMENTREF REFNAME="CLEARCRF_CHANGEDINFO" ORDER="3"/>
</PULLDOWNCONTROL>
```

- To remove an existing text value, delete the ELEMENTREF definition that refers to it. Update the ORDER attributes of the remaining ELEMENTREF definitions to reflect the change. For example, to cause the InForm application to ignore both transcription error messages, change the XML to look like this:

```
<PULLDOWNCONTROL REFNAME="CLEARCRF_CHANGEREASONPULLDOWN"
UUID="730C92AD-33AE-11D3-8D93-00902757C687"
NAME="CLEARCRF_CHANGEREASONPULLDOWN">
<ELEMENTREF REFNAME="CLEARCRF_NEWINFO" ORDER="1"/>
<ELEMENTREF REFNAME="CLEARCRF_CHANGEDINFO" ORDER="2"/>
</PULLDOWNCONTROL>
```

Note: Take care not to remove or replace any ELEMENTREF definitions for query states.

- Save your changes.
- Install the updated system forms by using the MedML Installer tool.

Changing query text

You can change the text of standard query reasons and responses by editing the \Resources\pfscrip.js file in the InForm application installation tree. Each set of strings, prefixed by a number in brackets, appears under different conditions in an InForm trial. For more information, see *Types of query text strings* (on page 415).

To change query text:

- In the QO section of the pfscrip.js file, add, edit, or remove query text strings as desired. Separate response strings with a comma and surround each string with double quotation marks. The QO section of the pfscrip.js file follows:

```
QO=[[ [1,2],["Item incomplete","Data does not match source","Missing units","Make
text more specific"],
      [7],["Original value is correct","Changed data per query","Query is
ambiguous","Data is not in patient record","Measurement skipped on this
visit"],
      [3],["Response from site required"],
      [4],["Query is invalid or does not apply","Query can be addressed
internally"],
      [5,6],["Response does not satisfy query"],
      [8],["Response satisfies query"],
      [9],["State chosen is acceptable"]]]
```

2 Use the following XML to install the updated pfscrip.js file with MedML Installer:

```
<RESOURCEDATA xmlns="PhaseForward-MedML-Inform4">
<RESOURCE
  FILENAME="pfscrip.js"
  UUID="PF_RESOURCE_GENERIC_SCRIPT_MAIN"
  DESCRIPTION="Main Script"
  DATATYPE="SCRIPT"
  LANGUAGE="English"/>
<HTMLTEMPLATE
  TEMPLATENAME="PF_SCRIPT_MAIN"
  BROWSERNAME="GENERIC"
  RESOURCEUUID="PF_RESOURCE_GENERIC_SCRIPT_MAIN"/>
</RESOURCEDATA>
```

Types of query text strings

The following table lists the types of query text strings included in the pfscrip.js file:

Identifier	Default strings	When used
[1,2]	["Item incomplete","Data does not match source","Missing units","Make text more specific"]	Creating a query
[3]	["Response from site required"]	Placing a candidate query in open state
[4]	["Query is invalid or does not apply","Query can be addressed internally"]	Deleting a query
[5,6]	["Response does not satisfy query"]	Reissuing a query
[7]	["Original value is correct","Changed data per query","Query is ambiguous","Data is not patient record","Measurement skipped on this visit"]	Responding to a query
[8]	["Response satisfies query"]	Closing a query
[9]	["State chosen is acceptable"]	Resolving a conflict

CHAPTER 11

Performing trial administration activities

In this chapter

Overview	418
About InForm application administrators	419
About administrative data	420
Using the Admin tree in the Trial Objects window	427
Managing users	429
Managing rights groups	438
Managing groups	443
Managing sites	448
Defining sponsors	451
Managing numbering	452
Defining system configuration settings	456

Overview

The trial administration feature of the InForm Architect application enables you to maintain administrative trial data much as you would in the InForm application, but without leaving the InForm Architect application. When you use the trial administration feature, the InForm Architect application opens a browser view inside the Design Workspace and displays pages that are the same as the screens accessed through the Admin button in the InForm application.

This chapter provides an overview of administrative data and describes how to maintain it by using the InForm Architect application or by manually specifying the appropriate MedML tags.

About InForm application administrators

The InForm application identifies the following types of administrators. The table below describes each type:

Administrator	Administrator Tool	Purpose
System Administrator	InForm application, MedML and MedML Installer utility	Setting system properties during trial setup
Application Engineer	InForm Architect application	Testing during trial development
Trial Administrator	InForm application	Maintaining security policy during the life of the trial

Note: This chapter describes the administrative functions you can perform through the InForm Architect application only.

About administrative data

Administrative data specifies who is running a trial, how the workload is distributed, and how the trial is configured. The following table lists the administrative data definitions that you can maintain and the tools you can use:

Administrative Data	Available through...		
	InForm Architect application	InForm application	MedML
Users	▪	▪	•
Rights	▪	▪	•
Groups	▪	▪	•
Sites	▪	▪	•
Events	▪	▪	•
Rules	▪	▪	•
Configuration		▪	•
System information		▪	•
Sponsors			•
Sequence numbers			•

Users

User definitions specify name, address, and contact information for an InForm user, as well as:

- The name of an image file that displays a user picture when the user logs on to the InForm application.
- The default home screen URL.
- The initial status, Active or Inactive.

Rights

Rights are predefined privileges to perform specific tasks in the InForm application. Multiple rights are organized into Rights Groups, which are in turn associated with users. The grouping of rights into Rights Groups and the association of users with Rights Groups determines how the workflow is distributed in a trial.

Trial activities and rights

The following table summarizes the rights and other authorizations that a user needs to perform typical trial activities. For information about the rights required to perform administrative and user management activities, including the rights to manage data validation checks, see the system administration overview.

To perform this activity	A user must have these rights
Miscellaneous user rights:	
Print a CRF	Print
Create and edit Frequently Asked Questions (FAQs) on CRF items	Edit Study FAQ
Create Visit Reports	Monitor
Create records of regulatory document reviews	Monitor
Run reports with the InForm Reporting and Analysis module	Reports
Screen and enroll trial candidates	Enroll Patients Enter Data into a CRF View CRF
View Screening Log (read-only)	Enroll or Monitor, with no CRF rights
Display Required Signatures page and see CRF signing status for all signature groups (read-only)	View CRF Signature Information
Display Required Signatures page and see Case Book signing status for all signature groups (read-only)	View Case Book Signature Information
Export listings of CRF data to an external data source	Data Export Listings
Transfer patient records from one site to another	Change site for patient
CRF rights:	
Enter CRF data	Enter Data into a CRF View CRF

Edit CRF data	Edit Data on a CRF View CRF
Enter comments on a CRF item or on a whole CRF	Enter Comments into a CRF View CRF
Freeze a CRF	Freeze a CRF View CRF
Unfreeze a CRF	Unfreeze a CRF View CRF
Lock a CRF	Lock a CRF View CRF
Unlock a CRF	Unlock a CRF View CRF
Indicate a CRF is ready for source verification, or indicate that a CRF that was ready for source verification is no longer ready	Mark a CRF as Ready for SV View CRF
Perform source verification on CRF data	Mark and Unmark a CRF as SV'd View CRF To view the Source Verification Listing, the Monitor right is also required.
Sign a CRF	Sign a CRF View CRF Additionally, the user must be in the same signature group as the CRF to sign.
Case Book rights:	
Sign a Case Book	Sign a Case Book View CRF Additionally, the user must be in the same signature group as the CRF designated as the signing CRF for the Case Book.
Indicate a Case Book is ready for source verification, or indicate that a Case Book that was ready for source verification is no longer ready	Mark and Unmark a Case Book as Ready for SV View CRF
Freeze a Case Book	Freeze a Case Book View CRF
Unfreeze a Case Book	Unfreeze a Case Book View CRF
Lock a Case Book	Lock a Case Book View CRF
Unlock a Case Book	Unlock a Case Book View CRF

Query rights:	
Open a Candidate query so that it becomes visible to the site	<p>Change Query State from Candidate to Open View CRF</p> <p>Additionally, the user must be defined as a Sponsor user.</p>
Delete a Candidate query	<p>Change Query State from Candidate to Deleted View CRF</p> <p>Additionally, the user must be defined as a Sponsor user and must be a member of the same query group as the user who created the Candidate query.</p>
Close an answered query	<p>Change Query State from Answered to Closed View CRF</p> <p>Additionally, the user must be a member of a query group.</p>
Close a query in the Opened state	<p>Change Query State from Open to Closed View CRF</p> <p>Additionally, the user must be defined as a Sponsor user and be a member of the same query group as the user who opened the query</p>
Close a query that has been reissued in the Candidate state	<p>Change Query State from Reissued Candidate to Closed View CRF</p> <p>Additionally, the user must be defined as a Sponsor user and be a member of the same query group as the user who reissued the query</p>
Answer a query	<p>Answer Query View CRF</p>
Enter a query in the Candidate state so that it is not visible to the site	<p>Enter Query in Candidate State View CRF</p> <p>Additionally, the user must be defined as a Sponsor user</p>
Enter a query in the Opened state so that it is visible to the site	<p>Enter Query in Open State View CRF</p>
Reissue an answered query in the Candidate state so that it is not visible to the site	<p>Re-issue Query in Candidate State Enter Query in Candidate State View CRF</p> <p>Additionally, the user must be defined as a Sponsor user and must be a member of the same query group as the user who created the query.</p>

Reissue an answered query in the Opened state so that it is visible to the site	<p>Re-issue Query in Open State Enter Query in Open State View CRF</p> <p>Additionally, the user must be a member of the same query group as the user who originally created the query.</p>
Resolve a query that is in a conflict state because multiple users using different copies of a trial acted on it before synchronizing	<p>Resolve Query in Conflict View CRF</p> <p>Additionally, to resolve a Site Conflict, a user must be defined as a Site user. To resolve a Sponsor Conflict, a user must be defined as a Sponsor user.</p>
Rule rights:	
Cancel an active rule	Deactivate a rule
Run an existing rule	Run a rule
Edit an existing rule	Modify a rule
Navigation rights:	
Move through a trial by visit across patients at the same site.	Navigate by Visit View CRF
Move through a trial by form across patients at the same site.	Navigate by Form View CRF
Place patients (at a site) in a particular order for more efficient data entry.	Reordering Of Patients View CRF
System Admin rights:	
Set remote update options	Manage Software Patch
View status of patch installation and distribution; apply a stored patch to a remote server	View Software Patch Status
Portal rights:	
Configure global settings for the InForm Portal application, and create and manage Portal Administrators (InForm Portal users with certain administrator rights).	InForm Portal Config Admin

Groups

Users are associated with the definitions of the following types of groups, which determine the types of activities to which the users have access:

- **Query Group**—Gives a user who has been assigned the right to close queries the additional right to close queries initiated by another member of the same Query Group.
- **Signature Group**—Gives a user the right to sign documents requiring signature. To sign a site's documents, a user must be in a Signature Group and the appropriate Site Group.
- **Item Group**—Identifies a list of items appearing on forms whose display override characteristics can be changed through a rights group.
- **Manager User Group**—Defines a set of users whose activities are included in the CRA versions of legacy ASP reports delivered with the InForm application. A user who is a designated manager of a Manager User group can view the CRA versions of reports.
- **Reporting Group**—Identifies the types of reports group members can generate

Sites

Site definitions specify address and contact information about a trial location. Sites are associated with specific users, and this association determines the data to which users have access. For example, a CRC who works at only one site obtains access to the data for that site by being associated with the site definition. A CRA who travels to multiple sites might be associated with multiple site definitions and thus obtain access to data for all of those sites.

Events

Event definitions specify the query text and initial query status for events in the InForm application. You can update both the query text and initial status through the administration tool.

Rules

Rule definitions specify every aspect of rules that are triggered by events in the InForm application. You can update the following parameters through the administration tool:

- **Rule description**—Descriptive rule name.
- **Status**—Status of the rule: Active or Inactive. Only rules in Active status fire when the form is submitted.
- **Help Text**—User-friendly description of the edit check, calculation, conversion, or randomization performed by the rule. This text appears along with CRF Help for the data items with which the rule is associated.
- **Rule Script Text**—Text of the script that runs when the rule executes.

Configuration information

Configuration definitions specify system configuration settings, which determine various InForm application behaviors at the trial level. For example, configuration settings specify security policies such as the number of minutes of inactivity that can pass before the InForm application requires a user to log in again, the number of days that can pass before the InForm application requires users to change their passwords, and the number of minutes that a session can be active before the InForm application requires a user to log in again.

System

The System Information View screen summarizes system information about the current installation of the InForm application, including version numbers and DLL names. The information is divided into four sections:

- **InForm**—Displays the version numbers of the InForm application and the database schema of the InForm application database.
- **Database**—Displays the database name and version number and the name of the database driver.
- **Webserver**—Displays the version numbers of the Microsoft Windows operating system and of the Microsoft Internet Information Server (IIS) running on the server.
- **Components**—Lists the InForm application DLLs.

Sponsors

Sponsor definitions specify address and contact information about a sponsor. This information is for documentation and reporting purposes only. Sponsor properties do not appear in the InForm application user interface.

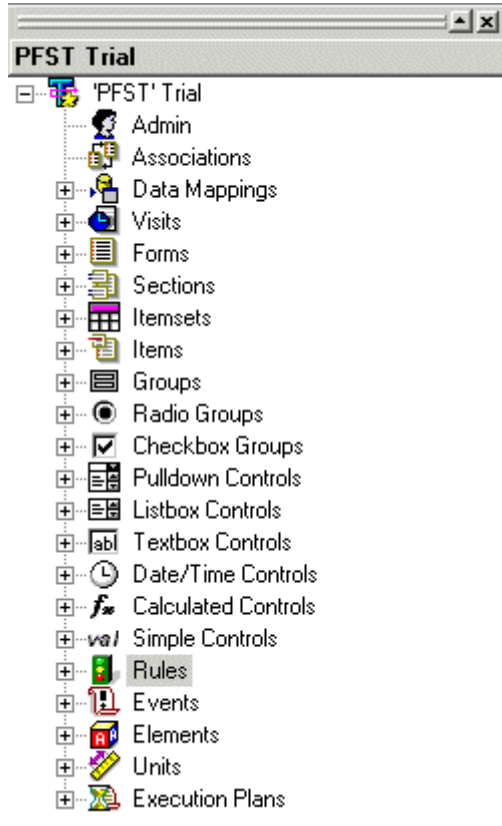
Sequence numbers

The InForm application generates numbers for tracking screened patients, enrolled patients, and randomization drug kits. As new patients are added to the system or randomized, the InForm application automatically assigns sequence numbers by using a numbering scheme that you can specify.

The InForm application includes default sequence number formats for screening and enrollment numbers, and configuration of these types of numbers is optional. If you configure the InForm application to support randomization, you must specify the sequence number format. For information on how to specify sequence number format, see *Managing numbering* (on page 452).

Using the Admin tree in the Trial Objects window

The Trial Objects window of the InForm Architect application includes an Admin node that provides access to the screens used for performing administrative activities that affect users, groups, rights, and sites:



To display the home screen used for maintaining administrative data, click the **Admin** icon. The InForm application Users page appears, providing access to all InForm application administration screens.

Users						
X	Last Name ▲	First Name	Account Name	Status	Personnel Type	User Group Properties
<input type="checkbox"/>	Admin	Trial	admin	Active	SPONSOR	Properties
<input type="checkbox"/>	Admin	System	sysadmin	Active	SYSTEM	Properties
<input type="checkbox"/>	Administrator	User	UA	Active	SPONSOR	Properties
<input type="checkbox"/>	Allen (CRA)	Tom	tallen	Active	SPONSOR	Properties
<input type="checkbox"/>	Coding	Clintrial	ctcoding	Active	SYSTEM	Properties
<input type="checkbox"/>	Collins (CRC)	Ann	acollins	Active	SITE	Properties
<input type="checkbox"/>	Conrad (CDM)	Bill	bconrad	Active	SPONSOR	Properties
<input type="checkbox"/>	D'Amico (CDM)	Robert	radmico	Active	SPONSOR	Properties
<input type="checkbox"/>	InForm	User	InFormUser	Active	SYSTEM	Properties
<input type="checkbox"/>	Jackson (CRC)	John	jjackson	Active	SITE	Properties
<input type="checkbox"/>	Jones	Peter	pijones	Active	SPONSOR	Properties
<input type="checkbox"/>	Lam (PM)	Gil	pm	Active	SPONSOR	Properties
<input type="checkbox"/>	Larkin (PI)	Abby	alarkin	Active	SITE	Properties
<input type="checkbox"/>	Manager	User	UM	Active	SPONSOR	Properties
<input type="checkbox"/>	Merkel	Susan	sumerkel	Active	SPONSOR	Properties

Page 1 of 3

Managing users

The InForm Architect application provides the same user administration features as the Admin functionality of the InForm application. This section describes how to use the InForm Architect application to:

- Create a user.
- Add or update a user password.
- Assign a user to a group or remove a user from a group.
- Assign a user to a site.
- Assign rights to a user or remove a user from a rights group.
- Activate, deactivate, terminate, or reinstate a user.

Creating a user

To create a new user:

- 1 In the **Trial Objects** window, click the **Admin** icon.

X	Last Name	First Name	Account Name	Status	Personnel Type	User Group Properties
<input type="checkbox"/>	Admin	Trial	admin	Active	SPONSOR	Properties
<input type="checkbox"/>	Admin	System	sysadmin	Active	SYSTEM	Properties
<input type="checkbox"/>	Administrator	User	UA	Active	SPONSOR	Properties
<input type="checkbox"/>	Allen (CRA)	Tom	tallen	Active	SPONSOR	Properties
<input type="checkbox"/>	Coding	Clintrial	ctcoding	Active	SYSTEM	Properties
<input type="checkbox"/>	Collins (CRC)	Ann	acollins	Active	SITE	Properties
<input type="checkbox"/>	Conrad (CDM)	Bill	bconrad	Active	SPONSOR	Properties
<input type="checkbox"/>	D'Amico (CDM)	Robert	radmico	Active	SPONSOR	Properties
<input type="checkbox"/>	InForm	User	InFormUser	Active	SYSTEM	Properties
<input type="checkbox"/>	Jackson (CRC)	John	jjackson	Active	SITE	Properties
<input type="checkbox"/>	Jones	Peter	piijones	Active	SPONSOR	Properties
<input type="checkbox"/>	Lam (PM)	Gil	plam	Active	SPONSOR	Properties
<input type="checkbox"/>	Larkin (PI)	Abby	alarkin	Active	SITE	Properties
<input type="checkbox"/>	Manager	User	UM	Active	SPONSOR	Properties
<input type="checkbox"/>	Merkel	Susan	sumerkel	Active	SPONSOR	Properties

- 2 Click the **Add User** button.

3 On the new user page, enter the following information about the user:

- User account name, used to log in to the InForm application.

Note: You cannot create a new user name if the only difference in the new one is the case of the characters (for example, Jdoe and jdoe).

- First and last name.
- User type (site or sponsor). User type is significant for query activities: queries in Candidate status are invisible to Site users but are visible to Sponsor users.
- Address, including street address, city, state or province, country, and postal code.
- Contact data including phone number, alternate phone number, beeper number, fax number, and email address.
- Information used for custom online display:

Display Name—User name as it appears under the picture of the user in the navigation panel.

Home Page—URL of the page that appears when the user logs in to the InForm application. This URL can be any valid URL pointing to an internal or external site identified by an IP address or host name. The URL must specify the full path name, beginning with http://

Date Format—The format of dates as displayed on the InForm application screens: Month/Day/Year, Day/Month/Year, or Year/Month/Day.

- 4 Indicate the user's status by selecting the **User Active** check box to make the user active or leaving it unselected to make the user inactive.

Note: If you intend to assign a password to the user immediately, leave the status inactive; you can only create a password for an inactive user.

- 5 Click **Submit**.

Assigning or updating a user password

After a user is created, the Change Password button is available on the User page, and you can assign the user an initial password. After the user logs on for the first time, he or she is required to change this initial password. To assign or update a user's password:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 If the user's status, as shown in the **Status** column, is Active, change the status to Inactive:
 - a Select the user whose password you want to create or update by clicking the check box in the X column.
 - b Click the **Deactivate** button.
 - c Clear the confirmation message that appears by clicking **OK**.
- 3 Click the name of the user.

The screenshot displays the 'Users' management page in InForm Architect. The breadcrumb trail at the top includes: Users | Rights | Groups | Sites | Configuration | Events | Rules | Patch Admin | Patch Status | System | Synchronization. The main form is titled 'Users' and contains the following sections:

- Basic User Info:**
 - 1. User Name: acollins
 - 2. First Name: Ann
 - 3. Last Name: Collins (CRC)
 - 4. User Type: Site User Sponsor User
 - 5. User Active: User Active
 - 6. Title: [Empty field]
 - 7. Description: [Empty field]
 - 8. User must change password at next logon: Yes No
- User Address:**
 - 9. Street: [Empty field]
 - 10. Street2: [Empty field]
 - 11. City: [Empty field]
 - 12. State/Province: [Empty field]
 - 13. Country: [Empty field]
 - 14. Postal Code: [Empty field]
- User Contact:** [Empty field]

At the bottom of the form, there is a 'Change Password' button on the left, and 'Submit' and 'Return' buttons on the right.

- 4 Click the **Create Password** button (for a new user) or **Change Password** button (for an existing user).
The **Password** page appears.
- 5 Enter a password for the user in the **New password** box.
- 6 Re-enter the password for confirmation in the **Confirm new password** box.
- 7 Click **Submit**.

Conforming to password standards

When you set trial configuration options, as described in *Defining system configuration settings* (on page 456), you specify the following options pertaining to password standards:

- Minimum password length.
- Number of failed logon attempts before the user's password is inactivated.
- Requirement of at least one numerical character.
- Requirement of at least one uppercase character.
- Requirement of at least one nonalphanumeric character.
- Permission to reuse a password.

When you create a new password or modify an existing password, make sure that the new password you assign conforms to the standards that have been set for your trial.

Note: You cannot use a left angle bracket (<) or ampersand (&) in user passwords.

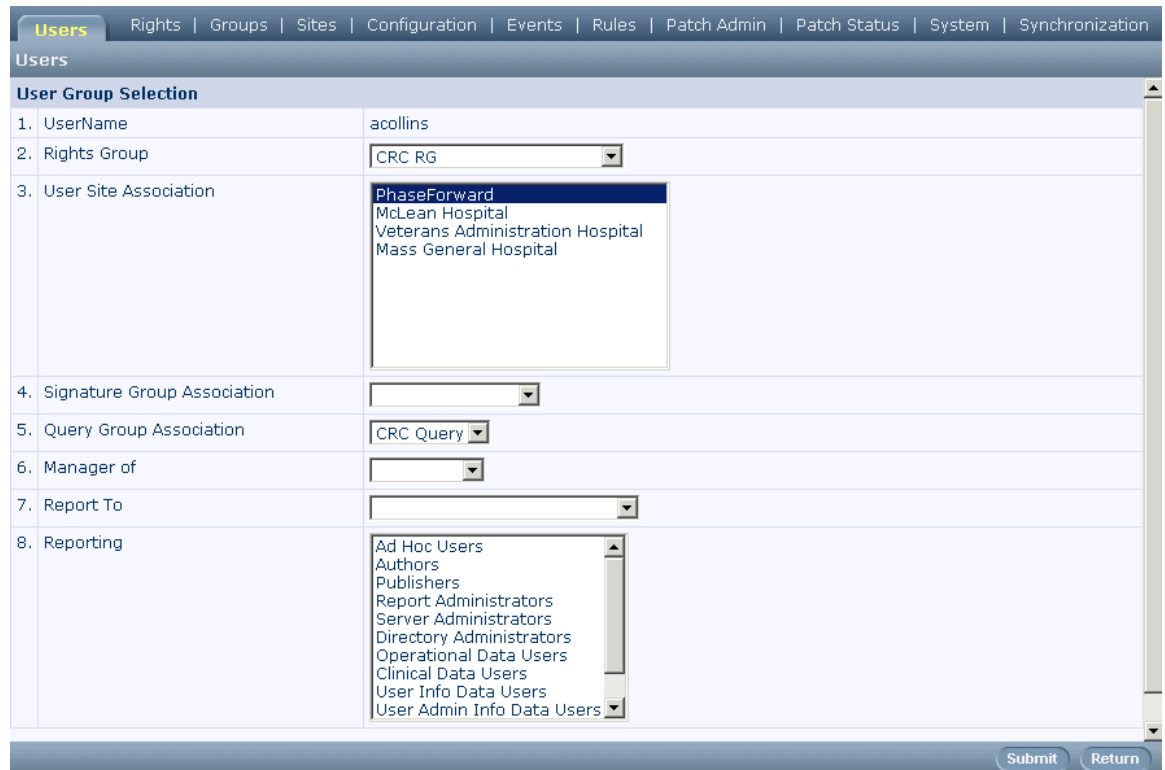
Managing group assignments

You can add a user to a query group, signature group, or reports group.

Adding a user to a group

After a user is created, you can assign the user to a query group, signature group, or reports group:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 In the **User Group Properties** column, click the **Properties** link for the user you want to assign to groups.



- On the **User Group Selection** page, select the groups to which you want to assign the user:

Task	Action
Assign the user to a signature group	Select the group from the Signature Group Association list.
Assign the user to a query group	Select the group from the Query Group Association list.
Specify the types of reports the user can generate	Select the report types from the Reporting list.

- Click **Submit**.

Removing a user from a group

To remove a user from a group:

Note: You can only remove a user from a group by using the InForm application user interface or the Admin user interface in the InForm Architect application. You cannot remove a user by running the MedML Installer.

- In the **Trial Objects** window, click the **Admin** icon.
- Click the **Groups** tab.

Users Rights Groups Sites Configuration Events Rules Patch Admin Patch Status System Synchronization				
Groups				
Group Name ▲	Group Type	Member Count	Description	Members ▲
Ad Hoc Users	Reporting	0	Users who have access to Ad Hoc Reporting	Change
Authors	Reporting	0	Users who have access to Ad Hoc Reporting and Report Studio	Change
Clinical Data Users	Reporting	0	Users who have access to clinical data	Change
CRA Query	Query	9	CRA Query	Change
CRA Sigs	Signature	3	CRA Sigs	Change
CRC Query	Query	3	CRC Query	Change
Directory Administrators	Reporting	0	Users who can administer authorization rights. They administer groups, accounts, contacts, distribution lists, data sources, and printers	Change
Investigator Sigs	Signature	2	Investigator Sigs	Change
LabQuestion	ItemGroup	1	Clinical significance question	Change
Operational Data Users	Reporting	0	Users who have access to operational data (including queries and comments) except user data	Change
Publishers	Reporting	0	Users who can create folders and other content within the Public Folder space	Change
Report Administrators	Reporting	0	Users who can administer authorized public content such as setting permissions on folders and reports	Change
ReportList	ManagerUser	0	CRA List for Report Viewing	Change
Server Administrators	Reporting	0	Users who can administer servers, dispatchers, and jobs	Change
Site Users	Reporting	9	Site Users	Change

Add Group Page 1 of 2

- In the **Members** column, click the **Change** link for the group from which you want to remove the user.

Users Rights Groups Sites Configuration Events Rules Patch Admin Patch Status System Synchronization	
Groups	
Change Membership of Group: CRA Query	
autoquery tallen mm janwilson mroberts memiller bjsmith sumerkel pjones	system UM UA InFormUser pfarchuser sysadmin pfreportuser ctvalidation ctcoding amichelle esnow radmico pm ss js
<input type="button" value=" << Add << "/> &br/> <input type="button" value=" >> Remove >> "/> &	
<input type="button" value="Submit"/> <input type="button" value="Return"/>	

- On the **Change Membership of Group** page, select the user's name in the group members list on the left, and then click the **Remove** button. To select more than one user at a time, hold down the **Ctrl** key while selecting each name.
- Click **Submit**.

Managing site assignments

To manage site assignments:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 In the **User Group Properties** column, click the **Properties** link for the user you want to assign to one or more sites.

The screenshot shows the 'Users' management interface. At the top, there is a navigation bar with tabs: Users, Rights, Groups, Sites, Configuration, Events, Rules, Patch Admin, Patch Status, System, and Synchronization. Below this is the 'User Group Selection' form. The form contains the following fields:

1. Username	acollins
2. Rights Group	CRC RG
3. User Site Association	PhaseForward McLean Hospital Veterans Administration Hospital Mass General Hospital
4. Signature Group Association	
5. Query Group Association	CRC Query
6. Manager of	
7. Report To	
8. Reporting	Ad Hoc Users Authors Publishers Report Administrators Server Administrators Directory Administrators Operational Data Users Clinical Data Users User Info Data Users User Admin Info Data Users

At the bottom right of the form, there are two buttons: 'Submit' and 'Return'.

- 3 In the **User Site Association** list on the **User Group Selection** page, select the sites to which you want the user to have access. To select more than one site, hold down the **Ctrl** key while selecting each site.
- 4 Click **Submit**.

Assigning rights to a user

After a user is created, you can assign rights to the user by associating the user with a rights group. A user can have only one rights group. To assign rights to a user:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 In the **User Group Properties** column, click the **Properties** link for the user you want to assign to a rights group.
- 3 In the **Rights Group** list on the **User Group Selection** page, select the rights group to which you want to assign the user.
- 4 Click **Submit**.

Removing rights from a user

To remove rights from a user:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 Click the **Rights** tab.
- 3 In the **List Users** column, click the **Change Users** link for the rights group from which you want to remove the user.
- 4 On the **Change Members in Rights Group** page, select the user's name in the rights group members list on the left, and then click the **Remove** button. To select more than one user at a time, hold down the **Ctrl** key while selecting each name.
- 5 Click **Submit**.

Updating user status

User status can be active, inactive, or terminated:

- **Active**—Users can login and perform the InForm application activities for which they have rights.
- **Inactive**—Users can login only for the purpose of changing their passwords.

Note: For an administrator to be able to change a user's password, the user must be in the inactive state.

- **Terminated**—Users cannot login for any purpose.

You can activate or deactivate a user from the Users detail page or from the list of users. To terminate or reinstate a user, use the buttons on the list of users.

From the Users detail page

To update user status from the Users detail page:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 In the list of users, click the name of the user you want to activate or deactivate.
- 3 Do one of the following:
 - To activate an inactive user, select the **User Active** check box.
 - To deactivate a user, clear the User Active check box by reselecting it.
- 4 Click **Submit**.

From the list of users

To update user status from the list of users:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 In the list of users, select the check box next to each user whose status you want to change.

- 3 Do one of the following:
 - To activate the selected users, click **Activate**.
 - To deactivate the selected users, click **Deactivate**.
 - To terminate the selected users, click **Terminate**.
 - To reinstate the selected users, click **Reinstate**.

Managing rights groups

Rights groups are collections of *rights*. A right is the permission to perform a specific activity. The InForm application comes with a predefined set of rights and rights groups covering the activities that are typically restricted to specific roles in a trial. When a new InForm application user is created, an administrator with the right to modify user information assigns the user to a rights group and thus confers on the user permissions to perform specific trial activities.

This section describes how to set up rights groups, assign users to them, and specify display override characteristics. For a summary of specific rights and other authorizations that a user needs to perform typical trial activities, see the InForm application online Help.

Note: To perform administrative activities related to rights groups, you need the **Manage Rights Groups** right. You cannot make changes to the rights group in which you are a member, unless you are the System user.

Creating rights groups

To create a rights group:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 Click the **Rights** tab.

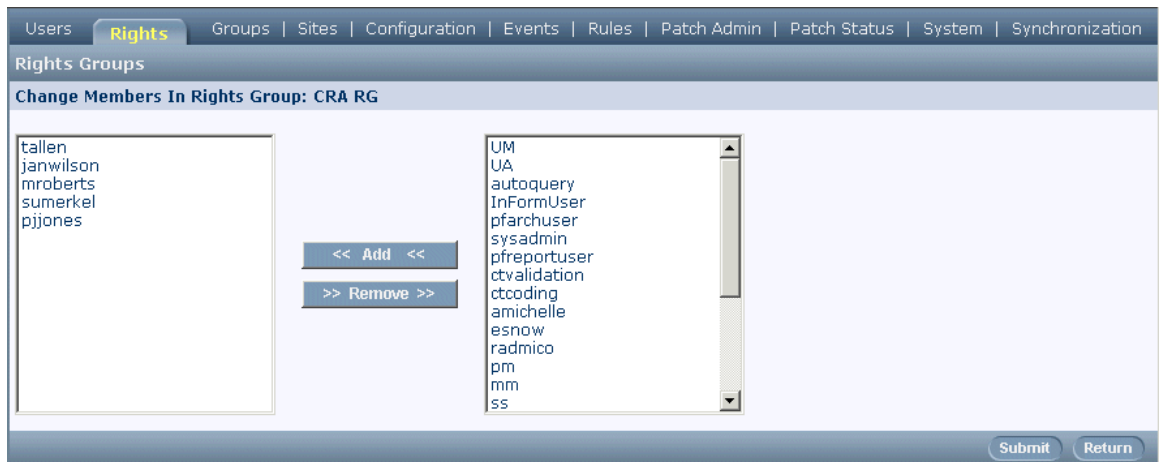
Rights Group	Member Count	Display Overrides	List Users
Admin RG	1	Change Overrides	Change Users
AutoQuery RG	1	Change Overrides	Change Users
CDM RG	2	Change Overrides	Change Users
CRA RG	5	Change Overrides	Change Users
CRC RG	5	Change Overrides	Change Users
InForm Server Group	1	Change Overrides	Change Users
Medical Monitor RG	1	Change Overrides	Change Users
PFArchUser Rights Group	1	Change Overrides	Change Users
PFReportUser Rights Group	1	Change Overrides	Change Users
Principal Investigator RG	3	Change Overrides	Change Users
Project Manager RG	1	Change Overrides	Change Users
Sync Admin	1	Change Overrides	Change Users
Sync User	1	Change Overrides	Change Users
SysAdmin Rights Group	1	Change Overrides	Change Users
System Creator Group	1	Change Overrides	Change Users

- 3 Click the **Add Rights Group** button.
- 4 On the **Rights Group** page, enter the name of the rights group in the **Rights Group Name** box.
- 5 Select each check box that represents a right you want to include in the rights group.
- 6 Click **Submit**.

Assigning users to rights groups

To assign a user to a rights group:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 Click the **Rights** tab.
- 3 In the **List Users** column, click the **Change Users** link for the rights group for which you want to update membership.



- 4 On the **Change Members in Rights Group** page, modify rights group membership as necessary:
 - To add users to the rights group, select the users' names in the users list on the right, and then click the **Add** button. To select more than one user at a time, hold down the **Ctrl** key while selecting each name. To deselect a user while preserving the selection of other users, hold down the **CTRL** key while clicking the name again.
 - To remove users from the rights group, select the users' names in the group members list on the left, and then click the **Remove** button. To select more than one user at a time, hold down the **CTRL** key while selecting each name. To deselect a user while preserving the selection of other users, hold down the **CTRL** key while clicking the name again.
- 5 Click **Submit**.

Note: An alternative method for changing rights group membership is to update the rights group membership of a specific user. For details, see *Assigning rights to a user* (on page 435).

Assigning display overrides to rights groups

About display overrides

Display overrides enable you to use rights groups to refine user access to individual data items on CRFs. You can control data item access by using the following methods:

- **User rights**—Each of these rights applies to all data items on all CRFs:

A user with this right...	Can...
View CRF	View all items in all CRFs.
Enter Data into a CRF and View CRF	Add data to items that have not had data entered previously. This right is commonly assigned to users in a CRC role. Note that both the Enter Data into a CRF and the View CRF rights must be assigned to enable this access.
Edit Data on a CRF and View CRF	Update data in items in which data has been entered previously. This right is commonly assigned to users in a CRC role. Note that both the Edit Data on a CRF and the View CRF rights must be assigned to enable this access.

- **Item definition display overrides**—The definition of an item includes a Display Override property that enables you to specify whether the item is Hidden, Editable, or Read-Only. Each of these definitions overrides for a single item the access conferred by the View CRF, Enter Data into a CRF, and Edit Data on a CRF rights.

Therefore, a user in a rights group with the View CRF and Enter Data into a CRF rights would be unable to see an item defined with the Hidden Display Override property and would be unable to enter data in an item defined with the Read-Only Display Override property.

For details about the Display Override property, see *Defining item properties* (on page 105).

- **Rights group display overrides**—With rights group display overrides, you can create groups of items and specify whether they are Read-Only, Editable, or Hidden. Each of these specifications overrides for all items in the item group the access conferred by membership in a rights group and the access conferred by the definition of an item-level display override. This additional level of security enables you to give users with the same set of rights different access to specific items.

Enabling rights group overrides

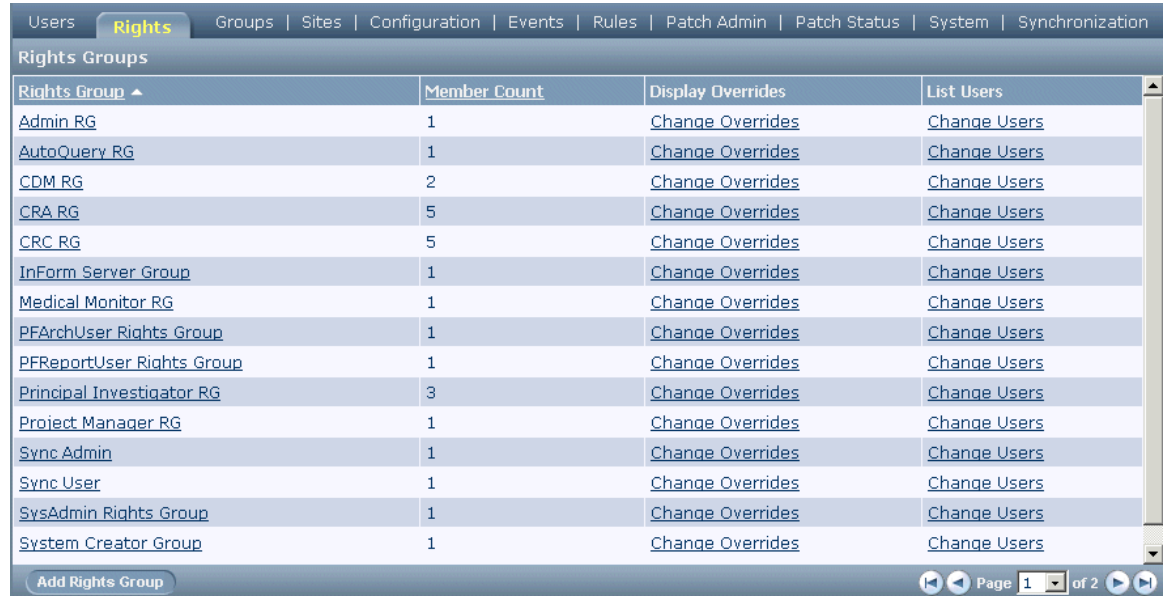
To enable rights group overrides:

- 1 Create an item group that specifies the items for which to apply display overrides. For details, see *Creating item groups*.
- 2 Follow the instructions below to assign or modify the display overrides associated with a rights group.

Assigning or modifying rights group overrides

To assign or modify the display overrides associated with a rights group:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 Click the **Rights** tab.



Rights Group	Member Count	Display Overrides	List Users
Admin RG	1	Change Overrides	Change Users
AutoQuery RG	1	Change Overrides	Change Users
CDM RG	2	Change Overrides	Change Users
CRA RG	5	Change Overrides	Change Users
CRC RG	5	Change Overrides	Change Users
InForm Server Group	1	Change Overrides	Change Users
Medical Monitor RG	1	Change Overrides	Change Users
PFArchUser Rights Group	1	Change Overrides	Change Users
PFReportUser Rights Group	1	Change Overrides	Change Users
Principal Investigator RG	3	Change Overrides	Change Users
Project Manager RG	1	Change Overrides	Change Users
Sync Admin	1	Change Overrides	Change Users
Sync User	1	Change Overrides	Change Users
SysAdmin Rights Group	1	Change Overrides	Change Users
System Creator Group	1	Change Overrides	Change Users

- 3 In the **Display Overrides** column, click the **Change Overrides** link for the group for which you want to update display overrides.

The **Change Read-Only Display Override in Rights Group** page appears. (The page title reflects the default display override type, Read-Only.)



Change Read-Only Display Override in Rights Group: CRC RG

LabQuestion

<< Add <<

>> Remove >>

Read-Only Editable Hidden

Submit Return

- 4 On the Change Read-Only Display Override in Rights Group page, click the button for the display override type you wish to maintain: **Read-Only**, **Editable**, or **Hidden**.

- 5 Add to or modify the list of item groups that make up the display override:
 - To add item groups to the rights group, select the item group names in the list of item groups on the right; then click the **Add** button. To select more than one item group at a time, hold down the CTRL key while selecting each name. To deselect an item group while preserving the selection of other item groups, hold down the **Ctrl** key while clicking the name again.
 - To remove item groups from the rights group, select the item group names in the group members list on the left then click the **Remove** button. To select more than one item group at a time, hold down the CTRL key while selecting each name. To deselect an item group while preserving the selection of other item groups, hold down the CTRL key while clicking the name again.
- 6 Click **Submit**.

Note: An item group can be in only one display override per rights group. For example, if you move an item group into the Read-Only override list and click Submit, that item group is not available for selection in the current rights group for the Editable override.

Managing groups

Groups allow you to associate users who have similar roles in a trial. Groups provide an advanced level of authorization. In order to perform certain activities, a user must both have rights to perform the activities and be in a group for which the activities are authorized. This section describes the types of groups supported in the InForm Architect application and tells how to define and assign users to these types of groups.

Types of groups

The InForm application allows you to define and maintain the following types of groups:

Group Type	Description
Query	<p>Allows any user who is a member of a query group to change the status of a query that was opened by another member of the query group, as long as both users were members of the group at the time the query was opened.</p> <p>A user can be a member of only one query group at a time, but can be reassigned to a different query group.</p> <p>A user who opens a query can change the status of that query at any time, regardless of group membership status.</p>
Signature	<p>Allows any member of the same signature group to sign forms if the user has the Sign Form right, or to sign case books if the user has the Sign Case Book right.</p> <p>A user can only be a member of one Signature Group. When it is defined in the database, a form that requires a signature can be associated with zero, one, or more signature groups. When a form is associated with a signature group, it must be signed by a member of that signature group. When a form is associated with more than one signature group, it must be signed by a member of each signature group.</p> <p>For example, to ensure that a form is assigned by both a sponsor and a site representative, create a signature group for the sponsor and a signature group for the site, and associate the form with both signature groups. Assign sponsor users who have signature rights to the sponsor signature group, and assign site users who have signature rights to the site signature group.</p>
ItemGroup	<p>Identifies a list of items appearing on forms for which display override characteristics can be changed through a rights group.</p>
ManagerUser	<p>Defines a set of users whose activities are included in the CRA versions of legacy ASP reports. A user who is a designated manager of a ManagerUser group can view the CRA versions of reports.</p>
Reporting	<p>Defines the reporting functionality and type of access available to users with reporting rights. Some Reporting groups allow members to access only standard reports; others allow members access to ad hoc reporting.</p>

Creating groups

To create a group:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 Click the **Groups** tab.

The screenshot shows the 'Groups' tab in the application. At the top, there are navigation tabs: Users | Rights | **Groups** | Sites | Configuration | Events | Rules | Patch Admin | Patch Status | System | Synchronization. Below the tabs is a table with the following columns: Group Name, Group Type, Member Count, Description, and Members. The table lists various groups such as Ad Hoc Users, Authors, Clinical Data Users, CRA Query, CRA Sigs, CRC Query, Directory Administrators, Investigator Sigs, LabQuestion, Operational Data Users, Publishers, Report Administrators, ReportList, Server Administrators, and Site Users. Each row has a 'Change' link in the Members column. At the bottom left of the table is an 'Add Group' button, and at the bottom right is a pagination control showing 'Page 1 of 2'.

Group Name	Group Type	Member Count	Description	Members
Ad Hoc Users	Reporting	0	Users who have access to Ad Hoc Reporting	Change
Authors	Reporting	0	Users who have access to Ad Hoc Reporting and Report Studio	Change
Clinical Data Users	Reporting	0	Users who have access to clinical data	Change
CRA Query	Query	9	CRA Query	Change
CRA Sigs	Signature	3	CRA Sigs	Change
CRC Query	Query	3	CRC Query	Change
Directory Administrators	Reporting	0	Users who can administer authorization rights. They administer groups, accounts, contacts, distribution lists, data sources, and printers	Change
Investigator Sigs	Signature	2	Investigator Sigs	Change
LabQuestion	ItemGroup	1	Clinical significance question	Change
Operational Data Users	Reporting	0	Users who have access to operational data (including queries and comments) except user data	Change
Publishers	Reporting	0	Users who can create folders and other content within the Public Folder space	Change
Report Administrators	Reporting	0	Users who can administer authorized public content such as setting permissions on folders and reports	Change
ReportList	ManagerUser	0	CRA List for Report Viewing	Change
Server Administrators	Reporting	0	Users who can administer servers, dispatchers, and jobs	Change
Site Users	Reporting	9	Site Users	Change

- 3 Click **Add Group**.

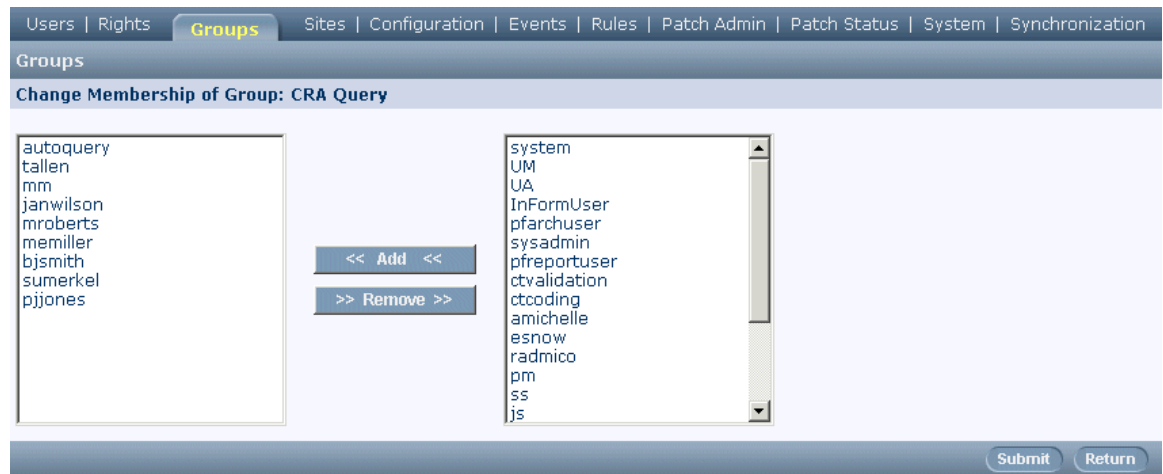
The screenshot shows the 'Edit Group Properties' form. At the top, there are navigation tabs: Users | Rights | **Groups** | Sites | Configuration | Events | Rules | Patch Admin | Patch Status | System | Synchronization. Below the tabs is a form with the following fields: 1. Group Type (a dropdown menu with 'Query' selected), 2. Group Name (a text input box), and 3. Group Description (a larger text input box). At the bottom right of the form are two buttons: 'Submit' and 'Return'.

- 4 On the **Groups** detail page, select the group type from the **Group Type** list.
- 5 In the **Group Name** box, enter the name of the group.
- 6 In the **Group Description** box, enter a description of the group.
- 7 Click **Submit**.

Assigning users to groups

To assign users to a group:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 Click the **Groups** tab.
- 3 In the **Members** column, click the **Change** link for the group for which you want to update membership.



- 4 On the **Change Membership of Group** page, modify group membership as necessary:
 - To add users to the group, select the users' names in the users list on the right then click the **Add** button. To select more than one user at a time, hold down the **Ctrl** key while selecting each name. To deselect a user while preserving the selection of other users, hold down the **Ctrl** key while clicking the name again.
 - To remove users from the group, select the users' names in the group members list on the left then click the **Remove** button. To select more than one user at a time, hold down the **Ctrl** key while selecting each name. To deselect a user while preserving the selection of other users, hold down the **Ctrl** key while clicking the name again.
- 5 Click **Submit**.

Note: An alternative method for changing group membership is to update the group membership of a specific user. For details, see *Managing group assignments* (on page 432).

Creating item groups

Item groups are sets of items that you group together so that you can assign a display override to them. Display overrides enable you to specify that, for a particular rights group, the group of items that makes up an item group is Hidden, Editable, or Read-Only. This designation overrides the rights conveyed by membership in the rights group and also overrides the display properties of the items in the group. For additional information about display overrides, see *Assigning display overrides to rights groups*.

To create an item group:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 Click the **Groups** tab.
- 3 Click **Add Group**.

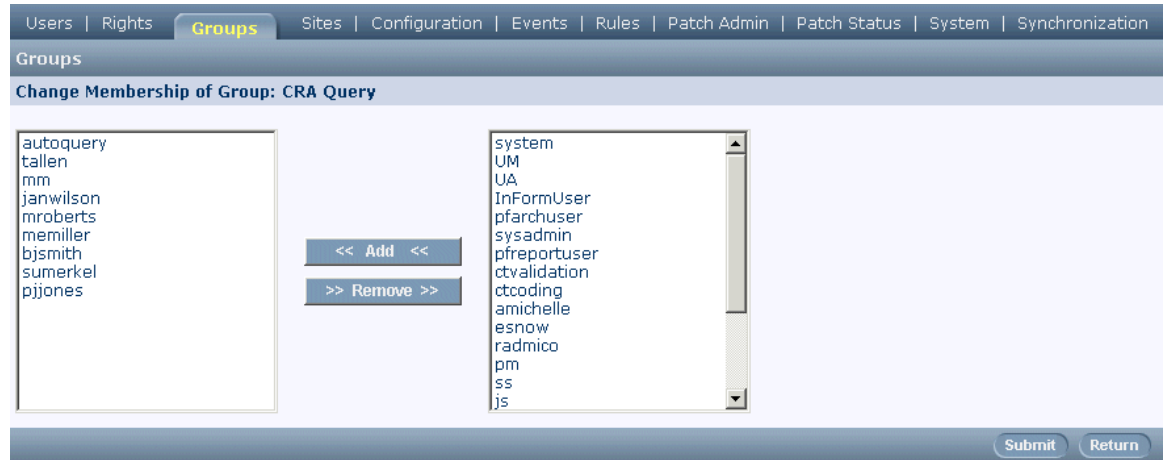
Edit Group Properties		
1.	Group Type	Query
2.	Group Name	<input type="text"/>
3.	Group Description	<input type="text"/>

- 4 On the **Groups** detail page, select **ItemGroup** from the **Group Type** list.
- 5 In the **Group Name** box, enter the name of the group.
- 6 In the **Group Description** box, enter a description of the group.
- 7 Click **Submit**.

Assigning items to item groups

To assign items to item groups:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 Click the **Groups** tab.
- 3 In the **Members** column, click the **Change** link for the item group for which you want to add or change items.



- 4 On the **Change Membership of Item Group** page, modify item group membership as necessary:
 - To add items to the item group, select names in the items list on the right then click the **Add** button. To select more than one item at a time, hold down the **Ctrl** key while selecting each name. To deselect an item while preserving the selection of other items, hold down the **Ctrl** key while clicking the name again.
 - To remove items from the item group, select names in the item members list on the left then click the **Remove** button. To select more than one item at a time, hold down the **Ctrl** key while selecting each name. To deselect an item while preserving the selection of other items, hold down the **Ctrl** key while clicking the name again.
- 5 Click **Submit**.

Note: An item can belong to only one item group.

Managing sites

The InForm Architect application enables you to create sites and associate them with users.

Creating sites

To create a site:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 Click the **Sites** tab.

Site Name	Site Abbrev	EmailAddress	User Count	Users	Patients
Mass General Hospital	MGH	-	11	Change	List
McLean Hospital	MCLEAN	-	7	Change	List
PhaseForward	PF	-	15	Change	List
Veterans Administration Hospital	VA	-	7	Change	List

- 3 Click the **Add Site** button.

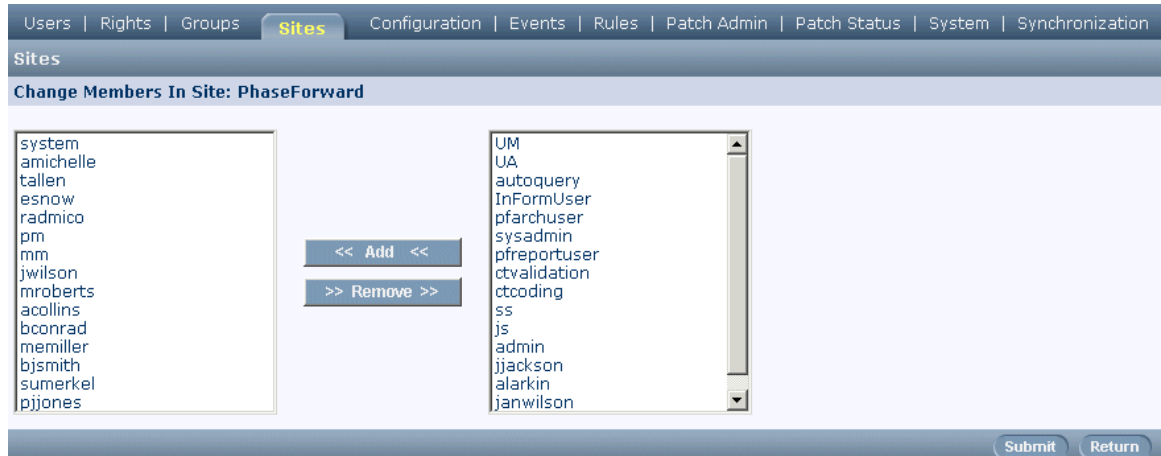
Site Information	
1. Site Name	<input type="text"/>
2. Site Mnemonic	<input type="text"/>
3. Address1	<input type="text"/>
4. Address2	<input type="text"/>
5. City	<input type="text"/>
6. State/Province	<input type="text"/>
7. Country	<input type="text"/>
8. Postal Code	<input type="text"/>
9. Day Phone Number	<input type="text"/>
10. Alternate Phone Number	<input type="text"/>
11. Fax Number	<input type="text"/>
12. Email	<input type="text"/>
13. TimeZone	(GMT-12:00) Eniwetok, Kwajalein
14. Date Format	<input type="text"/>
15. Study Version: <i>(Change it at the current MedML Installer Server.)</i>	Study Version 1.0

- 4 On the Sites detail page, enter the following information about the site:
 - **Name and mnemonic** (an abbreviation used to associate protocol and form updates with the site).
 - **Address information**—Street address, city, state or province, country, and postal code.
 - **Contact data**—Day phone number, alternate phone number, fax number, and email address.
 - **Time zone.**
 - **Date format**—Month/Day/Year, Day/Month/Year, or Year/Month/Day.
 - **Study version.**
 - **Site server**—Read-only.
- 5 Click **Submit**.

Assigning users to sites

To assign a user to a site:

- 1 In the **Trial Objects** window, click the **Admin** icon.
- 2 Click the **Sites** tab.
- 3 In the **Users** column, click the **Change** link for the site for which you want to update user authorization.



- 4 On the **Change Members in Site** page, modify site authorization as necessary:
 - To add users to the site, select the users' names in the users list on the right, and then click the **Add** button. To select more than one user at a time, hold down the **Ctrl** key while selecting each name. To deselect a user while preserving the selection of other users, hold down the **Ctrl** key while clicking the name again.
 - To remove users from the site, select the users' names in the site users list on the left, and then click the **Remove** button. To select more than one user at a time, hold down the **Ctrl** key while selecting each name. To deselect a user while preserving the selection of other users, hold down the **Ctrl** key while clicking the name again.

5 Click **Submit**.

Note: You can also assign an individual user to one or more sites by using the User Group Selection screen. For details, see *Managing site assignments* (on page 435).

Defining sponsors

Sponsor definitions specify address and contact information about a sponsor. Sponsor information is for documentation and reporting purposes only and does not appear in the InForm user interface.

To create a sponsor definition:

- 1 Create an XML file that includes the appropriate SPONSOR MedML tags.
- 2 Install the sponsor definition in the InForm application trial database by processing it with the MedML Installer tool.

Creating a sponsor

Create a sponsor definition for the organization sponsoring the trial. The syntax of the SPONSOR tag is shown below.

```
<SPONSOR
  NAME="name"
  [PROGRAM="program-name of trial"]
  [THERAPEUTICAREA="therapeutic area of trial"]
  [NOTE="notes"]
  [ADDRESS="addr1"]
  [ADDRESS2="addr2"]
  [CITY="name"]
  [STATE="name"]
  [PROVINCE="name"]
  [ZIPCODE="code"]
  [POSTCODE="code"]
  [COUNTRY="name"]
  [PHONE="num"]
  [ALTPHONE="num"]
  [FAX="num"]
  [EMAIL="addr"]
  [LANGUAGE="name"]
  [LOGOFILE="file"]
  [LOGOTYPE="GIF|JPEG|TEXT"]>
</SPONSOR>]
```

To define a sponsor:

- 1 Start the block by opening the <SPONSOR tag.
- 2 Enter the NAME. This is the name of the organization and must be unique among sponsors.

Note: NAME is the only required attribute.

- 3 Define the sponsor's contact information by specifying the address, telephone numbers and other attributes. Add as many of these as are appropriate. Note that if you specify a logo file, you must also specify its type.
- 4 Close the SPONSOR tag.

Managing numbering

The InForm application automatically generates screening and enrollment numbers. Additionally, the InForm application can generate drug kit numbers when patients are randomized. Each of these numbering types is configurable.

To set up numbering in the InForm application, the following tags are required:

- **SEQUENCETYPE**—Defines types of sequence numbers. The default sequence types are part of the Base installation of the InForm application.
- **SEQUENCE**—Defines specific sequences within sequence type. The default sequences are also part of the Base installation.
- **SYSCONFIG**—Defines a system configuration variable that specifies the scheme for formatting a sequence number.

Creating sequence types

The default sequence types are part of the Base installation of the InForm application. No additional setup is required. When you install the InForm application, the sequence types for screening, enrollment, and randomization are loaded with the following tags:

```
<SEQUENCETYPE SEQUENCETYPENAME="Screening"/>
<SEQUENCETYPE SEQUENCETYPENAME="Enrollment"/>
<SEQUENCETYPE SEQUENCETYPENAME="Randomization"/>
```

Creating sequences

The default sequences are part of the Base installation of the InForm application. No additional setup is required for screening or enrollment numbers. When you install the InForm application, the sequence types for screening, enrollment, and a randomization sequence for a Simple Central randomization scheme are loaded with the following tags:

```
<SEQUENCE SEQUENCENAME="Screening Number Sequence"
  SEQUENCETYPENAME="Screening"
  UUID="F7F1B3B8-0B5C-11D2-A418-00A0C963E0AC" />
<SEQUENCE SEQUENCENAME="Enrollment Number Sequence"
  SEQUENCETYPENAME="Enrollment"
  UUID="EB75B898-078B-11D2-A417-00A0C963E0AC" />
SEQUENCENAME="Simple SimpleCentral"
SEQUENCETYPENAME="Randomization"
UUID="4F4A0246-5009-11D2-931C-00A0C9769A13" />
```

Note: Alphabetic characters in required UUIDs must be uppercase.

If you configure your trial to support a different randomization scheme, you must create a sequence definition for that scheme:

- 1 Start by typing the **<SEQUENCE** tag.
- 2 Optionally, use the **UUID** attribute to assign a unique identification key to the sequence.
- 3 Use the **SEQUENCENAME** to identify this sequence. The name must be unique among all sequences.

- 4 Use the **SEQUENCETYPENAME** to specify the “Randomization” sequence type. This type must have been defined previously.
- 5 Close the **SEQUENCE** tag.

Specifying sequence number format

In the InForm application, you can configure the format of screening numbers, patient enrollment numbers, and randomization numbers. This section describes the formatting string you must create to specify the format of each of these number types. Specification of numbering formats for screening and enrollment numbers is optional. If you configure the InForm application to generate randomization drug kit numbers, specification of a randomization numbering format is required.

Note: Patient numbers can be automatically generated by the InForm application, using the format that you specify as described in this section, or you can configure the InForm application to enable site personnel to assign patient numbers manually. For information on enabling users to assign manual patient numbers, see *Patient numbers* (on page 360).

The format of each type of sequence number can contain fixed text, site name, stratification code, and a sequential number that can also include a text string. These formatting specifications are in the following order:

[*SPR*]:[*string*][%*flag*][*width*]*type*][*string*]

- *SPR*—Code that indicates the type of screening, patient enrollment, or randomization number. Codes are not case-sensitive. For details, see *Sequence Number Codes* (on page 454).
- *string*—Text string that appears at the beginning of each generated number.
- %—Required character preceding each flag/width/type specification.
- *flag/width/type*—A series of specifications that indicates how the InForm application substitutes values in each generated number. You can include multiple flag/width/type specifications, where each specification includes the following values:
 - *flag*—Character that specifies a prefix used to fill in the numerical portion of the sequence number to maintain the number of characters specified in the width variable. For details, see *Flag characters* (on page 454).
 - *width*—Minimum number of characters in the numerical portion of the sequence number, to be left-filled by the character specified in the flag variable. This specification is optional.
 - *type*—Code that indicates the inclusion of the site name, stratification code, and sequential number in the sequence number. Codes are not case-sensitive. For details, see *Type codes* (on page 455).
- *string*—Text string added to the end of each generated number.

Sequence number codes

Code	Type of sequence number
SC	Simple Central randomization —The trial uses one list of drug kits. Each new patient is assigned the next sequential drug kit number on the list.
CS	Central Stratified randomization —The trial has multiple lists of drug kits. Each new patient is assigned to a drug kit list based on entered patient data. Then, the patient is assigned the next sequential drug kit number on that list.
SS	Simple Site randomization —Each site has a different drug kit list. Each new patient is assigned the next sequential drug kit number on the list for the patient's site.
SR	Stratified by Site randomization —Each site has multiple lists of drug kits. Each new patient is first assigned to the set of lists for the patient's site. Then, the patient is assigned to one of the site's drug kit lists based on entered patient data. Finally, the patient is assigned the next sequential drug kit number on that list.
S1	Cross-site screening number —Screening numbers are assigned sequentially across the trial without regard to which site screens the patient.
SN	Site-based screening number —Each site has a separate sequence of screening numbers.
P1	Cross-site patient number —Enrollment numbers are assigned sequentially across the trial without regard to which site enrolls the patient.
PN	Site-based patient number —Each site has a separate sequence of enrollment numbers.

Flag characters

Character	Description
0	InForm application adds zeroes to the left of the numerical sequence until the specified width is reached
#	InForm application adds pound signs to the left of the numerical sequence until the specified width is reached
-	InForm application adds hyphens to the left of the numerical sequence until the specified width is reached

Type codes

Code	Description
S	Includes the site mnemonic. This type is optional and can be used in screening, patient, and randomization sequence numbers.
T	Includes the stratification code. This type applies to randomization sequence numbers and is required if the randomization type is Central Stratified or Stratified by Site.
Q	Includes a numeric value assigned sequentially to each new patient who is screened, enrolled, or randomized. This type is required for all sequence number format definitions.

Examples

SC:RND-%q

Randomization sequence number for a Simple Central randomization scheme. Each number consists of the string RND and a sequential number, for example: RND-12.

CS:RND-%t-%q-PF105

Randomization sequence number for a Central Stratified randomization scheme. Each number consists of the string RND, the stratification code, a sequential number, and the trial name, PFST, for example: RND-CS_WT150-12-PFST

SN:SCR-%s-%05q

Screening sequence number. Each number consists of the string SCR, the site mnemonic, and a five-digit sequential number that is left-filled with zeroes, for example: SCR-PF-00012.

Note: Randomization number configuration is required, if you are configuring your trial to support generating drug kit numbers. If you do not configure patient and screening number formats, the InForm application uses the following default formats:

Screening number—SN:SCR-%s-%q (the string SCR, followed by site mnemonic, followed by generated number)

Patient enrollment number—PN:ENR-%s-%q (the string ENR, followed by site mnemonic, followed by generated number)

Loading sequence number formats

To load the sequence number format into the database, include a SYSCONFIG element for the sequence number format in the .xml file used to set configuration variable values. The following shows the default format for drug kit numbers under Simple Central randomization that appears in the core_SystemConfig.xml file in the XMLBase directory of the InForm application installation.

```
<SYSCONFIG CONFIGNAME="RandSimpleCentral"
  TYPE="0"
  VALUE="SC:RND-%q" />
```

Defining system configuration settings

The settings of configuration parameters specify customizable, system-wide InForm behaviors. Default system configuration settings are installed with the Base trial in the `core_SystemConfig.xml` file.

This section describes how to set configuration parameters. For a description of each specific parameter you can set, see the SysConfig tag definition in the MedML online Help.

Setting configuration parameters with SYSCONFIG

Use the SYSCONFIG tag to specify the values of system configuration parameters. The format of the SYSCONFIG tag is as follows:

```
<SYSCONFIG  
  [UUID="id"]  
  CONFIGNAME="name"  
  TYPE="0"  
  VALUE="n" />
```

To set the value of a system configuration parameter:

- 1 Start the block with the <SYSCONFIG tag.
- 2 Optionally, specify a unique identifier for the parameter with the UUID attribute.
- 3 Specify the name of the configuration parameter by using the CONFIGNAME attribute. The SysConfig topic of MedML online help describes each parameter you can set.
- 4 Assign a value of “0” to the TYPE attribute.
- 5 Use the VALUE attribute to set the value of the parameter.
- 6 Close the block with the /> symbol.

CHAPTER 12

Updating study definition data

In this chapter

Overview	458
Creating new form versions	459
Updating a study version	460
Activating a study version.....	462
Revising without updating the study version	463

Overview

Trial forms and the corresponding user documentation are typically revised from time to time. After making the revisions to the forms and user documentation, you incorporate the changes into the trial by updating the STUDYVERSION and STUDYVERSIONDOC tags that describe the configuration of the trial. Then, as each site is ready to use the new version, you cut over the site by updating its STUDYVERSIONSITE tag. You make all of these revisions by editing the appropriate XML files and installing the new files by using the MedML Installer tool.

Note: When you revise a control, you cannot change its data type.

This system of controlling versions preserves a record of each version of the trial and the forms that comprised it. It also allows you to cut over sites selectively. That way if the changes require site approval, the new version can be activated as each site grants its approval.

Methods for updating the trial

This chapter describes how to update trial definition data by creating a new study version. This method has been developed to comply with Good Clinical Practice (GCP) and is strongly recommended.

The InForm application also supports a second method, in which you revise form definitions directly without creating a new study version. This method is described in *Revising without updating the study version* (on page 463).

MedML tags for trial updates

A STUDYVERSION tag represents all of a trial's forms and formsets at a particular point in time. Each time forms change, or are added or dropped, a new study version must be prepared to incorporate the changes into the trial definition. Study versions are identified by a VERSIONDESCRIPTION attribute. The attribute is simply a key that you assign to uniquely identify the version.

A study version also has a corresponding document set, represented by the STUDYVERSIONDOC tag. This tag lists all the documentation files that apply to a version identified by a particular VERSIONDESCRIPTION value.

Study versions (and study version docs) are mapped to trial sites using the STUDYVERSIONSITE tag. The STUDYVERSIONSITE tag specifies the current version active at a site. This allows you to control the version in use on a site-by-site basis. The cutover to a new version occurs when the STUDYVERSIONSITE for the site is processed by the MedML Installer.

Creating new form versions

When a form containing patient data changes after a trial has begun because of an approved protocol change or other sponsor request, sites use the new version of the form for as many trial patients as makes sense. For example, if a form change involves an additional item of data to collect, and the old version of the form has already been completed for a patient, a site does not attempt to collect the new data if it depends on a time that has already passed—for example, a blood draw that must occur one hour after the patient receives a dose of the trial drug. However, if the data item is not time-sensitive, a site generally tries to collect the item, and it requires the new version of the form to do so. This means that, when you create a new version of a form, you must consider whether multiple versions of the form will be needed under the same STUDYVERSION:

- If a form change does not require sites to collect data from patients for whom the form has already been completed, revise the Form element, including additional or changed controls, Items, or Sections, as appropriate.
- If a form change **does** require sites to collect data from patients for whom the form has already been completed, revise the Form element, including additional or changed controls, Items, or Sections, as appropriate. In the FORM definition, include the ALTREFNAME attribute to indicate that you are creating an alternate definition of an existing form. For information on creating an alternate version of a form, see *Alternate versions of forms* (on page 341).

Changes made to Visit Reports and Regulatory Document forms are revised differently from the forms listed above. Since Visit Reports and Regulatory Document forms do not contain patient data, an audit trail of changes does not need to be maintained. Therefore, when a change is made to one of these forms, it is automatically reflected in all instances of the form and only one version of the form is preserved.

Updating a study version

Changes in a trial protocol typically require IRB approval, and their implementation needs to be carefully timed. When a protocol change is approved, the trial can begin to use the new forms, if any, associated with the change when you:

- 1 Create a new STUDYVERSION, including the new or changed forms, by copying or revising the definition of the old STUDYVERSION, changing the VERSIONDESCRIPTION attribute, and updating the STUDYVERSIONDOC description in the same way.
- 2 Deploy the new STUDYVERSION definition at the sites by updating the VERSIONDESCRIPTION attribute of each STUDYVERSIONSITE element where the change applies. For a form change that does not involve a protocol amendment, update the STUDYVERSIONSITE element for each site. For a change that does involve a protocol amendment, update the STUDYVERSIONSITE element for a site only as it receives IRB approval or other authorization (for example, communication from the sponsor when a safety-driven change requires expedited implementation with approval pending).

Updating the STUDYVERSION tag

The STUDYVERSION tag identifies the version of the formsets (and the versions of the forms they contain) that comprise a version of the trial. The version is identified by a unique VERSIONDESCRIPTION key.

Note that the InForm application automatically keeps track of form and formset versions as you modify them. However, these modifications do not become part of a study version until you load a new definition of the STUDYVERSION tag. When the InForm application loads trial definition data for the new study version, it takes a snapshot of the latest versions of the forms and incorporates them into the new study version.

Thus, to update a STUDYVERSION tag:

- 1 Review the forms and formsets to make sure they contain the latest information.
- 2 Display the STUDYVERSION you want to update in an editor. The syntax of the tag is shown below.

```
<STUDYVERSION
STUDYNAME=" name "
VERSIONDESCRIPTION=" text "
[ PROTOCOL=" name " ]
[ SPONSORDATE=" date " ]
[ TRADEDRUGNAME=" name " ]
[ GENERICDRUGNAME=" name " ]
[ SPONSORDRUGNAME=" name " ]>
<FORMSET+ attributes/>
</STUDYVERSION>
```

- 3 Assign a new VERSIONDESCRIPTION to the trial. It must be a string which uniquely identifies this version. You may want to incorporate the date of the version into the string.
- 4 Review the list of FORMSET attributes. Make any additions or deletions required, as described in *Creating new form versions* (on page 459).
- 5 Close the STUDYVERSION tag.

Updating the STUDYVERSIONDOC tag

Each study version must have a corresponding STUDYVERSIONDOC version. This is true even if there are no changes to the documentation. The STUDYVERSIONDOC tag specifies the documentation files that describe the version.

- 1 Review the documentation updates and note the documentation files that are to be replaced, added, and deleted.

Note: To preserve an older version of a documentation file, make sure you do not overwrite the original file when making a revision. Instead, assign the revised file a new name.

- 2 Display the STUDYVERSIONDOC you want to update in an editor. The syntax of the tag is shown below:

```
<STUDYVERSIONDOC  
VERSIONDESCRIPTION= "text "  
DOCREFNAME= "name "  
[ORDER= "n" ] />
```

- 3 Update the VERSIONDESCRIPTION key to the same value you used in the corresponding STUDYVERSION tag.
- 4 Update the DOCREFNAME attributes to list the files that comprise this version of the documentation.
- 5 You can specify the order of the tab displayed for the document by following the DOCREFNAME attribute with an ORDER attribute. By default, the tab order is the same as the order of the documents in the list.
- 6 Use the MedML Installer tool to process the STUDYVERSION and STUDYVERSIONDOC tags to create the new version.

A new version of the trial has now been created.

Activating a study version

As a site is ready to use a new version, activate the version using a `STUDYVERSIONSITE` tag. Note, that in some cases, a site must approve the changes to the trial before the version may be activated.

Updating the `STUDYVERSIONSITE` tag

To update a `STUDYVERSIONSITE` tag:

- 1 In a text editor, display the study version XML file that you want to update. The syntax of the tag is shown below:

```
<STUDYVERSIONSITE  
  VERSIONDESCRIPTION="text "  
  [ SITENAME="name" ]  
  [ ITEMNEMONIC="name" ]  
  [ ACCEPTDATE="date" ]/>
```

- 2 The `VERSIONDESCRIPTION` key identifies the version. Update the key to the same value you used in the corresponding `STUDYVERSION` tag.
- 3 Either `SITENAME` or a `ITEMNEMONIC` attribute is required to identify the site.
- 4 Optionally, set the value of the `ACCEPTDATE` attribute to the date of IRB approval. Note that this date is for documentation purposes only. The actual date that a site moves to a new StudyVersion is the date that the StudyVersionSite component for the site is revised in the database.
- 5 Use the MedML Installer to process the `STUDYVERSIONSITE` tag when the site is ready to use the new version.

Revising without updating the study version

For most trial definition changes, creating a new study version is strongly advised. However, from time to time you may find that you need to make *minor* cosmetic changes to a form and would prefer not to create an entirely new version of the trial. For example, you may want to correct a typographical error on a form, or change the position of a caption.

For these types of changes, the following method is available. In this method, you apply changes directly to the current version of a form. Because the form version does not change, the study version does not need to be updated. You can make changes to the following trial component definitions in this way:

- **Section**—by using the UPDATE_FORM_SECTION MedML tag
- **Item**—by using the UPDATE_SECTION_ITEM MedML tag

Any changes completed using this method must be re-applied manually on all destination machines after migration.

The impact of such changes should be tested thoroughly against existing patient data as well as new patient data.

Note: Making trial definition changes without creating a new study version violates Good Clinical Practice (GCP), and can cause problems with trial data. Be sure you understand the consequences of making changes this way before you implement those changes in a live trial. *This method should never be used to change the datatype of a control or remove an existing control from an item especially if that item contains previously entered data.*

Note: Making trial definition changes without creating a new study version violates Good Clinical Practice (GCP), and can cause problems with trial data. Be sure you understand the consequences of making changes this way before you implement those changes.

Revising a section on a form

To revise a section, use the UPDATE_FORM_SECTION MedML tag. This tag has the following syntax:

```
<UPDATE_FORM_SECTION
  FORM_REFNAME="name"
  FORM_REVISION="number"
  SECTION_REFNAME="name"
  SECTION_REVISION="number" />
```

- 1 Stop the InForm application trial and make sure that no one can access the trial or modify the trial data.
- 2 Make the necessary changes to the XML for the section you want to modify.
- 3 Install the new version of the section using the MedML Installer.

- 4 Create an XML file containing an UPDATE_FORM_SECTION tag for the section you want to change. You can use SQL to retrieve the form and section revision numbers or you can look in the database directly.
 - For FORM_REFNAME, specify the RefName of the form in which you are updating the section.
 - To find the FORM_REVISION number, open the PF_PAGE table and look up the most recent (highest) PAGEREVISIONNUMBER for the corresponding PAGEREFNAME.
 - For SECTION_REFNAME, specify the RefName of the section you have revised.
 - To find the SECTION_REVISION number, open the PF_SECTION table and look up the most recent (highest) SECTIONREVISIONNUMBER for the corresponding SECTIONREFNAME.

Note: These instructions assume that you want to revise the current version of the form. If your trial has multiple study versions, and you want to apply the revision to all versions of the form, create a separate UPDATE_FORM_SECTION tag for each revision.

- 5 Run the MedML Installer and apply the XML file containing the UPDATE_FORM_SECTION tag. The installer validates that the form RefName and revision is correct, and that the form contains the section you updated.
- 6 Restart the trial.
- 7 Test the update form version after applying this change. The MedML Installer does not validate the metadata against any existing patient data.

Revising an item on a form

To revise an item, use the UPDATE_SECTION_ITEM MedML tag. This tag has the following syntax:

```
<UPDATE_SECTION_ITEM
  SECTION_REFNAME="name"
  SECTION_REVISION="number"
  ITEM_REFNAME="name"
  ITEM_REVISION="number" />
```

- 1 Stop the InForm application trial and make sure that no one can access the trial or modify the trial data.
- 2 Make the necessary changes to the XML for the item you want to modify.
- 3 Install the new version of the item using the MedML Installer.
- 4 Create an XML file containing an UPDATE_SECTION_ITEM tag for the section you want to change. You can use SQL to retrieve the section and item revision numbers or you can look in the database directly.
 - For SECTION_REFNAME, specify the RefName of the section in which you are updating the item.
 - To find the SECTION_REVISION number, open the PF_SECTION table and look up the most recent (highest) SECTIONREVISIONNUMBER for the corresponding SECTIONREFNAME.
 - For ITEM_REFNAME, specify the RefName of the item you have revised.

- To find the ITEM_REVISION number, open the PF_ITEM table and look up the most recent (highest) ITEMREVISIONNUMBER for the corresponding ITEMREFNAME.

Note: If the item you are modifying is contained in an itemset, the RefName and revision number you enter here must be for the itemset. Itemset RefNames and revision numbers are also stored in the PF_ITEM table.

These instructions assume that you want to revise the current version of the form. If your trial has multiple study versions, and you want to apply the revision to all versions of the form, create a separate UPDATE_SECTION_ITEM tag for each revision.

- 5 Run the MedML Installer and apply the XML file containing the UPDATE_SECTION_ITEM tag. The installer validates that the form RefName and revision is correct, and that the form contains the section you updated.
- 6 Restart the trial.
- 7 Test the update form version after applying this change. The MedML Installer does not validate the item metadata against any existing patient data.

Metadata Update report

When you update a section or item definition by using the UPDATE_FORM_SECTION or UPDATE_SECTION_ITEM tags, you can generate a report that summarizes the updates you have made and indicates whether the updates involve items in which patient data has previously been entered. This report, called the Metadata Update report, is available in the Reports module of the InForm application. To view the report, you must be in a rights group that includes the Admin Reports right.

For detailed information about the Metadata Update report, see the InForm application online Help. The following figure illustrates the Metadata Update report.

The screenshot displays the InForm Clinical Research Administrator interface in a Microsoft Internet Explorer browser window. The browser's address bar shows the URL: `http://beachmont/pf206/pfts.dll?S=750a3a97&C=Navigator_RenderTopFrameset`. The application's navigation menu includes tabs for Patient, Case Records, Query, Source Verification, Admin (highlighted), and Custom. A sidebar on the left contains a user profile for CRA and buttons for HOME, HELP, LOGOUT, Enroll, Patients, Queries, Monitor, Signatures, Documents, Admin, and Reports. The main content area is titled "Metadata Update Report" and shows a list of updated items for the form "Ophthalmologic Examination (EYE)".

Form: Ophthalmologic Examination (EYE)

Section: Visual Acuity Distance and Near (VISION)

- Item:** Visual Acuity for Distance (left eye): (LDIST)
- Item:** Visual Acuity for Distance (right eye): (RDIST)
- Item:** Visual Acuity for Near (left eye): (LNEAR)
- Item:** Visual Acuity for Near (right eye): (RNEAR)

Section: Retinal Examination (RETINAL)

- Item:** Are cotton-wool spots present? If so, indicate percentage of the retina involved (right eye): (RWOOL)
- Item:** Are cotton-wool spots present? If so, indicate percentage of the retina involved (left eye): (LWOOL)
- Item:** Is nicking of the retinal vessels evident? If so, indicate grade (right eye): (RNICK)
- Item:** Is nicking of the retinal vessels evident? If so, indicate grade (left eye): (LNICK)

A "Return" button is located at the bottom right of the report content area.

CHAPTER 13

Defining electronic trial documentation

In this chapter

Overview	468
Choosing an authoring tool.....	469
Creating a trial protocol	470
Creating a documentation definition XML file.....	472
Examples of documentation definition XML files.....	473
Creating a Table of Contents file	476
Linking between document files.....	477
Trial documentation and study versions	479
Creating CRF Help	481
Revising trial documentation	485

Overview

This chapter describes how to create trial-specific documents that users can display in the Document window during an InForm session. The Document window opens when a user clicks the Documents button. The chapter describes how to:

- Create an online trial protocol.
- Create online CRF help.
- Set up links between document files so users can navigate between them by clicking highlighted link text.
- Create a documentation definition file to load trial documents into the database.
- Associate a documentation definition with a StudyVersion.

Note: To define trial documentation, you must manually code the required MedML and formatting tags described in this chapter. Trial documentation definition is not supported by the InForm Architect application.

This chapter does not cover the development of online Help for the InForm application or the Cognos 10 Business Intelligence tools. Online Help for the InForm application is available from the Help button in the navigation pane of the InForm application when you install the software. Online Help for the Cognos 10 Business Intelligence tools is available in the following locations when a trial is configured for reporting:

- **Using Reporting Tools link**—Available from the Help button in the navigation pane of the InForm application. This link provides access to the Cognos documentation set. Use this help link to understand the mechanics of using the Reporting and Analysis module, such as how to create different kinds of reports, how to create calculations and charts, and how to sort on different columns.
- **Cognos 10 Business Intelligence Ad Hoc tools help link**—Available when you access Ad Hoc Reporting. This link provides access to the Cognos documentation for Query Studio (Query Studio is the Cognos application used by the InForm software to provide Ad Hoc Reporting capability).

Choosing an authoring tool

Trial documentation in the InForm application consists of sets of HTML files that you store in the InForm application database by processing them with the MedML Installer. To create files in HTML, you can:

- Use an HTML editor to author directly in HTML. Any text editor, such as Microsoft Windows Notepad, provides this capability. However, you may find it convenient to use a tool with specific HTML capabilities. Microsoft FrontPage enables HTML authoring with a graphical interface similar to a word processor. Many other HTML editors provide text editing features with specialized tools that make it easy to add HTML tags.
- Use a word processor with an HTML conversion tool. Microsoft Word provides this capability. When you finish preparing your document in regular Word format, you use a simple menu command to convert the document to HTML format. Following the conversion, you can edit the converted file with an HTML or text editing tool to create Table of Contents or other links and to make any necessary formatting changes.

Creating a trial protocol

Step	Description	More information
1	Create one or more HTML files containing the text of the trial protocol. One file per protocol section is suggested.	
2	Within each file, make code additions and changes required for processing by the MedML Installer utility.	<i>Modifying the trial protocol HTML files</i> (on page 471)
3	Create an XML file that defines the trial protocol in a DOCUMENTATION tag and as many DOCTYPE tags as there are trial protocol HTML files. Use the BOOKMARKDOC DOCTYPE attribute.	<ul style="list-style-type: none"> • <i>Creating a documentation definition XML file</i> (on page 472) • <i>Examples of documentation definition XML files</i> (on page 473) • MedML online Help
4	Create a Table of Contents file that includes each heading or subsection to which you want users to be able to link.	<i>Creating a Table of Contents file</i> (on page 476)
5	Link each item in the Table of Contents file to the appropriate trial protocol HTML file.	<i>Linking between document files</i> (on page 477)
6	Associate the DOCUMENTATION definition with a study version.	<i>Trial documentation and study versions</i> (on page 479)
7	Process the trial documentation in the MedML Installer utility to load the files into the InForm database.	<i>Utilities Guide</i>

Modifying the trial protocol HTML files

After you create the HTML files for the trial protocol, make the following modifications:

- 1 Replace the `</head>` and `<body>` tags at the beginning of each file with the following Javascript function definition and call:

```
<script language="Javascript">
function OnLoad()
{
%s
}
</script>
</head>

<body ONLOAD="OnLoad()">
```

Note: Do not include this Javascript function and call in the Table of Contents file.

- 2 Search each file for percent signs (%), for example in table width specifications. Except for the %s string substitution command that appears in the OnLoad function definition at the beginning of each file, precede each % sign with another % sign as an escape character. For example:

```
<table border="0" width="80%%">
<tr>
<td width="50%%"></td>
<td width="50%%"></td>
</tr>
</table>
```

- 3 Search each file for exclamation points (!). To include an exclamation point in the text of the trial protocol, insert an additional explanation point as an escape character. For example:

```
<p><b>\Warning!!<b><p>
```

- 4 Use HTML formatting tags as necessary to create a document whose structure is clear and easy to understand. The following formatting conventions are suggested:

- Code the protocol title as an `<h1>` tag. For example:

```
<h1>PF-206 Clinical Drug Trial</h1>
```

- Code the name of each protocol section as an `<h2>` tag. For example:

```
<h2>Objectives of Study</h2>
```

Creating a documentation definition XML file

The documentation definition XML file specifies the RefNames of the HTML files used in creating trial documentation and specifies the locations of those files for the MedML Installer utility. Additionally, this file characterizes the trial documentation files by type and provides a count of links so that the MedML Installer utility can reserve processing space.

Note that the first and last tags in the file must be `<MEDMLDATA>` and `</MEDMLDATA>`. To create a documentation definition file, use the following tags for each trial document:

- 1 Use the `<DOCUMENTATION>` and `</DOCUMENTATION>` start and end tags. To define a complete trial protocol, use one `DOCUMENTATION` tag. To define a complete set of CRF Help, use another `DOCUMENTATION` tag. Both definitions can be part of the same file.
- 2 Assign a RefName to the document using the `REFNAME` tag.

For more information, see:

- *Examples of documentation definition XML files* (on page 473).
- MedML online Help.

Examples of documentation definition XML files

The files associated with the DOCUMENTATION definitions are available in the InForm user interface after you:

- 1 Load the DOCUMENTATION definitions into the trial database using the MedML Installer utility. For more information, see the *Utilities Guide*.
- 2 Associate the DOCUMENTATION definitions with a study version. For more information, see *Trial documentation and study versions* (on page 479).

Trial protocol documentation definition

This example illustrates the definitions used to load the files in a trial protocol into the trial database. The files included with this set of DOCUMENTATION definitions appear in the Documentation window when a user clicks the Documentation button and then the Protocol tab in the InForm user interface.

```
<MEDMLDATA>

<DOCUMENTATION REFNAME="Protocol"
  DOCNAME="Protocol Guide"
  DOCTYPE="BOOKMARKDOC"
  HELPTTEXT="Click here to view Protocol Guide"
  TOC=".\\StudyDoc\\TOC.htm"
  TOCLINKS="25">
  <DOCBODY REFNAME= "OBJECT" FILENAME=".\\StudyDoc\\Objectives.htm"
LINKS="0"/>
  <DOCBODY REFNAME= "DESIGN" FILENAME=".\\StudyDoc\\StudyDesign.htm"
LINKS="9"/>
  <DOCBODY REFNAME= "DROPOUT" FILENAME=".\\StudyDoc\\DropoutDrug.htm"
LINKS="10"/>
  <DOCBODY REFNAME= "INCLEXCL" FILENAME=".\\StudyDoc\\InclExclCriteria.htm"
LINKS="3"/>
  <DOCBODY REFNAME= "TESCHEDULE" FILENAME=".\\StudyDoc\\TESchedule.htm"
LINKS="0"/>
  <DOCBODY REFNAME= "MEDRESTRICT" FILENAME=".\\StudyDoc\\MedRestrict.htm"
LINKS="1"/>
  <DOCBODY REFNAME= "PLACEBO" FILENAME=".\\StudyDoc\\PlaceboRunIn.htm"
LINKS="2"/>
  <DOCBODY REFNAME= "WEEK-4" FILENAME=".\\StudyDoc\\Week-4.htm"
LINKS="1"/>
  <DOCBODY REFNAME= "WEEK-2" FILENAME=".\\StudyDoc\\Week-2.htm"
LINKS="2"/>
  <DOCBODY REFNAME= "WEEK0" FILENAME=".\\StudyDoc\\Week0.htm"
LINKS="7"/>
  <DOCBODY REFNAME= "TREATMENT"
FILENAME=".\\StudyDoc\\TreatmentPhase.htm" LINKS="5"/>
  <DOCBODY REFNAME= "WEEK1" FILENAME=".\\StudyDoc\\Week1.htm"
LINKS="1"/>
  <DOCBODY REFNAME= "WEEK2" FILENAME=".\\StudyDoc\\Week2.htm"
LINKS="2"/>
  <DOCBODY REFNAME= "WEEK4" FILENAME=".\\StudyDoc\\Week4.htm"
```

```

LINKS="2"/>
  <DOCBODY REFNAME= "WEEK5" FILENAME=".\\StudyDoc\\Week5.htm"
LINKS="5"/>
  <DOCBODY REFNAME= "WEEK8" FILENAME=".\\StudyDoc\\Week8.htm"
LINKS="1"/>
</DOCUMENTATION>

</MEDMLDATA>

```

CRF Help documentation definition

This example illustrates the definitions used to load the files that define CRF Help into the trial database. The files included with this set of DOCUMENTATION definitions appear in the Documentation window when a user clicks the Documentation button and then the CRF Help tab in the InForm user interface.

```

<MEDMLDATA>

<DOCUMENTATION REFNAME="Study"
  DOCNAME="CRF Help"
  DOCTYPE="BOOKMARKFAQDOC"
  HELPTEXT="Click here to view information on completing CRFs"
  TOC=".\\StudyGuide\\SG_TOC.htm"
  TOCLINKS="10">
  <DOCBODY REFNAME="SCREENHELP"
FILENAME=".\\StudyGuide\\ScreeningLog.htm"/>
  <DOCBODY REFNAME="ELIGHELP" FILENAME=".\\StudyGuide\\Eligibility.htm"/>
  <DOCBODY REFNAME="SSHELP" FILENAME=".\\StudyGuide\\Signs_Symptoms.htm"/>
  <DOCBODY REFNAME="DEMHELP" FILENAME=".\\StudyGuide\\Demographics.htm"/>
  <DOCBODY REFNAME="VSHHELP" FILENAME=".\\StudyGuide\\VitalSigns_BP.htm"/>
  <DOCBODY REFNAME="CHESTHELP"
FILENAME=".\\StudyGuide\\ECG_Chest_XRay.htm"/>
  <DOCBODY REFNAME="PEHELP" FILENAME=".\\StudyGuide\\Physical_Exam.htm"/>
  <DOCBODY REFNAME="PEW8HELP"
FILENAME=".\\StudyGuide\\Week8_Physical_Exam.htm"/>
  <DOCBODY REFNAME="CMHELP" FILENAME=".\\StudyGuide\\Con_Meds.htm"/>
  <DOCBODY REFNAME="SCHELP"
FILENAME=".\\StudyGuide\\Study_Completion.htm"/>
</DOCUMENTATION>

</MEDMLDATA>

```

Visit Calculator and Sample Case Book documentation definition

This example illustrates the definitions used to make the Visit Calculator available in the Documentation window when a user clicks the Documentation button and then the Visit Calculator tab in the InForm user interface.

```
<MEDMLDATA>
<DOCUMENTATION REFNAME="Visit"
  DOCNAME="Visit Calculator"
  DOCTYPE="VISITDOC"
  HELPTEXT="Click here to view Patient Visit Calculator"
</DOCUMENTATION>
```

This example illustrates the definitions used to make a sample Case Book available in the Documentation window when a user clicks the Documentation button and then the Sample Book tab in the InForm user interface.

```
<MEDMLDATA>
<DOCUMENTATION REFNAME="CRB"
  DOCNAME="Sample Book"
  DOCTYPE="CRBDOC"
  HELPTEXT="Click here to view Sample Case Book forms"
</DOCUMENTATION>
</MEDMLDATA>
```

Creating a Table of Contents file

- 1 In a text file, list the topics to which you want users to be able to link.
- 2 For each topic, create a link to the corresponding file as defined in the DOCUMENTATION definition for the trial document or CRF Help. For more information, see *Linking between document files* (on page 477).
- 3 Save the file in the location specified in the TOC element of the DOCUMENTATION definition.

Example of a Table of Contents file

The following Table of Contents has an entry for each section defined in the documentation definition for CRF Help. For more information, see *CRF Help documentation definition* (on page 474).

```
Screening Log
Eligibility (Run-In)
Signs and Symptoms
Demographics
Vital Signs
ECG/Chest X-Ray
Ophthalmologic Examination
Physical Examination
Physical Examination - Week 8
Concomitant Medication
Study Completion
```


Linking between document files

To create a link between normal HTML files, you use an `<A>` tag with an `HREF` attribute in the file that contains the link. The `HREF` attribute is set to the name of the file to which you want to link (the target file). For example, to create a link on the text “road map” to a file called `roadmap.htm`, you would enter the following HTML text in the file where the link appears:

```
<a href="roadmap.htm">road map</a>
```

If the link is to an HTML bookmark within the file, you add the bookmark name to the `HREF` attribute:

```
<a href="roadmap.htm#appwindows">application windows</a>
```

To link between HTML files defined by `DocBody` tags, you still use an `<A>` tag with an `HREF`, but you replace the name of the target file with the following string:

```
"javascript:top.location.href='./PFTS.dll?pfSessionCode=!s!&pfCommand=TrialMgr_Help&pfSiteID=!i!&pfDocDisp=!d!&pfDocName=DOCUMENTATION_REFNAME&pfDocBodyName=DOCBODY_REFNAME&pfDocLink=FILE_BOOKMARKNAME&pfTimeout=1'"
```

In the string, replace the italicized variables with the following information:

- **DOCUMENTATION_REFNAME** — RefName of the document in which the file appears, as defined in a `DOCUMENTATION` tag; for example:

```
&pfDocName=AboutInform
```
- **DOCBODY_REFNAME** — RefName of the target file to which you want to link, as defined in a `DOCBODY` tag; for example:

```
&pfDocBodyName=ROADMAP
```
- **FILE_BOOKMARKNAME** — Name of the HTML bookmark within the file. Use this part of the string only when you are linking to a location within the target file. To link to the top of the target file, leave this part of the string out. An example of the `&pfDocLink` part of the string follows:

```
&pfDocLink=appwindows
```

Example of a complete link

The following text defines a link to the section of the protocol describing the treatment phase of the trial. Online, the link appears as follows:

Treatment Phase

```
<a href="javascript:top.location.href='./PFTS.dll?pfSessionCode=!s!&pfCommand=TrialMgr_Help&pfSiteID=!i!&pfDocDisp=!d!&pfDocName="Protocol Guide"&pfDocBodyName="TREATMENT"&pfDocLink=appwindows&pfTimeout=1'">Treatment Phase</a>
```

Note: The names of HTML files and of bookmarks within those files must not contain spaces or special characters.

Only links between files defined with `DocBody` tags require the special string. To create a link within a single file, use a simple HTML `<A HREF>` tag to a bookmark.

Example of a Table of Contents file with links

The following Table of Contents file defines the link between each Table of Contents entry and a file specified in the DOCUMENTATION definition for CRF Help. Elements that appear in the DOCUMENTATION definition are shown in bold.

```

<html>
<head>
<title>Study Guide - Table of Contents</title>
</head>
<body>

<h3>Contents</h3>

<p class="toc1"><a
href="javascript:top.location.href='./PFTS.dll?pfSessionCode=!s!&pfCommand=Trial
Mgr_Help&pfSiteID=!i!&pfDocDisp=!d!&pfDocName=Study&pfDocBodyName=SCREENHELP&pfT
imeout=1'">Screening Log</a></p>

<p class="toc1"><a
href="javascript:top.location.href='./PFTS.dll?pfSessionCode=!s!&pfCommand=Trial
Mgr_Help&pfSiteID=!i!&pfDocDisp=!d!&pfDocName=Study&pfDocBodyName=ELIGHELP&pfTim
eout=1'">Eligibility (Run-In)</a></p>

<p class="toc1"><a
href="javascript:top.location.href='./PFTS.dll?pfSessionCode=!s!&pfCommand=Trial
Mgr_Help&pfSiteID=!i!&pfDocDisp=!d!&pfDocName=Study&pfDocBodyName=SSHELP&pfTimeo
ut=1'">Signs and Symptoms</a></p>

<p class="toc1"><a
href="javascript:top.location.href='./PFTS.dll?pfSessionCode=!s!&pfCommand=Trial
Mgr_Help&pfSiteID=!i!&pfDocDisp=!d!&pfDocName=Study&pfDocBodyName=DEMHELP&pfTime
out=1'">Demographics</a></p>

<p class="toc1"><a
href="javascript:top.location.href='./PFTS.dll?pfSessionCode=!s!&pfCommand=Trial
Mgr_Help&pfSiteID=!i!&pfDocDisp=!d!&pfDocName=Study&pfDocBodyName=VSHELP&pfTimeo
ut=1'">Vital Signs</a></p>

<p class="toc1"><a
href="javascript:top.location.href='./PFTS.dll?pfSessionCode=!s!&pfCommand=Trial
Mgr_Help&pfSiteID=!i!&pfDocDisp=!d!&pfDocName=Study&pfDocBodyName=CHESTHELP&pfTi
meout=1'">ECG/Chest X-Ray</a></p>

<p class="toc1"><a
href="javascript:top.location.href='./PFTS.dll?pfSessionCode=!s!&pfCommand=Trial
Mgr_Help&pfSiteID=!i!&pfDocDisp=!d!&pfDocName=Study&pfDocBodyName=EYEHELP&pfTime
out=1'">Ophthalmologic Examination</a></p>

<p class="toc1"><a
href="javascript:top.location.href='./PFTS.dll?pfSessionCode=!s!&pfCommand=Trial
Mgr_Help&pfSiteID=!i!&pfDocDisp=!d!&pfDocName=Study&pfDocBodyName=PEHELP&pfTimeo
ut=1'">Physical Examination</a></p>

<p class="toc1"><a
href="javascript:top.location.href='./PFTS.dll?pfSessionCode=!s!&pfCommand=Trial
Mgr_Help&pfSiteID=!i!&pfDocDisp=!d!&pfDocName=Study&pfDocBodyName=PEW8HELP&pfTim
eout=1'">Physical Examination - Week 8</a></p>

<p class="toc1"><a
href="javascript:top.location.href='./PFTS.dll?pfSessionCode=!s!&pfCommand=Trial
Mgr_Help&pfSiteID=!i!&pfDocDisp=!d!&pfDocName=Study&pfDocBodyName=CMHELP&pfTimeo
ut=1'">Concomitant Medication</a></p>

<p class="toc1"><a
href="javascript:top.location.href='./PFTS.dll?pfSessionCode=!s!&pfCommand=Trial
Mgr_Help&pfSiteID=!i!&pfDocDisp=!d!&pfDocName=Study&pfDocBodyName=SCHELP&pfTimeo
ut=1'">Study Completion</a></p>

</body>
</html>

```

Trial documentation and study versions

Trial documentation has versions in the same way that a trial does. The version of a set of trial documentation is tied to a study version through the VERSIONDESCRIPTION attribute of the following MedML elements:

- **STUDYVERSION**—Assigns a version to a set of FORMSETS (visits).
- **STUDYVERSIONDOC**—Assigns a version to a set of trial documentation and associates the trial documentation with a specific study version.
- **STUDYVERSIONSITE**—Applies a study version and the trial documentation that is associated with it to a site.

If the trial documentation changes after you initially install it, you must update the study version of the trial if you want the new version of the documentation to apply only to new patients.

```
<MEDMLDATA>
<STUDYVERSION
  VERSIONDESCRIPTION="1"
  STUDYNAME="Hypertension Study"
  PROTOCOL="Protocol XYZZY">
  <FORMSET REFNAME="Visit1" TITLE="Week -4"
    LANGUAGE="English"TYPE="Visit"
    SCHEDULED="true"ORDER="1">
    <FORMREF REFNAME="DEM"/>
    <FORMREF REFNAME="FH"/>
  </FORMSET>
  <FORMSET REFNAME="Visit2" TITLE="Week -2"
    TYPE="Visit"SCHEDULED="true">
    <FORMREF REFNAME="VS"/>
  </FORMSET>
</STUDYVERSION>
</MEDMLDATA>
```

The VERSIONDESCRIPTION attribute of the trial documentation must match the VERSIONDESCRIPTION attribute of the study version.

```
<MEDMLDATA>
<!-- Documents -->
<STUDYVERSIONDOC VERSIONDESCRIPTION="1" DOCREFNAME="Protocol"
  ORDER="1"/>
<STUDYVERSIONDOC VERSIONDESCRIPTION="1" DOCREFNAME="Study"
  ORDER="2"/>
<STUDYVERSIONDOC VERSIONDESCRIPTION="1" DOCREFNAME="Visit"
  ORDER="3"/>
<STUDYVERSIONDOC VERSIONDESCRIPTION="1" DOCREFNAME="CRB"
  ORDER="4"/>
</MEDMLDATA>
```

The VERSIONDESCRIPTION attribute of the study version and trial documentation must match the VERSIONDESCRIPTION attribute of each site where the versions are updated.

```
<STUDYVERSIONSITE VERSIONDESCRIPTION="1" SITEMNEMONIC="PF"  
ACCEPTDATE="6/30/1998" />  
<STUDYVERSIONSITE VERSIONDESCRIPTION="1" SITEMNEMONIC="BID"  
ACCEPTDATE="6/30/1998" />  
<STUDYVERSIONSITE VERSIONDESCRIPTION="1" SITEMNEMONIC="BCH"  
ACCEPTDATE="6/30/1998" />  
<STUDYVERSIONSITE VERSIONDESCRIPTION="1" SITEMNEMONIC="MGH"  
ACCEPTDATE="6/30/1998" />  
<STUDYVERSIONSITE VERSIONDESCRIPTION="1" SITEMNEMONIC="BWH"  
ACCEPTDATE="6/30/1998" />
```

Creating CRF Help

CRF Help consists of a set of HTML files that users can link to from CRF screens.

Step	Description	More information
1	Create one or more HTML files containing CRF data entry help. One file per CRF is suggested.	<i>Modifying CRF data entry help files</i> (on page 481).
2	Create an XML file that defines the CRF Help in a Documentation tag and as many DocBody tags as there are HTML CRF Help files. If users will be able to create FAQ entries, use BOOKMARKFAQDOC as the DOCTYPE, otherwise, use BOOKMARKDOC.	<ul style="list-style-type: none"> • <i>Creating a documentation definition XML file</i> (on page 472) • <i>Examples of documentation definition XML files</i> (on page 473) • MedML online Help
3	Create a Table of Contents file that contains each CRF for which you are providing CRF Help.	<i>Creating a Table of Contents file</i> (on page 476)
4	Link each item in the Table of Contents file to the appropriate CRF Help HTML file.	<i>Linking between document files</i> (on page 477)
5	Include the Documentation RefName in a StudyVersionDoc definition	<i>Trial documentation and study versions</i> (on page 479)
6	Process the CRF Help with the MedML Installer utility to load the files into the InForm database.	<i>Utilities Guide</i>

Modifying CRF data entry help files

After you create the HTML files for the CRF Help, make the following modifications:

- 1 Replace the </head> and <body> tags at the beginning of each file with the following Javascript function definition and call:

```
<script language="Javascript">
function OnLoad()
{
%s
}
</script>
</head>

<body ONLOAD="OnLoad()">
```

Note: Do not include this Javascript function and call in the Table of Contents file.

- 2 Search each file for percent signs (%), for example in table width specifications. Except for the %s string substitution command that appears in the OnLoad function definition at the beginning of each file, precede each % sign with another % sign as an escape character. For example:

```
<table border="0" width="80%">
<tr>
<td width="50%"></td>
<td width="50%"></td>
</tr>
</table>
```

- 3 Search each file for exclamation points (!). To include an exclamation point in the text of the CRF help, insert an additional explanation point as an escape character. For example:

```
<p><b>Warning!!<b><p>
```

- 4 Code the name of each CRF data item as an <h3> tag with a bookmark, created with an tag, on the item name. When the document is displayed in the Document window, this name appears in the pulldown list of item names to which users can attach FAQ information. Note that the bookmark name may not contain embedded spaces. For example:

```
<h3><a name="Item1">Item 1 &nbsp;&nbsp;Gender</a></h3>
```

- 5 If users will be able to create FAQ entries, create FAQ placeholders. For details, see *Creating a FAQ placeholder* (on page 482).
- 6 If help is available for rules on data items, create Rule Help placeholders. For details, see *Creating a Rule Help placeholder* (on page 483).
- 7 Use HTML formatting tags as necessary to create a document whose structure is clear and easy to understand. The following formatting conventions are suggested:

Code the name of the CRF as an <h1> tag. For example:

```
<h1>Demographics</h1>
```

Code the name of the CRF section as an <h2> tag. For example:

```
<h2>Smoking History</h2>
```

Creating a FAQ placeholder

For each item on which users will be able to create a FAQ entry, create an unordered list that contains no list items but contains the following text:

```
!Q item_bookmark_name!
where item_bookmark_name is the bookmark name used in the <a name> specification
in the <h3> tag for the data item. Note that this name may not include embedded
spaces. For example, if the heading for Item 1 is:
<h3><a name="Item1">Item 1 &nbsp;&nbsp;Gender</a></h3>
```

Code the FAQ placeholder as follows:

```
<ul>!Q Item1!</ul>
```

The FAQ placeholder should follow the help text on the data item and precede the Rule Help placeholder, if one exists. See *Example: Creating a FAQ placeholder* (on page 483).

Example: Creating a FAQ placeholder

Example

The following example shows a section of an HTML file that defines a CRF Help topic with a FAQ placeholder:

```
<h3><a name="Item4">Item 4&nbsp;&nbsp;&nbsp;Height</h3>
  <ul>
    <li>Height must be measured.<br></li>
    <li>Measure height with shoes off. </li>
    <li>Report height in either inches or centimeters.<br>
      (The entry will be converted to the desired<br>
      units for report purposes.) </li>
  </ul>
  <ul>!Q Item4!</ul>
</a>
```

Creating a Rule Help placeholder

For each item that has a rule for which rule help has been created, create an unordered list that contains no list items but contains the following text:

```
!R item_REFNAMEPath!
```

where `item_REFNAMEPath` is the `REFNAMEPath` used to address the data item. For example, if the rule is on the Weight item of the Demographics form, code the Rule Help placeholder as follows:

```
<ul>!R Visit1:DEM:DEM:WEIGHT!</ul>
```

The Rule Help placeholder should follow the help text on the data item and precede the FAQ placeholder. See *Example: Creating a Rule Help placeholder* (on page 484).

Creating rule help text

The text of the help that appears in the location where you create a rule help placeholder is part of the rule definition or part of the definition of a rule context. For information on defining rule help text, see *Specifying rule properties* (on page 294). For information on defining context-specific rule help text, see *Creating a context* (on page 300).

Example: Creating a Rule Help placeholder

The following example shows a section of an HTML file that defines a CRF Help topic with both a Rule Help and a FAQ placeholder:

```
<h3><a name="Item4">Item 4    &nbsp;  Height</h3>
  <ul>
    <li>Height must be measured.<br></li>
    <li>Measure height with shoes off. </li>
    <li>Report height in either inches or centimeters.<br>
      (The entry will be converted to the desired<br>
      units for report purposes.) </li>
  </ul>
  <ul>!R Visit1:DEM:DEM:HEIGHT!</ul>
  <ul>!Q Item4!</ul>
</a>
```


Revising trial documentation

You may periodically need to revise the documents associated with a trial, including Help documents and sponsor-provided documents.

To implement a new document version:

- 1 Update the files that make up the document.
- 2 If any link counts change, or if you add or remove a file from the document, update the XML file in which you specify the document definition.
- 3 Load the new or updated document files into the database using the MedML Installer to process the document definition XML file.
- 4 If you have made any changes to CRF Help, use the MedML Installer to reload into the database the definition of each Form to which the changed help text applies.
- 5 Create a new StudyVersion to include the updated document by changing the VERSIONDESCRIPTION attribute of the StudyVersion definition.
- 6 Associate the new StudyVersion with each site at which it will be applied by changing the VERSIONDESCRIPTION attribute of the StudyVersionSite tag to match the StudyVersion definition.
- 7 Associate the updated document definition with the new StudyVersion definition by changing the VERSIONDESCRIPTION attribute of the StudyVersionDoc tag to match the StudyVersion definition.
- 8 Use the MedML Installer to load the updated definitions into the database in the following order: StudyVersion, StudyVersionDoc, StudyVersionSite.

For more information, see *Trial documentation and study versions* (on page 479).

APPENDIX A

HTML Tags and Special Characters

In this appendix

Overview	488
HTML formatting tags.....	489
HTML special characters.....	490
Disallowed characters.....	493

Overview

This appendix describes the HTML tags and special characters supported by the InForm application. You can use these tags and characters in any text-based trial component definitions; for example, trial protocols, CRF Help, CRF questions and notes, rule help, and FAQs.

HTML formatting tags

The following tags are available for formatting text:

HTML tags	Used for
	Bold text
 	Line break
<CENTER></CENTER>	Centering text an equal distance from the left and right edges of the document
<I></I>	Italic text
	List items
	Ordered (numbered) lists
<P></P>	Paragraphs
<PRE></PRE>	Preformatted plain text; for example, computer output
<S></S>	Strikethrough text
<STRIKE></STRIKE>	Strikethrough text
	Subscript text
	Superscript text
<TT></TT>	Monospace font
<U></U>	Underlined text
	Unordered (bulleted) lists

HTML special characters

The HTML specification includes numerous character sequences for specifying special characters. When you include HTML special characters in a trial component definition file, the MedML Installer passes the characters along to the database, and they are retrieved and processed by the forms rendering component of the InForm application. To ensure that special characters are rendered correctly across synchronized trials, you must convert the leading “&” of the numeric or symbolic code to the code for the ampersand character. In other words, if you want to use the yen symbol, you should write it as follows:

- &yen;
- or
- &#165;

Note: Special characters entered through the InForm Architect application are automatically converted to the correct HTML form.

Not all special characters in the HTML specification are supported by PDF, the output format in which the PFExport utility exports printable CRFs. A complete list of special character definitions available for use in the InForm application and supported by PDF is shown in the following table:

Character	Description	Numeric Code	Symbolic Code
"	quotation mark	"	"
&	ampersand	&	&
<	less-than sign	<	<
>	greater-than sign	>	>
	non-breaking space	 	
¡	inverted exclamation	¡	¡
¢	cent sign	¢	¢
£	pound sterling	£	£
¤	general currency sign	¤	¤
¥	yen sign	¥	¥
§	section sign	§	§
¨	umlaut(dieresis)	¨	¨
©	copyright	©	©
^a	feminineordinal	ª	ª
«	left angle quote, guillemotleft	«	«
¬	not sign	¬	¬
-	soft hyphen	­	­

Character	Description	Numeric Code	Symbolic Code
®	registered trademark	®	®
ˉ	macron accent	¯	¯
°	degree symbol	°	°
²	superscript two	²	²
¶	paragraph sign	¶	¶
¸	cedilla	¸	¸
º	masculine ordinal	º	º
»	right angle quote, guillemotright	»	»
¼	fraction one-fourth	¼	¼
¾	fraction three-fourths	¾	¾
¿	inverted question mark	¿	¿
°	degree symbol	°	°
Â	capital A, circumflex accent	Â	Â
Ã	capital A, tilde	Ã	Ã
Ä	capital A, dieresis or umlaut mark	Ä	Ä
Å	capital A, ring (Angstrom)	Å	Å
Æ	capital AE diphthong (ligature)	Æ	Æ
Ç	capital C, cedilla	Ç	Ç
Ê	capital E, circumflex accent	Ê	Ê
Ë	capital E, dieresis or umlaut mark	Ë	Ë
Ì	capital I, grave accent	Ì	Ì
Í	capital I, acute accent	Í	Í
Ï	capital I, dieresis or umlaut mark	Ï	Ï
Ð	capital Eth, Icelandic	Ð	Ð
Ñ	capital N, tilde	Ñ	Ñ
Ø	capital O, slash	Ø	Ø
Ù	capital U, grave accent	Ù	Ù
Ú	capital U, acute accent	Ú	Ú
Ü	capital U, dieresis or umlaut mark	Ü	Ü

Character	Description	Numeric Code	Symbolic Code
Ý	capital Y, acute accent	Ý	Ý
Ð	capital THORN, Icelandic	Þ	Þ
ß	small sharp s, German (sz ligature)	ß	ß
ã	small a, tilde	ã	ã
ä	small a, dieresis or umlaut mark	ä	ä
å	small a, ring	å	å
æ	small ae diphthong (ligature)	æ	æ
ç	small c, cedilla	ç	ç
ð	small eth, Icelandic	ð	ð
ñ	small n, tilde	ñ	ñ
ô	small o, circumflex accent	ô	ô
õ	small o, tilde	õ	õ
ö	small o, dieresis or umlaut mark	ö	ö
ø	small o, slash	ø	ø
ù	small u, grave accent	ù	ù
ú	small u, acute accent	ú	ú
û	small u, circumflex accent	û	û
ü	small u, dieresis or umlaut mark	ü	ü
ý	small y, acute accent	ý	ý
þ	small thorn, Icelandic	þ	þ
ÿ	small y, dieresis or umlaut mark	ÿ	ÿ

Disallowed characters

To prevent rendering and other formatting problems:

- Avoid copying and pasting from Microsoft Word into text boxes and query text, as Word can change characters to unicode.
- Do not use the following special characters:

Character	Where not to use
Double quotes	<ul style="list-style-type: none"> • Data entry text box • Form title • Question text • Query answer text • Query text • Rights group name
Single quote	<ul style="list-style-type: none"> • Question text • Query answer text • Query text • Site name • Rights group name
Apostrophe (')	<ul style="list-style-type: none"> • Form title • Question text • Rights group name
\ or \\	Anywhere
> or <	Anywhere; use HTML escape character equivalents: <ul style="list-style-type: none"> • > — &#62; or &gt; • < — &#60; or &lt;
Comma	Rights group name
Percent sign	Rights group name
Superscript and subscript formatting specifications	Elements that will be used in pulldown controls

APPENDIX B

Creating custom reports

In this appendix

About custom reports	496
The Reporting COM object	500
Creating custom text reports.....	506
Creating custom graphs	510
Loading a custom report into InForm application.....	514

About custom reports

The Reporting and Analysis module provides a wide range of trial management and clinical data reports. Additionally, the Ad Hoc Reporting feature of the InForm application enables users to create unlimited custom reports by using Cognos Query Studio and the InForm Trial Management and Clinical Data Models. These features are described in the following InForm application documentation:

- *Reporting and Analysis* guide
- *Installation and Configuration Guide*

In most cases, you will want to create custom reports by using the features of the Reporting and Analysis module.

This appendix describes a custom report development method, using Active Server Pages (ASP) technology, that existed before the Reporting and Analysis module was developed. The purpose of this appendix is to assist users who may be involved in the maintenance of ASP-based reports created with older versions of the InForm application.

What is involved

Before you can create a custom report, you need to know how to use a language like Visual Basic Script (VBScript) or Javascript to create ASPs that define objects and SQL queries. This section explains the interface you must use to define instances of the Reporting COM object of the InForm application, and the specific syntax that must be included in your ASPs. Several examples (written in VBScript) of retrieving and formatting different data are also provided.

Output formats

Once you have determined what information you want to extract, you can display the output as either of the following:

- **Text**—For example, you can write the data directly to an HTML table or as an XML string, and specify formatting,
- **Graphs**—You can choose a pie chart or a horizontal or vertical bar chart layout.

Text output formats

A text report is the simplest type of report to generate. A SELECT statement (generated in ASP code) defines the report query. The Command property of the Reporting COM object executes the query string and builds the report resultset. The GetData object property extracts data points from the query's resultset. The data points of interest are integrated with HTML (via ASP code) to form a tabular report layout. You determine the data points, their location in the table, the table column headings, and so on.

Remember that reports derived from a Customer Defined Database (CDD) automatically deactivate internal filtering performed by the InForm application. Thus, the resultset includes information from all sites—not just those sites the user running the report has privileges to access.

In addition to a simple tabular format, two other text output alternatives are possible using the HTML and XML properties of the Reporting COM object:

- **HTML property**—Displays the entire resultset as a table.
- **XML property**—Returns the complete resultset as an XML string. To display XML data in conjunction with HTML requires that you parse the XML string through the properties and methods of the XML Document Object Model provided through your browser. For more information, see the documentation for your browser.

Graphical output formats

You can also produce custom reports in two graphical output formats: pie charts and bar charts. These formats can help readers better visualize the information you are presenting.

Both pie charts and bar charts use the Reporting COM object to define a report query, extract data from the resultset, and display the chart. The chart's graphic content consists of title(s) for each pie shape or group of bars, legends defining colors, descriptions of pie slices or bars, a title for the report, and so on. The Reporting COM object uses this information to generate the chart and provides the means for you to display it.

Data values, pie charts, and bar charts

To produce a graphic report (that is, pie charts or bar charts) the Reporting COM object must associate data values returned from a query with GROUP and INDEX properties. These properties are independent of the chart type, and can be thought of as the rows and columns of a two-dimensional array, respectively:

- **Pie charts**—Each pie represents a GROUP and the slices of the pie represent an INDEX associated with the group. Thus, if you create a report that summarizes adverse effects by site, you might choose to associate “sites” with the GROUP attribute and “range of possible adverse effects” with the INDEX attribute. The resultset from your query would quantify the occurrences of adverse effects by site. And your ASP program would associate data points of the query with the appropriate GROUP/INDEX values defined in your program.

Each pie would reflect a separate site (a GROUP) and the slices of the pie would represent a particular adverse effect at that site (an INDEX). The size of each slice would indicate the relative number of occurrences of the adverse effect with respect to the total of all occurrences at the site. The color for each slice is determined by the InForm application.

- **Bar charts**—Each group of bars is associated with a specific INDEX and there are as many “groups of bars” as there are INDEXes. (Each group of bars is labeled via an INDEX label. For example, severe, moderate, mild, and so on, might be labels for adverse effects.) The height of a bar represents the GROUP/INDEX data value (that is, the occurrences of the adverse effect at the site) extracted from the report. Colors for each bar are assigned by the software.

If you create a bar chart using the same data assignment as for the pie chart (that is, GROUP properties represent sites and INDEX properties represent adverse effects), you would see the adverse effects (severe, moderate, mild, and so on) listed on the X axis. For each effect, there would appear a group of bars (the sites), and the height of each bar would represent the occurrences of the adverse effect at the site (the GROUP/INDEX data value).

GROUP and INDEX assignments

The pie chart and bar chart graph the same data from different perspectives. For example, if you reverse the assignment of GROUP and INDEX values—assign site data to INDEX properties and adverse effects to GROUP properties—the bar chart would display sites along the X axis, and the occurrences of adverse effects at a site would be represented via each group of bars.

Graphical details of pie charts and bar charts

You also use the GROUP and INDEX properties to define the annotations associated with a bar or pie chart such as labels, graph legends, titles, and so on.

Reporting COM object properties by function

The following table describes all the Reporting COM object properties by function. This will provide a sense of “where and why” each property could be used. Detailed descriptions of the properties can be found in the “The Reporting COM object” section.

When you want to...	Use...	Notes
Define a query for the report	<ul style="list-style-type: none"> • CommandSet • CustomFilterSet • ManagerFilter 	
Define chart type	SetGraphDisplay	Not required for text formats.
Define graph parameters	PieColumnsSetTitle	The PieColumns property applies only to pie charts.
Define graph legends and titles	<ul style="list-style-type: none"> • SetLegendRect • AddLegend • SetGroupLabel 	
Extract data from a query	<ul style="list-style-type: none"> • GetData • RowCount • ColCount 	

When you want to...	Use...	Notes
Extract site and user information	<ul style="list-style-type: none"> • SiteCount • GetSite • MemberCount • GetMember 	
Associate data with the GROUP/INDEX pair	SetGraphValue	Used for either pie or bar charts; not required for text formats.
Display a report	<ul style="list-style-type: none"> • GetGraphBits • ImageHTML • HTML • XML 	GetGraphBits and ImageHTML properties apply to graphical formats; HTML and XML properties apply to text formats.
Define user/connection activities	<ul style="list-style-type: none"> • SetCustomDB • CurrentUserDBUID • IDtoRefName • Filtering 	
Define a date format	<ul style="list-style-type: none"> • UserDate • FormatDate 	

The Reporting COM object

No matter what format you choose for your report, your ASPs must reference the properties of the Reporting COM object defined in the InForm application. The object properties are described in the following table:

Property	Description	Example
Command	Specifies the query (string that specifies an SQL SELECT statement) used to generate a result for the report. The query is executed when this property is set.	rptobj.Command = "select * from patient"
XML	Returns the result of the query (Command property) as an XML string. The XML string returned may have to be parsed (via an XSL style sheet and/or the appropriate methods of the XML Document Object Model) to be viewable in browser.	xmlstr = rptobj.XML
HTML	Returns the result of query (Command property) as an HTML table. Result can be viewed directly in browser. Column headings of HTML table reflect column names specified in query (Command property).	Response.Write rptobj.HTML
RowCount	Returns the number of rows in the result (Command property) as a long integer.	numrows = rptobj.RowCount
ColCount	Returns the number of columns in the result (Command property) as a long integer.	numcols = rptobj.ColCount
GetData(Row, Column)	Returns the data for the specified row/column of the result (Command property) as a Variant data type. Row and Column indexes are long integers and the row/column index is 1-based. Specifying a zero value for a Row or Column index generates an error.	dataval = rptobj.GetData(1,1)

Property	Description	Example
Filtering	<p>Specifies whether filtering should be applied. Note that if you are not connecting to a CDD (you are using the trial database), this property is enabled by default. Conversely, if you connect to a CDD (with the SetCustomDB property) this property is disabled.</p> <p>Filtering, if True, includes only data from sites that the current user has access privileges to view. Filtering should always be False when evaluating results from a CDD (i.e., user privileges are not considered, hence data from all sites is reflected in Result).</p>	<code>rptobj.Filtering = False</code>
SetCustomDB	Specifies a custom DSN (CDD) for reporting. Three string arguments are required that specify the DSN, username, and password, respectively.	<code>rptobj.SetCustomDB "democdd", "democdduid", "democddpid"</code>
UserDate(Date)	Returns a date string formatted via the user's Date format selection (as specified via the InForm application user rights). UserDate accepts a single Date argument.	<code>datestr = rptobj.UserDate(Date)</code>
CurrentUserDBUID	Returns the current user's Database User ID (DBUID) as a long integer.	<code>userid = rptobj.CurrentUserDBUID</code>
FormatDate(Date, sFormat, sSeparator)	<p>Returns a date string formatted according to the sFormat string argument with the separator character specified in the sSeparator string argument. The Date argument specifies the date to be converted.</p> <p>sFormat may be any combination of mm, dd, and yyyy with a separator character in between the values. If the separator character used in sFormat does not match the character specified in sSeparator, the sFormat character prevails.</p>	<code>datestr = rptobj.FormatDate(Date, "dd- mm-yyyy", "-")</code>
IDtoRefName(UserDBID)	Returns the user's RefName given the user's database ID (DBUID). RefName is returned as a string data type. Accepts a long integer as an argument that specifies the DBUID.	<code>userid = rptobj.CurrentUserDBUID refname = rptobj.IDtoRefName(userid)</code>

Property	Description	Example
MemberCount (UserDBID)	Returns the number of members in a manager's group. Result is returned as a long data type. Accepts a long integer as an argument that specifies the user's database ID.	userid = rptobj.CurrentUserDBUID memcount = rptobj.MemberCount(userid)
GetMember(Index)	Returns a long integer that represents the user's database ID pertaining to the index argument. The zero-based index is a long integer used to select a particular member from the list of members. The total number of members may be obtained through MemberCount.	mem = rptobj.GetMember(index)
SiteCount (UserDBID)	Returns the number of sites the user is allowed to access. Result is returned as a long data type. Accepts a long integer as an argument that specifies the user's database ID.	userid = rptobj.CurrentUserDBUID memcount = rptobj.SiteCount(userid)
GetSite(Index)	Returns a long integer that represents the SiteID pertaining to the index argument. The "Zero based" index is a long integer that is used to select a particular site from the list of sites. The total number of sites may be obtained through SiteCount.	mem = rptobj.GetSite(index)
SetCustomFilter strWhereClause	Specifies a WHERE clause to further refine the Result generated via the Connect property. Accepts a string as an argument that specifies the details of the WHERE clause.	rptobj.SetCustomFilter "t_frmae.visitid = visitmap.id"
SetManagerFilter	Specifies in the WHERE clause only users that report to the manager. The user executing this property must be a manager and should have users reporting to him/her. No arguments are required as the WHERE clause is generated automatically.	rptobj.SetManagerFilter "t_frmae.visitid = visitmap.id"

Property	Description	Example
SetGraphDisplay ChartType, AxisStyle, Width, Height	<p>Sets up the graphic engine to generate a particular graphic output style.</p> <p>Arguments are:</p> <p>ChartType—Long data type that defines the type of chart to be produced:</p> <ul style="list-style-type: none"> 0x00020000 (13107210)—Horizontal bar chart 0x00030000 (19660810)—Vertical bar chart 0x00040000 (26214410)—Pie chart <p>AxisStyle—Long data type that defines axes</p> <p>0x01000000 (1677721610)—Style chosen from graph</p> <p>Width—Long data type that defines width of pie chart (in pixels)</p> <p>Height—Long data type that defines height of pie chart (in pixels)</p>	<p>rptobj.SetGraphDisplay 0x0040000, 0x01000000, 1000, 1000</p>
ImageHTML(Width, Height)	<p>Returns a string that specifies an HTML image tag. This property is used to generate the finished graphic image. Accepts two long integer arguments that represent the desired width and height of the image, respectively, in pixels.</p>	<p>strimage = rptobj.ImageHTML(500, 500) Response.Write strimage</p>
GetGraphBits Bits, Width, Height	<p>Returns a binary array that specifies the graphic image. This property is used to generate the finished graphic. Accepts an argument to hold the binary array and two long integer arguments that represent the desired width and height of the image, respectively, in pixels.</p>	<p>rptobj.GetGraphBits Bits, Width, Height Response.BinaryWrite Bits</p>
PieColumns	<p>Set to specify the number of columns of pie charts spaced horizontally across a page to display before wrapping to a new line. For example, assume that data in a report is segregated by site, there are four sites, and each pie chart represents a site. If PieColumns is set to 1 then the resulting chart will consist of four vertically oriented (rows of) pie charts. If PieColumns is set to 2, the output will consist of two rows with two pie charts per row.</p>	<p>rptobj.PieColumns = 2</p>

Property	Description	Example
SetTitle strReportTitle	Set to specify the title of the report. Accepts a single string argument that specifies the report title.	rptobj.SetTitle "Title of My Report"
AddLegend Group, Label	<p>Bar Chart—AddLegend associates a text label with a GROUP number. The charting software will also assign a color to the GROUP number.</p> <p>Pie Chart—AddLegend defines the title of the pie specified by the GROUP number.</p> <p>AddLegend accepts two arguments. A long integer (Group) that specifies GROUP number and a string that represents a title (pie chart) or a legend (bar chart).</p>	rptobj.AddLegend 0, "Bar Chart Legend or Pie Chart Title"
SetGroupLabel Label, Index	<p>Bar Chart—SetGroupLabel defines the text description associated with an INDEX (i.e., a group of bars).</p> <p>Pie Chart—SetGroupLabel defines the text label, used in the legend. The charting software will also assign a color to the INDEX number.</p> <p>SetGroupLabel accepts two arguments:</p> <ul style="list-style-type: none"> • A string that represents a legend (pie chart) or an INDEX description (bar chart) • A long integer (INDEX) that specifies an INDEX number. 	rptobj.SetGroupLabel "Index description (Bar) or legend (Pie)", 0
SetGraphValue Index, Group, Value	Specifies a single data value for the pie chart or bar chart. The INDEX argument, a long integer, correlates to the Index value in the SetGroupLabel property, and the GROUP argument, also a long integer, correlates to the Group value in the AddLegend property. The Value argument, a Double numeric type, specifies the data value.	rptobj.SetGraphValue 1, 0, GetData(row, col)

Property	Description	Example
SetLegendRect Left, Top, Right, Bottom	Sets the size of the legend rectangle by specifying its left, top, right, and bottom coordinates. The graphing software uses these parameters to determine the size and position of the legend. The actual position and size are somewhat dependent on a number of software and computer parameters. Make sure the legend that appears is appropriately displayed and scale these parameters as needed.	rptobj.SetLegendRect 1, 1, 100, 100

Note: To perform multiple queries, you must define a separate SQL statement for each Report object instance.

Creating custom text reports

Let us first look at creating an ASP to extract data and produce a straightforward text report. We will begin by reviewing the basic statements you must include in the ASP, then we'll filter the original query to extract specific data. Finally, we will perform a few simple formatting operations.

ASP basics for text reports

To produce a custom text report, an ASP must:

- Create a report object.
- Specify an SQL query to extract information from a database.
- Set the SQL statement as the value of the Command property of the Reporting COM object.

The following example shows a working ASP that contains all of these elements and clears the report object when done:

```
<%
'Create the report object
set obj = Server.CreateObject("PFReportObj.PFRptObject")
'Define an SQL statement
str = "SELECT DCV_PatientState.PATIENTID, "&_
"PF_SITE.SITEMNEMONIC, "&_
"PF_REVISIONHISTORY.PFTIMESTAMP FROM "&_
"DCV_PatientState, "&_
"PF_REVISIONHISTORY, DCV_SITE PF_SITE "&_
"WHERE DCV_PatientState.PATIENTID = "&_
"PF_REVISIONHISTORY.DBUID "&_
"AND DCV_PatientState.PATIENTREVISIONNUMBER = "&_
"PF_REVISIONHISTORY.REVISIONNUMBER "&_
"AND DCV_PatientState.SITEID = PF_SITE.SITEID"

%>
```

Example 1

This script defines a report object, specifies an SQL query, and sets the Command property equal to the SQL statement, and clears the report object when done.

```
'Set the SQL statement as the value of the Command property
obj.Command = str

'Clear the report object
objXML.Save(Response)
set obj=nothing
```

Example 2

The SQL statement shown in Example 1 selects patient information for all sites that the current user can access, and filters it through a simple WHERE clause. The information is extracted by default from the InForm application database. To extract data from a customer-defined database (CDD) instead, you need to specify the DSN for the CDD, and a valid user ID and password, as shown in the following example (remember that user IDs and passwords must contain all alphabetic or all alphanumeric characters and begin with a letter. Do **not** use all numeric characters):

```
'Set a custom database connection
obj.SetCustomDB "pfstrep", "pfstuid", "pfstpid" 'DSN, userid, pwd
```

This statement specifies that the data should come from a customer-defined database (i.e., not the InForm application database). Note that the data is not filtered.

Filtering data queries

The InForm application automatically filters the data returned to any query by limiting it to information that the current user is allowed to access. In other words, if a site manager asks to see all patient data for all sites, the information returned will be filtered by the manager's predefined rights. A nurse might have different rights, and would see different results from the same query.

Any custom filters that you define will be applied on top of the built-in InForm application filter, unless you turn filtering off. To do that, and thereby allow any user to see *all* information returned, add this statement to the ASP:

```
'Turn off filtering
obj.Filtering = false
```

The basic InForm application filtering algorithm expects a column in the SQL query that can be referenced as PF_Site.SiteID. Filtering is performed by altering the SQL with further conditions upon this column. If this column does not exist, an error occurs.

Now let us add some custom filters. One common report request might be to see specific data for all users associated with a particular manager. The InForm application contains a predefined WHERE clause that you can add to apply this filter easily:

```
'Filter the data based on Manager ID by adding the
'following predefined WHERE clause:
'"DCV_REPCRAUSERS.MANAGERID = CurrentUserID"
obj.SetManagerFilter()\
```

As another exercise, let us specify a custom WHERE clause to filter data based on a patient's current state (screening, enrolling, etc.). The following statement specifies patients who have been screened but not yet enrolled:

```
'Set a custom WHERE clause
obj.SetCustomFilter "DCV_PatientState.STATE < 4"
```

Sizing a table

When you are producing a table, you might find it useful to know the number of rows and columns in the table in advance. To determine these dimensions, you could count the rows and columns returned to your query, and then write out the data:

```
'Get the number of rows in the record set
nRow = obj.RowCount

'Get the number of columns in the query
nCol = obj.ColCount
'Write the specified number of rows and columns
Response.Write "<br>Row = " & CStr(nRow)
Response.Write "<br>Column = " & CStr(nCol)
```

Associating data with a label

With the following statements, you can get the current user's ID and RefName, and label them in the output:

```
'Get and write the current user's ID
DBUID = obj.CurrentUserDBUID
Response.Write "<br>User ID = " & CStr(DBUID)
'Get and write the user's refname
strUserName = obj.IDToRefName(DBUID)
Response.Write "<br>User Name = " & strUserName
```

Retrieving specific fields

To report data from an exact location in the record set returned by the query, you must specify an index to the position in the record set. Note that the indices for the GetData property start from 1.

```
'Get the value in the first record, third column
'(PF_REVISIONHISTORY.PFTIMESTAMP)
myData = CDate(obj.GetData(1,3))
```

Requesting all users for a particular manager

You might want to see a list of all users who report to a particular manager. To do this, you must specify the manager's DBUID in the query. If you specify 0 for the DBUID, the current user's ID will be used as the manager's. If you want to use the sample database provided with the InForm application, log on as the user **cdm**.

```
'Get all members of a specific manager's group
nBound = obj.MemberCount(DBUID)

'Write the list of members
Response.Write "<br>The user " & strUserName & " has the &_
following members: "
for i=0 to nBound-1
ID = obj.GetMember(i)
Response.Write "<br>id= " & CStr(id) & " name = " &_
obj.IDToRefName(id)
next
```

Requesting all sites for a particular user

To discover all the sites that a user can access, add the following statements:

```
'Get list of sites the user can access
nBound = obj.SiteCount(DBUID)
'Write the list of sites
for i=0 to nBound-1
ID = obj.GetSite(i)
Response.Write "<br>site id= " & CStr(id)
next
```


Simple text formatting

Unless you add explicit instructions for formatting the results of your queries, no data will be displayed. You can format the information as either an XML string or as an HTML table.

To write data out as an XML string, add the following statement to your ASP:

```
'Write the record set as an XML string
Response.Write obj.XML
```

To write data directly to an HTML table, add the following statement:

```
'Write the record set directly to an HTML table
Response.Write obj.HTML
```

Specifying date formats

Here are two examples of how you can manipulate date output. First let us specify that we want to produce the date in the form *mm-dd-yyyy*, using a hyphen (-) as a separator. In the report, we will show both the original and the new formats, with brief text explanations:

```
'Format myDate as mm-dd-yyyy, use - as separator
strDate = obj.FormatDate(myDate, "mm-dd-yyyy", "-")
Response.Write "<br>The original date is: " & CStr(myDate)
Response.Write "<br>Formatted date is: " & strDate
```

Next, let us get and write the date in the format stored for an individual user. This format corresponds to the local style that the user has chosen for displaying dates. For example, a user in the United States would typically write dates in the form *mm-dd-yyyy*, while a European user would write *dd-mm-yyyy*.

```
'Get and write the user-formatted date
strDate = obj.UserDate(myDate)
Response.Write "<br>The user date is: " & strDate
```

Creating custom graphs

In this section you will learn how to produce custom reports that display their output in graphs. Rather than drawing data directly from the InForm application database, the examples in this section use hypothetical values to create graphs of different modes of transportation (planes, trains, automobiles, etc.)

ASP basics for reports with graphics

To produce a custom report in the form of a graph, you must create an ASP that:

- Changes the HTTP header so the browser will know that image data is being passed, rather than HTML.
- Creates the report object.
- Specifies an initial size for the graph.
- Generates an HTML image tag that references the image source, or saves the image in binary format.
- Sends the image tag or the binary information to the browser.

The following example shows these elements. Note that it also explicitly declares the variables that will be used, and clears the report object when done.

```
<%  
Response.Expires = -1  
Response.Buffer = TRUE  
'Clear out the existing HTTP header information and  
'change the HTTP header to reflect that an image is  
'being passed (rather than HTML)  
Response.Clear  
Response.ContentType = "IMAGE/JPEG"  
'Create the report object  
set obj = Server.CreateObject("PFReportObj.PFRptObject")  
'Initialize the graph  
'(nAxisStyle is determined by the InForm application's 'underlying charting  
tool)  
obj.SetGraphDisplay nChartType, nAxisStyle, nSize, nSize  
'Get an HTML image tag  
strImage = obj.ImageHTML nSize, nSize  
'Write the image to the browser  
Response.Write(strImage)  
set obj = nothing  
>%
```

Filtering data for graphs

You apply filters to graphs in the same way you apply them to text reports. Remember to turn off the InForm application built-in filtering if you do not want to restrict your report output to only the information that the current user is allowed to access.

Formatting graphics

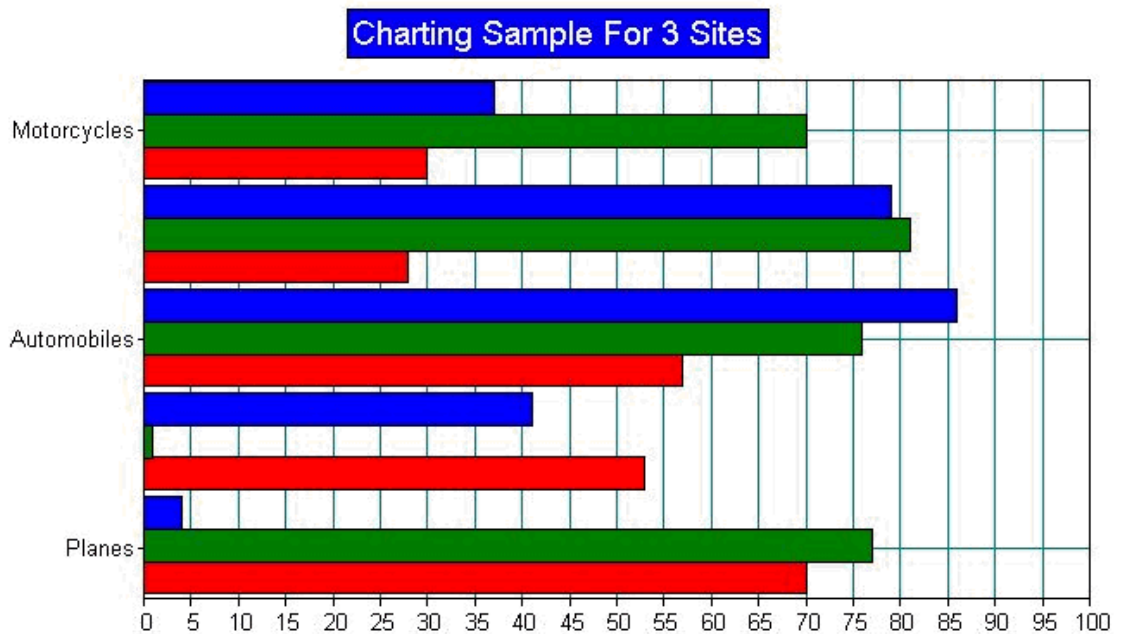
The first decision you need to make about formatting is what kind of graph you want to display. The following chart types are predefined in the InForm application:

To create a...	Specify this value for the chart type...
Horizontal bar chart	&H20000
Vertical bar chart	&H30000
Pie chart	&H40000

To determine what type of graph to display, the following statements check for the value of nChartType. If no value is specified, a pie chart is displayed by default.

```
'Specify the type of graph to display
nChartType = &H40000
If IsEmpty(Request.QueryString("Pie")) or
Request.QueryString("Pie")=0 Then nChartType = &H30000
If Request.QueryString("H")>0 Then nChartType = &H20000
```

The following figure shows an example of a horizontal bar chart:

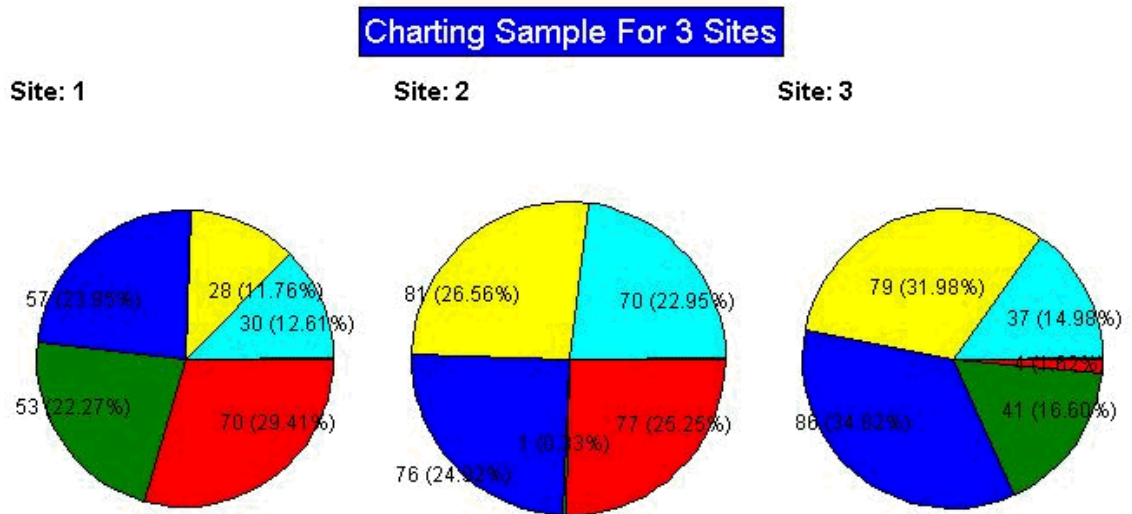


Sizing the display area

When you produce reports as pie charts, you will want to consider how to space the graphs in the display to make them easy to read. For example, if you want to produce pie charts for several sites, make sure that you allow enough room to display a reasonably sized chart for each site:

```
'Multiply the number of columns by an arbitrary number of
'pixels to determine column width
If nChartType = &H40000 AND nCols>3 Then
nSize = nCols*300
Else
nSize = 900
End If
```

The following figure shows three pie charts spaced evenly across the browser window:



Wrapping output to a new line

To specify how many graphs should be displayed on one line before the output wraps to another line, add statements similar to these (this step is necessary only for pie charts):

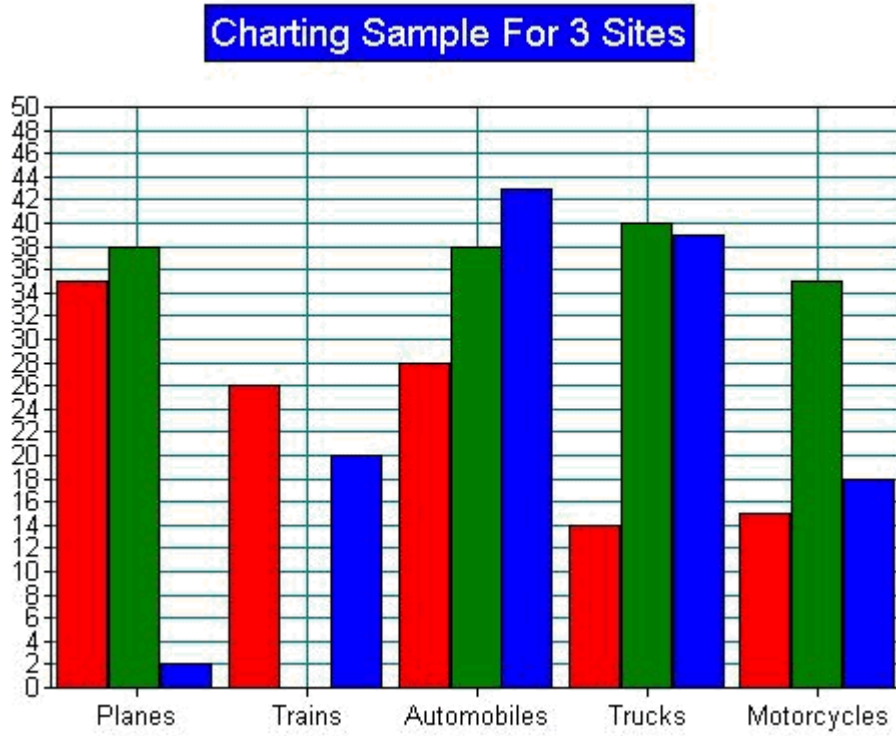
```
'Set the number of pie columns before wrapping to
'the next line
obj.PieColumns = nCols
obj.SetTitle "Charting Sample For " & nSite & " Sites"
For i=0 To nSite-1
```

Creating labels and legends

For both pie charts and bar graphs, labels can make the information represented in the graphs easier to understand. The following statements add labels, and a legend explaining them, to a pie chart. Notice that the legend for the pie chart is actually the subtitle for the individual charts. The group labels become the master legend since the grouping belongs to all pies.

```
obj.AddLegend i, "Site: " & i+1
obj.SetGraphValue 0, i, Int(rnd*100)
obj.SetGraphValue 1, i, Int(rnd*100)
obj.SetGraphValue 2, i, Int(rnd*100)
obj.SetGraphValue 3, i, Int(rnd*100)
obj.SetGraphValue 4, i, Int(rnd*100)
Next
obj.SetGroupLabel "Planes", 0
obj.SetGroupLabel "Trains", 1
obj.SetGroupLabel "Automobiles", 2
obj.SetGroupLabel "Trucks", 3
obj.SetGroupLabel "Motorcycles", 4
'Determine the size of the legend by specifying the position of
'its Left, Top, Right, and Bottom corners, in that order:
x = nSize
obj.SetLegendRect x,x,x+1000,x+1000
```

The following figure shows a vertical bar chart with a corresponding legend displayed beneath the chart:



Loading a custom report into InForm application

When you have created a custom report ASP file, you must:

- Define the report as an InForm application object by referencing it with a REPORT MedML tag in an XML file.
- Load the report definition into the database by processing the XML file with the MedML Installer tool.

This section describes how to use the REPORT MedML tag. For information on how to use the MedML Installer tool, see *InForm Utilities Guide*.

Creating an XML file for custom reports

To create an XML file for custom reports:

- 1 Start and end the file with a <METADATA> and </METADATA> tag.
- 2 For each custom report you want to load, create one REPORT tag. This tag has the following syntax:

```
<REPORT
  [ UUID=" id " ]
  ASPFILENAME=" name "
  ASPTEXT=" code "
  QUERYFILENAME=" name "
  QUERYTEXT=" text "
  [ SETUPFILENAME=" name " ]
  [ SETUPFILETEXT=" name " ]
  [ TITLE=" name " ]
  [ DESCRIPTION=" text " ]
  [ REPORTTYPE=" ADMIN | NORMAL " ]
  [ DATASOURCE=" text " ]
  [ DATASOURCEUSER=" text " ]
  [ DATASOURCEPASSWORD=" text " ]
  [ LANGUAGE=" name " ] />
```

- 3 Assign the following values to the REPORT tag attributes:

Attribute	Value
UUID	Leave blank. A blank UUID indicates that the definition is for a custom report.
ASPFILENAME or ASPTEXT	Name of the ASP file defining the report, or the actual ASP text. One of these is required.
QUERYFILENAME or QUERYTEXT	Name of an SQL file defining the patient data retrieval needed for the report, or the actual SQL statements. Optional. These attributes are for documentation only.
SETUPFILENAME or SETUPFILETEXT	Name of an SQL file defining report-specific tables, views, indexes, and stored procedures that are necessary to create the report, or the actual SQL statements. Optional. If specified, use one or the other of these attributes.

Attribute	Value
TITLE	Title of the report. This appears as a selectable link on the Custom tab and in the title bar of the report when it is generated online.
DESCRIPTION	Description of the report, for documentation only.
REPORTTYPE	Blank or NORMAL.
DATASOURCENAME, DATASOURCEUSER, DATASOURCEPASSWORD	Name, user name, and password of the ODBC data source used to access report data, if you are using a database other than the InForm application database—for example, a CDD.
LANGUAGE	Language used to display the caption. Optional. English is the default.