

IBM Global Services



DataInterchange Client User's Guide

Version 4 Release 1

NOTE: Before using this information and the product it supports, be sure to read the general information under “Notices” on page 405.

Third Edition (January 2002)

This edition replaces document number SB34-2010-02.

© Copyright IBM Corporation 1998, 2002. All rights reserved.

Government Users Restricted Rights - Use, duplication, or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.



Contents

.....

To the reader	xiii
Who should read this book	xiii
How this book is organized	xiii
How to send comments	xiv
Part 1. Introduction	1
Chapter 1. Installation and setup	3
Installing DataInterchange Client	3
Requirements	4
Accessing the DataInterchange host database	6
Client-server mode	6
Stand-alone mode	7
Middleware	8
DataInterchange Client databases	8
Config	8
System database	9
Configuration alternatives	9
Single user, stand alone	9
Multi-user, stand alone	10
Multi-user, client/server, all DB2 for MVS	11
Configuring systems	11
Defining additional single-user databases	12
Defining a multi-user database	13
Recommended configuration	14
Multi-user, multi-server with both stand alone and client/server access	14
Running DataInterchange Client	16
Installing EDI standards	16
Moving to DataInterchange Client	17
Moving setup profiles and Trading Partner profiles	17
Control Strings	18
Migrating data between versions and releases	19
Naming convention changes for DataInterchange Client	21
Host functions not available in the DataInterchange Client interface	21

Chapter 2. The DataInterchange Client interface	23
Using windows	24
List windows	24
Editor windows	27
Selecting commands	28
Menus	28
Tool bars	31
Shortcuts	34
Performing common file management tasks	35
Viewing an item	35
Copying an item	35
Editing an item	36
Renaming an item	36
Deleting an item	37
Printing an item	37
Using editor window grids	38
Working with multiple systems	39
Understanding window title bars	40
Setting preferences	41
Setting window preferences	41
Setting message log preferences	41
Selecting the system color	42
Selecting list window options	42
Customizing field tags	43
Viewing control and status bars	43
Getting help	44
Accessing online help	45
Using DataInterchange Client help	45
Using the Message Log	47
Viewing messages	47
 Chapter 3. Setup for translating data	 49
 Chapter 4. Export/Import	 51
About export/import	51
Export/import file specifications	52
Exporting	52
Exporting to an export/import file	52
Exporting to other systems	54
Importing from an export/import file	56
Importing a DTD file	57
Specifying referenced and associated types	58
Mailbox	58
Network profile	59
Application defaults profile	59
Continuous receive profile	60
Network Security profile	62
Trading Partner profile	62
Data format dictionary	64
Data format	64
EDI Standard dictionary	64
EDI Standard transaction	65

Envelope standard	65
Mapping	66
Control Strings	67
Part 2. Setup	69
Chapter 5. Activity Log profiles	71
About Activity Logs	71
Purpose	71
Setup overview	72
Creating Activity Log profiles	73
Chapter 6. Application Defaults profiles	75
About Application Defaults	75
Setup overview	76
Creating Application Defaults profiles	77
Chapter 7. Continuous Receive profiles	79
About Continuous Receive profiles	80
Setup overview	80
Creating Continuous Receive profiles	82
Using MQSeries with Continuous Receive	82
Chapter 8. CICS Performance profiles	85
About CICS Performance profiles	85
Purpose	85
Setup overview	86
Creating CICS performance profiles	87
Chapter 9. Envelope profiles	89
About Envelope profiles	89
Setup overview	89
Creating Envelope profiles	91
UN/EDIFACT (E) Envelope profile	92
ICS (I) Envelope profile	92
UN/TDI (T) Envelope profile	93
UCS (U) Envelope profile	93
X12 (X) Envelope profile	93
Chapter 10. Language profiles	95
About Language profiles	95
Setup overview	96
Chapter 11. Mailbox profiles	99
About mailbox profiles	99
Setup overview	99
Creating Mailbox profiles	101

Chapter 12. MQSeries Queue profiles	103
About MQSeries Queue profiles	103
Setup overview	104
Creating MQSeries Queue profiles	105
Chapter 13. Network profiles	107
About Network profiles	108
Setup overview	109
Creating Network profiles	110
Chapter 14. Network Commands profiles	113
About Network Commands	113
Setup overview	114
Creating Network Commands profiles	115
Chapter 15. Network Security profiles	117
About Network Security profiles	117
Setup overview	118
Creating Network Security profiles	120
Chapter 16. User Exit profiles	121
About User Exit profiles	121
Setup overview	122
Creating User Exit profiles	123
Part 3. Trading Partners	125
Chapter 17. Trading Partners	127
About Trading Partner profiles	127
Purpose	127
Setup overview	128
Creating Trading Partner profiles	130
Understanding processes and rules	132
Understanding minimal trading partners	133
Specifying usages and rules	136
Viewing usages and rules	136
Creating usages and rules	137
Creating contacts	141
Adding a contact	142
Part 4. Mapping	143
Chapter 18. Data formats	145
Creating a data format	146
Understanding how your application data is structured	146
Filling out a data format worksheet	151
Use the DataInterchange Client data format component editors	154
Using the data format editors	154
Accessing data format editors	154

Using the Data Format Dictionary editor	156
Creating a Data Format Dictionary	156
Importing a COBOL Copybook	157
Using the Data Format Record ID Information Editor	159
Creating a data format record ID information profile	159
Using the Data Format Editor	161
Creating a data format	162
Using the Data Format Loop Editor	164
Creating a loop	164
Using the Data Format Record Editor	166
Creating a record	167
Using the Data Format Structure Editor	169
Creating a structure	170
Using the Data Format Field Editor	171
Creating a field	172
Navigating Data Format Component Editors	173
Reusing data format components	176
Data types for data formats	177
 Chapter 19. Extensible Markup Language (XML)	183
Accessing XML editors	184
Using the XML Dictionary Editor	185
Creating an XML dictionary	185
Importing a DTD file	186
Using the DTD Editor	186
 Chapter 20. EDI Standards	189
About standards	190
Envelopes	190
Transactions	190
Using the EDI standards editors	191
Accessing EDI standards editors	192
Using the EDI Standard Dictionary Editor	193
Creating an EDI Standard dictionary	193
Using the EDI Standard Transaction Editor	195
Creating a transaction	195
Using the EDI Standard Segment Editor	197
Creating a segment	198
Using the EDI Standard Data Element Editor	199
Creating a data element	199
Using the Code List Editor	202
Creating a code list	202
Using the Envelope Standard Editor	203
Editing an envelope standard	203
Using Envelope Control String list window	206
 Chapter 21. Creating a map	207
Getting started	207
Data transformation maps vs send and receive maps	209
Creating a data transformation map	210
Creating a send or receive map	210

Chapter 22. Data transformation mapping	211
Using the Map Editor	211
Starting the Map Editor	212
Utilizing the Map Editor	213
Creating a data transformation map	213
Using the Map Editor	219
Utilizing the Map Command window pane	223
Specifying qualification	224
Qualifying repeating simple and compound elements	224
Advanced mapping techniques	228
Translation Tables	229
Creating the tables	230
Specifying map rules	231
Applying the minimal trading partners concept	231
Viewing map rules	232
Creating a map rule	232
Editing map rules	234
Copying map rules	234
Compiling control strings	235
Viewing compiled control strings	236
Defining Global Variables	236
Migrating a map to a different source or target document	237
 Chapter 23. Send and Receive mapping	 239
Using the Map Editor	239
Starting the Map Editor	240
Utilizing the Map Editor	241
Creating a send or receive map	241
Utilizing the Mapping Data Element Editor	249
Setting an Application Control Key	253
Specifying qualification	255
Qualifying loops and segments	255
Qualifying data elements	259
Using accumulators	262
Adding an accumulator to a map	263
Using literals and mapping commands	264
Adding a literal or mapping command to a map	265
Advanced mapping techniques	266
Translation Tables	269
Creating the tables	270
Specifying send and receive usages	272
Applying the minimal trading partners concept	272
Viewing usages	273
Creating a send or receive usage	273
Editing send and receive usages	276
Copying send or receive usages	276
Defining generic send usages	277
Defining generic receive usages	278
Compiling control strings	278
Viewing compiled control strings	279
Mapping hierarchical loops	279
Creating fixed-to-fixed maps	280

Migrating a map to a new standard	281
Client migration option	281
Part 5. Administration	283
Chapter 24. Queries	285
About queries	285
Purpose	285
Setup overview	286
Working with queries	287
Creating a query	287
Using filters in queries	289
Editing a query	291
Copying a query	292
Deleting a query	292
Running a query	292
Chapter 25. Reports	295
About reports	295
Setup overview	295
Working with reports	296
Creating a report	296
Editing a report	297
Deleting a report	298
Printing or previewing a report	298
Chapter 26. Transaction Store	301
About Transaction Store	301
Client overview	301
Host Setup	302
Using Transaction Store queries	303
Using Transaction Store reports	306
Chapter 27. Event Logs	307
About event logs	307
Viewing event logs	307
Appendix A. Using the 841 transaction set	309
The BIN segment ID	309
Length of the BIN segment	310
The EFI segment	310
Send processing for the binary segment	311
Mapping data from a file to a binary segment	311
Format specifications	312
Examples	314
Mapping an application field to a binary segment	315
Receive processing for the binary segment	316
Mapping data from a binary segment to a file	316
Mapping data from a binary segment to an application field	319

Appendix B. Data Transformation mapping commands and functions	321
Map Variables	321
Naming Variables	322
Literals	322
Comments	323
Keywords	323
Specifying a Path	323
Forward and Reverse References	324
Data Types Supported by DataInterchange mapping commands and functions	324
Expressions	325
Logical Operators	326
Comparison Operators	326
Arithmetic Operators	327
Unary Operators	327
Order of Precedence	327
Assignment	328
Conditional Commands	328
Commands	329
Functions	332
Message Properties	350
Target Document Properties	350
EDI Envelope Standard Generic Properties	350
EDI Envelope Standard Specific Properties	351
Appendix C. Advanced send and receive mapping	355
Using accumulators	355
Using literals	357
Using literals for send mapping	357
Segment creation for send mapping	357
Using literals for receive mapping	358
Format of literal data	358
Accumulator literals	359
Conditional processing literals	359
Literal keywords	360
Named variables	370
Expressions	372
Boolean operators	373
Comparison operators	373
Arithmetic operators	373
Unary operator	375
Special operators	375
Date conversion special operators	377
Order of precedence	380
DI variables	380
Mapping techniques for literal keywords	382
Examples of using literal keywords and named variables	384
Control data literals	392
Mapping specific service segment fields (receive only)	392
Mapping generic service segment fields (receive only)	393
Mapping service segment fields (send only)	395
Validation during mapping	396

Hierarchical loops	398
The HL segment	399
Preparing hierarchical loops	400
Literal keywords for the HL segment	404
Notices	405
Trademarks and service marks	406
Glossary	407
Index	415



To the reader

This book describes the setup and use of the DataInterchange Client 4.1 interface in creating and maintaining DataInterchange profiles, data formats, maps, and EDI standards.

Who should read this book

This book is intended for three types of readers: DataInterchange system administrators, translation analysts, and business managers responsible for trading relationships.

How this book is organized

This book has the following five parts:

- Part 1. Introduction

This part contains information on installing and configuring DataInterchange Client. It also includes information on how to use the DataInterchange Client interface and its export/import functions. This part is intended for all readers.

- Part 2. Setup

This part contains information on how to set up all your operational profiles. Most of the operational profiles are used for fine-tuning operations. The only required operational profiles are the Mailbox profile and the Network profile. This part is intended for DataInterchange administrators who are responsible for system maintenance.

- Part 3. Trading Partners

This part contains information on setting up and maintaining Trading Partner profiles, which contain key business and technical information on the company with which you do business. This part is intended to help business administrators and others who are responsible for maintaining Trading Partner relationships.

- Part 4. Mapping

This part covers the following items required for mapping; data formats, Extensible Markup Language (XML), and EDI Standards. You will also find information about creating maps, and specific instructions for creating data transformation maps, send maps, and receive maps.

- Part 5. Administration

This part contains information on queries, reports, and the Transaction Store database, which can contain images of all your DataInterchange translation activity. Queries control how DataInterchange Client displays information in its windows and allow you to create reports and extract data from the Transaction Store. This part is intended for all readers.

How to send comments

To submit comments and suggestions on this document, send e-mail to docbug@us.ibm.com. Do not send problem reports concerning the product itself to this address.

PART 1. Introduction

Installation and setup

DataInterchange Client is a Microsoft™ Windows-based interface for DataInterchange/MVS and DataInterchange/MVS-CICS. This interface allows you to use a PC to create and maintain DataInterchange profiles, EDI standards, data formats, and maps. It also can be used to load XML DTDs into DataInterchange.

Note that DataInterchange Client does not translate data. All translation functions still take place on the DataInterchange host products. DataInterchange Client provides the ease of use inherent in Windows to make your setup, mapping, and administration faster and easier.

Installing DataInterchange Client

DataInterchange Client has an Install Wizard that guides you through installation. As with any installation, you should begin by closing any applications you have running. Make sure you have enough space on the hard drive on which you are installing DataInterchange Client for the application, EDI standards, and data files. Minimum recommended space is 40 MB.

Requirements

DataInterchange Client is a 32-bit application designed to run on a Pentium PC with 64-MB RAM. Windows 98™, 2000™, ME™, and NT™ are supported.

DataInterchange Client is designed to support any database with Open Data Base Connectivity (ODBC) Version 2 drivers. The following are examples of database software products and database connectivity products that support ODBC. These were chosen for demonstrative purposes to describe various DataInterchange Client configuration alternatives. This is not a complete list.

- Microsoft Access 97
- Microsoft SQL Server 6.x
- IBM DB2 Universal Database Server Version 7.1
- IBM DB2 for MVS or OS/390
- Connectivity to IBM DB2 for MVS or OS/390 via IBM DB2 Connect Version 7.1

◆ To install DataInterchange Client:

1. Insert DataInterchange Client Installation CD ROM.

The install process should start automatically. If it does not, move to step 1a.

- a. Click Start on the menu bar and select Run.

The Run dialog box displays with the cursor in the Open field.

- b. Type x:\client\setup.exe in the Open field, and click OK. x indicates your CD ROM drive.

The Welcome Screen displays as the Install Wizard prepares to install DataInterchange Client.

2. Click Next.

A second Welcome screen displays with messages warning you to exit all Windows programs if you have not done so already and with a notice concerning the copyright.

- a. Click Next.
 - b. Read the copyright notice that displays.
 - c. Click Yes to indicate you have read the notice and agree to its terms.

3. Click Next.

If this is the first time you are installing DataInterchange Client 4.1, the Choose Destination Location dialog box displays. Select your installation location by clicking browse and choosing the appropriate drive and directory. The drive can be selected from the drop-down list at the bottom of the dialog box. The default destination directory is C:\Program Files\IBM\DataInterchange v4.1.



NOTE: Install DataInterchange Client 4.1 in a different directory than previous DataInterchange Client versions.

4. Click Next.

The Setup Type dialog box displays. Choose the type of setup you want from the following choices:

- **Typical:** Select this option the first time you install DataInterchange Client 4.1. This option installs all the common options and creates the software's databases.



ATTENTION: If you are reinstalling DataInterchange Client and you select this option, you receive a warning that Install will overwrite your databases. Install will only overwrite the default 4.1 database files installed through a previous 4.1 install. This option will not overwrite databases installed on a user's database system, such as DB2.

- **Compact:** Select this option to update DataInterchange Client's program file. This option installs the minimum required options.
- **Custom:** Select this option when you want to choose the options to install. This option is recommended for advanced users. If you are reinstalling Client, you should use the Custom setup to avoid overwriting your databases and drivers with the defaults.

5. Choose a Setup Type by clicking the appropriate option button. The default Setup type is Typical.



NOTE: Install DataInterchange Client 4.1 in a different directory than previous DataInterchange Client versions.

6. Click Next.

If you chose a Typical or Compact setup, go to the next step.

If you chose a Custom setup, the Select Components dialog box displays. You can choose whether to install:

- Program files
- Report files
- Default database files
- Database DDL files

Select a component by clicking the check box next to it. The size of each component is listed on the right side of the component window. The total size of the selected components displays in the Space Required field. The space available in the drive you have chosen displays in the Space Available field. It is to your advantage to install DataInterchange Client in a drive that has more than enough space to accommodate it.

7. Click Next.

The Select Program Folder dialog box displays. You can choose a new name for the folder in which the Program icons will be stored. Rename the folder by typing the new name into the Program Folders field, or by selecting from the existing folders displayed below the Existing Folders field; otherwise, the default name is used.

8. Click Next.

The Start Copying Files dialog box displays with a screen showing the setup choices you have made. If you want to change any of these choices, click Back.

9. Click Next.

The Install Wizard begins copying program files. You can stop this process at any point by clicking Cancel.

The Setup Complete dialog box displays. It is recommended that you review the Read Me file each time you install a new version of DataInterchange Client. If you do not want to view the Read Me file, uncheck the Yes, I Want to View the Read Me File check box.

10. Click Finish.

If you chose to view the Read Me file, it displays on your screen.

Select DataInterchange Client v4.1 from the program folder specified during installation to start DataInterchange Client.



NOTE: If you are upgrading DataInterchange Client from a previous release, see “Migrating data between versions and releases” on page 19 for more information.

Accessing the DataInterchange host database

The DataInterchange translator is a host-based application running under MVS or MVS/CICS. OS/390 is also supported in place of MVS. In this publication, MVS will be used to represent both, unless to clarify a difference. The translator requires an MVS-based database at run-time; DB2 is supported. Various factors will determine, or limit, how you reflect changes from the client onto the host. It may be by choice, or you may be limited by your network or database installation. You can use two methods: client-server mode or stand-alone mode.

Client-server mode

In client-server mode, DataInterchange Client accesses the DataInterchange Host database directly using an ODBC link. ODBC is an industry standard for making connections between a variety of software products and databases on different hardware platforms. Access to the host via ODBC is controlled by middleware, as illustrated in Figure 1. Middleware must be obtained and installed separately; it is not included with DataInterchange Client.

For DB2/MVS, we have used various middleware solutions that enable ODBC access. See “Middleware” on page 8 for more information. If you are using DB2 for MVS, consider client-server mode. The net result is real-time updating on the host from DataInterchange Client.

See “Client-Server Configurations” on page 7.

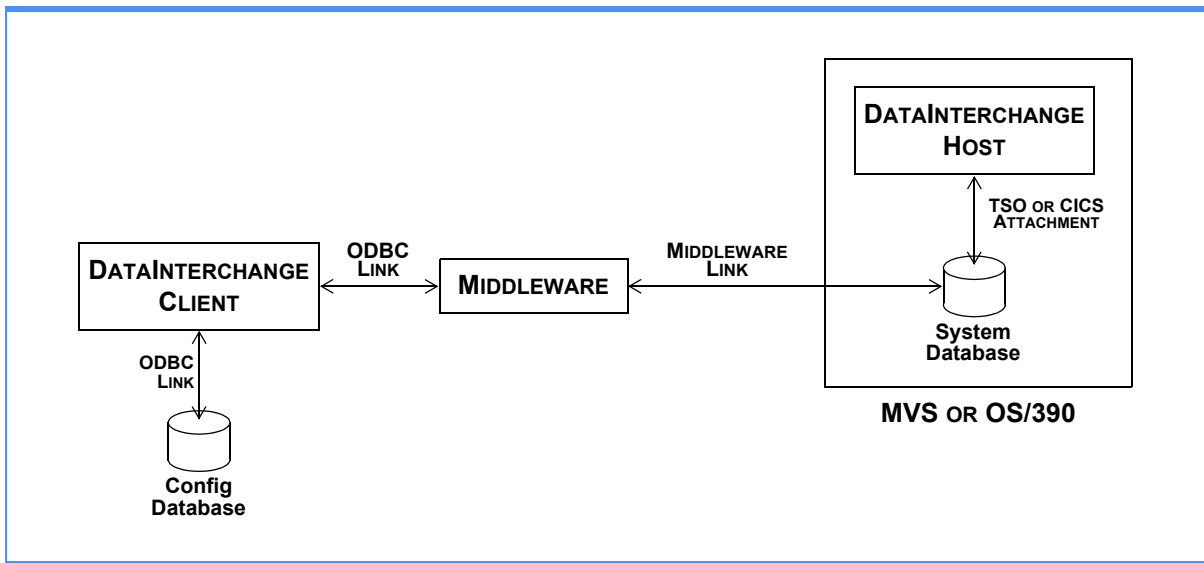


Figure 1. Client-Server Configurations

Stand-alone mode

In stand-alone mode, you make the connection with the host database through DataInterchange Client's Export/Import functions, which are described in Chapter 4 on page 51. Use this method if you cannot establish a client-server connection to the host.

When in stand-alone mode, you transfer a file to the host from the client by exporting the data into an export/import (EI) file. Then utilize file transfer software, which you must obtain separately, to upload the file to the host using the ASCII and CRLF options. On the host, you import the EI file into the DataInterchange Host database using the host's export/import functions, as illustrated in Figure 2.

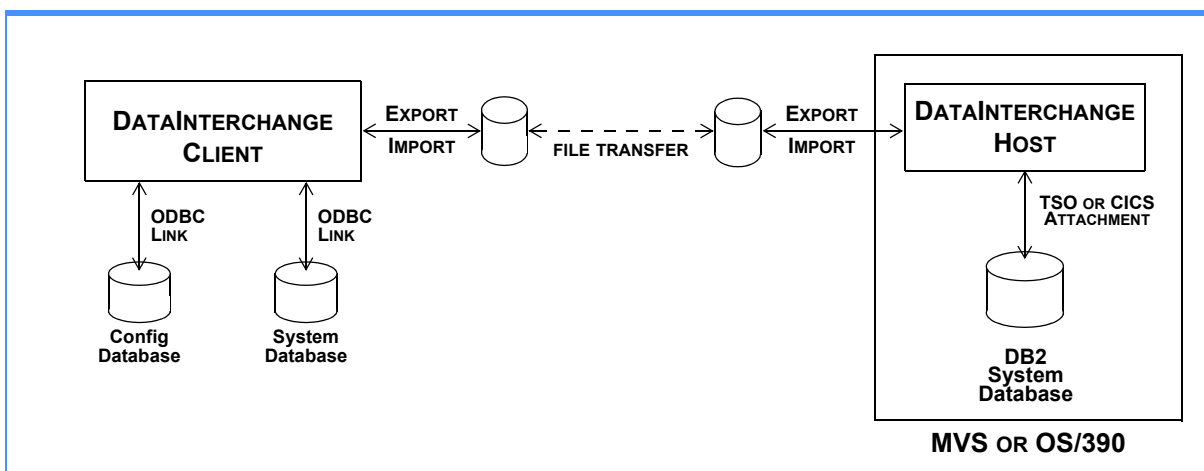


Figure 2. Stand-Alone Configuration

Middleware

The concept of middleware is broad and complex, encompassing many databases on many platforms. The complexity is somewhat reduced for our purposes, in that our main focus is on accessing DB2 on MVS. Even so, there are still many solutions by various software manufacturers that support ODBC connectivity to DB2 on MVS; either direct access from the client PC or through a gateway. There are numerous protocols and configuration options. Your middleware determines what paths are allowed between the PC and MVS. Many products work through NT, UNIX™, AIX, or OS/2 gateways that use the SNA or TCP/IP protocols to talk to MVS, while TCP/IP, NetBEUI or IPX is used to talk to the PCs.

For example, the Personal Edition PC product, DB2 Connect Version 7.1, connects the client workstation directly with DB2 for MVS over SNA networks or TCP/IP, without additional host software. An indirect (gateway) solution is offered in DB2 Connect's V7.1 Enterprise Edition.

DataInterchange Client databases

DataInterchange Client is designed to support any database with ODBC Version 2 drivers. Many database packages developed by various vendors provide this level, or above, of ODBC support -- each has its own database management system (DBMS). In DataInterchange, a database is classified as either single-user or multi-user. A single-user DBMS is one that allows only one user access at a time. These usually reside on each user's PC. A multi-user DBMS, on the other hand, typically resides on a host or server, and provides multiple users shared access to a common database. DB2 for MVS is an example of a host-based multi-user database. Microsoft's SQL Server, and IBM's Universal Database are examples of LAN server databases with ODBC support.

DataInterchange Client partitions its data into two databases. ODBC is used to access both:

- Config
- System

As distributed, DataInterchange Client provides both as Microsoft Access 97 single-user databases, supporting stand-alone operation. Since ODBC is a standard, you are not limited to using the provided Microsoft Access 97 databases. And, since each DataInterchange Client database is independent, each can, theoretically, reside in a different database management system. See "Configuration alternatives" on page 9 for variations and examples.

Config

Config is a local configuration database that stores installation-wide data, such as the list of installed systems, queries, report definitions, and user preferences like the columns that appear in list windows. There can be only one config database defined to the PC running DataInterchange Client. By default, it is named Config41.



NOTE: There is no need to deny privilege to this database because having access rights to the config database does not permit the user to connect to any particular system. Access to system databases is controlled separately.

System database

The system database contains all the tables that hold customization time data and runtime data. The ODBC Data Source Name can be any name of your choice, and any number of systems can be defined. This allows you to have multiple system databases on the same PC, along with multiple system database connections to remote PCs, servers, or hosts. Since ODBC is used to access the system database, the database is not limited to being defined in DB2/MVS. Even a single-user database, such as the distributed Microsoft Access 97 database, can be used for the system database; however, multiple clients cannot concurrently share the same data in this scenario. Meaning that in this configuration they cannot readily share data developed on one another's PCs.

If you have multiple DataInterchange Client users and require concurrent access, then define the system database in a multi-user database management system that supports ODBC. DB2/MVS can be used for this, but is not necessary. Any multi-user database supporting ODBC Version 2 or later can be used. Of course, if you choose not to use the distributed single-user Microsoft Access 97 database, you will need to define new tables. The data definition language (DDL) used to create databases, tables, and indexes is not the same for every DBMS. Sample DDL is distributed in subdirectory DDL when choosing Sample DDL Files during custom installation.



NOTE: You may have, or find, vendor software to aid in converting DDL from one DBMS to another.

Customization time data

Customization time data includes tables related to the customization of EDI standards, data formats, and maps.

Runtime data

Runtime data includes data that is required by the DataInterchange Host translator, including control strings, profiles, send and receive usages, map rules, translation tables, code lists, the transaction store, and event logs.

Configuration alternatives

You can configure your database using one of several alternatives, as explained below.

Single user, stand alone

Both databases, Config and System, reside on the client PC, as distributed, using Microsoft Access 97 databases.

- Easiest to set up, lowest cost.
- Updates to the host databases are not updated in real time. Must use export/import to exchange data with DataInterchange Host.
- Must use separate file transfer utility to upload and download export/import files between MVS and the PC.
- Must use export/import to exchange data with other DataInterchange Clients.

This is a good alternative if there is only one user of DataInterchange Client, or for initial testing.

Setup Example

1. Install DataInterchange Client (Typical) on the PC of each user.
2. Establish Export/Import procedures. Always use tagged export format when exchanging objects between host and client.
3. Establish file transfer procedures.



NOTE: Check your file transfer options; ASCII and CRLF are necessary. Allow for long, variable-length records: a variable-record format (RECFM=V or VB), a record length of 8152 (LRECL=8152), and block size of 8156 (BLKSIZE=8156).

4. If multiple DataInterchange Clients are involved, establish external controls:
 - a. Establish a naming convention so as not to overlay one another's work.
 - b. Establish procedures for sharing trading partners. For translation purposes, a trading partner who is common between DataInterchange Client users must be shared on the host so that you do not have two different names for the same trading partner on the host.

Multi-user, stand alone

Config resides on the DataInterchange Client PC as distributed using Microsoft Access 97 single-user database. The system database is shared on a server other than the host.

- Relatively easy to set up.
- All data is shared with other DataInterchange Client users.
- Updates to the host database is not real-time. Must use export/import to exchange data with DataInterchange Host.
- Must use file transfer utility, obtained separately, to upload and download export/import files between MVS and the PC.

Setup Steps

The following describes the steps for using DB2 Universal Database on Windows NT. These steps may be different for other databases and/or servers. TCP/IP connectivity between clients and NT server is assumed.

1. Install DataInterchange Client (Typical) on PC of each EDI user.
2. Establish Export/Import procedures. Always use tagged export format when exchanging objects between host and client.
3. Establish file transfer procedures.



NOTE: Check your file transfer options; ASCII and CRLF are necessary. Allow for long, variable-length records: a variable-record format (RECFM=V or VB), a record length of 8152 (LRECL=8152), and block size of 8156 (BLKSIZE=8156).

4. Install DB2 Connect Personal Edition on each client PC.

5. Install DB2 Universal Database on the server.
6. Create and load the system database on the server.
7. Create an ODBC data source for the system database on each client PC. Choose a name other than DIClient41Dev as this ODBC data source name is already taken by the distributed Microsoft Access 97 database.
8. Authorize users to access all system database tables on the server. Users will require SELECT, INSERT, UPDATE, and DELETE privileges on all tables defined on the server.

Multi-user, client/server, all DB2 for MVS

Config resides on the DataInterchange Client PC as distributed using Microsoft Access 97 single-user database. The system database is shared on DB2 for MVS.

- Data is shared with other DataInterchange Client and Host users in real-time.
- The DataInterchange Client rename action is limited because of DB2 restrictions.

This is a good alternative for DB2 customers who have many people working together on projects.

Setup steps

This configuration describes the steps for using IBM DB2 Connect V7.1.

1. Install DataInterchange Client (Typical) on the PC of each user.
2. Install IBM DB2 Connect V7.1 on gateway server. Follow the same steps as for Multi-user, Client-server example above.
3. Install IBM DB2 Connect for MVS on each client PC.
4. The ODBC data source name for the DB2/MVS subsystem is registered for you by DB2 Connect's Client Configuration assistant.
5. Grant user privileges to the DB2/MVS tables. The ability to change the system database from the client requires SELECT, INSERT, UPDATE, and DELETE privileges on a subset of DI/MVS DB2 tables.

Configuring systems

Many users set up more than one system. You may, for example, have a test system for testing some of the new maps you create. Your production system may be an active system that handles business transactions. On the host, these systems may even be separate installations of DataInterchange. DataInterchange Client allows you to manage multiple systems with a single installation.

Each system has its own database tables. For example, you can have a map named MAP1 in both test and production systems. Although you can only use one system at a time on DataInterchange Host, you can work in more than one system at a time with DataInterchange Client. For example, you can edit maps in test and production systems, simultaneously.

Each system database must be defined as a unique ODBC Data Source Name (ODBC DSN) in order for DataInterchange Client to access the system. The particulars of creating ODBC DSNs depend on the database management system. The Windows ODBC Manager maps Data Source Names (DSNs) to ODBC drivers. ODBC Manager is found in Settings/Control Panel.

DataInterchange Client is shipped with one system called Development. The ODBC DSN associated with it is DIClient41Dev. This DSN is added automatically during Typical DataInterchange Client installation, and it uses Microsoft Access 97 and points to the supplied Microsoft Access 97 database file DIClient41Dev.mdb. The Development system is ready to go for single-user, stand-alone mode after a Typical installation.



NOTE: DataInterchange Client distributes one other database file: config41.mdb. This has a DSN of Config41 and is also automatically added during Typical installation. There is only one config DSN per DataInterchange Client installation, not one per system. Config is necessary before accessing any system.

Whether using DataInterchange Client in stand-alone or client-server mode, there are three basic steps for creating a new system:

1. Create the system database and load the default data.
2. Create the ODBC DSN for the database.
3. Define a new system in DataInterchange Client using the new Data Source Names.

Defining additional single-user databases

As an example, suppose we want to add another single-user, stand-alone System called Test.

Setup steps

1. Copy the file c:\Program Files\IBM\DataInterchange Client v4.1\DIClient41Dev.mdb to DIClient41Dev2.mdb.



NOTE: Config41.mdb need not be copied since it contains tables common to all systems.

2. Create the ODBC data definition for the new database. Do the following:
 - a. From the Windows Control Panel select the ODBC icon. On some Windows systems this icon may be found within the Administrative Tools icon. The ODBC Data Sources window displays.
 - b. Select the User DSN tab, and then click Add.
 - c. Select the Microsoft Access 97 driver, and then click OK.

- d. To create a new system database entry, do the following:
 - 1) Enter a data source name, such as Test.
 - 2) Enter a description, such as Test database.
 - 3) Click Select.
 - 4) Select the database you copied to.
3. Set up the System definition in DataInterchange Client:
 - a. Select Systems from the View menu.
The Systems List window displays with a list of current systems.
 - b. Click New.
The Systems dialog box displays.
 - c. Enter a name in the System Name dialog box. The name can contain embedded spaces, and be up to thirty characters long. This is a required field.
 - d. Select MVS-OS/390 or CICS from the server platform field. This identifies the platform the host translator will run on. If the host translator will run on both MVS-OS/390 and CICS, select MVS-OS/390.
 - e. Select the data source name Test from the drop-down list in the Data Source Name field.
This field displays in the database ODBC Information group box.
 - f. Leave the Database Qualifier blank.

TIP: The Database Qualifier is used when connecting to multi-user or host databases. For example, on DB2 for Windows NT Server, this corresponds to the database schema. On DB2 for MVS, this is the high-level-qualifier specified on the host DB2 tables and views.
 - g. To change the color of the window background for this system, click Change. For details on changing system color, see Chapter 2, “Selecting the system color,” on page 42.
 - h. When you are finished, click OK.
 - i. You must restart DataInterchange Client before you can use the new system.
The name of the new system will display in the System drop-down list.

Defining a multi-user database

The particulars of defining multi-user or host databases and creating ODBC DSNs depend on your database management system and its operating system. After you have installed the appropriate ODBC drivers on the PCs that are running DataInterchange Client, you must configure them using the Windows ODBC data source administrator. In Windows, for example, go to Control Panel and double-click the ODBC icon. Some database products provide GUI tools to set up databases or connections and they may have an option to register the ODBC data sources for you. If not, manually use the ODBC Data Source Administrator to add your new DSN. Select

your database driver from the list of available ODBC drivers. Next, name your new DSN and provide any required parameters specific to your driver. Lastly, define your system definition in DataInterchange Client.

Recommended configuration

The following configuration combines elements of a source code control system with a rigidly managed deployment process for changes in a multiple translation server environment. This configuration uses both PC and server databases. It makes extensive use of client-server access, along with the Export to System function within DataInterchange Client to move artifacts, such as maps and DTDs, from system to system.

Multi-user, multi-server with both stand alone and client/server access

Each developer has a local database that resides on their PC. In addition, three server DB2 repositories are shared by developers, testers, and administrators.

- Complex setup, higher cost.
- Sharing between developers is easy because there is a single DB2 repository for all build time artifacts.
- Minimum response time is ensured for developers by keeping items currently in progress in a local database.
- Minimizes potential contamination of test environment by developers by separating the test database from the development database, and allowing only testers to have access to the test database.
- Minimizes potential contamination of the production environment by testers or developers by separating the production database from the test and development databases, and not allowing testers or developers access to the production database.
- Minimizes potential contamination of complex build time artifacts (XML, EDI standard, and Data Format document definitions, along with maps) by not deploying them to the test and production environment. Only runtime artifacts (such as trading partners, usages, rules, and control strings) are deployed, so testers and administrators cannot change build time artifacts.
- Check-out, check-in, and manual deployment (or promotion) of artifacts can be done directly by an authorized user with the DI Client Export to System function.
- Deployment can be automated with the use of the PERFORM EXPORT and PERFORM IMPORT server commands.

This is a good alternative for a medium to large scale B2B (Business to Business) shop.

The following is an illustration of the recommended configuration.

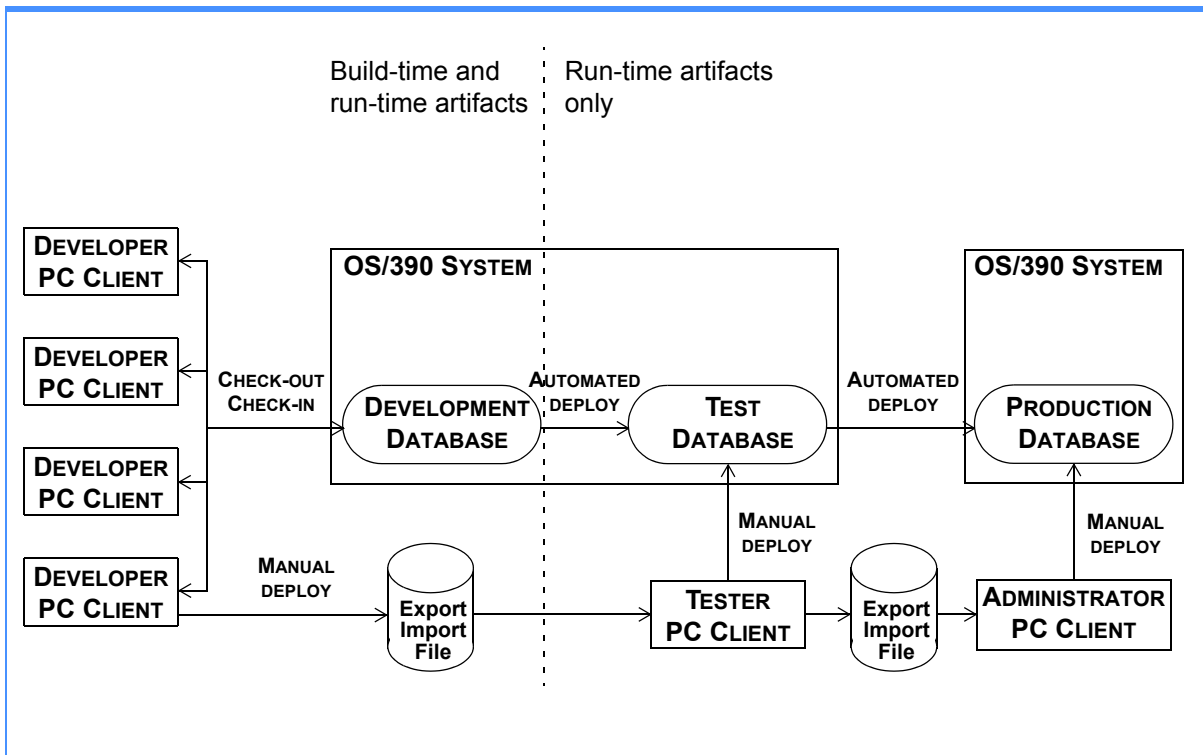


Figure 3. Recommended Configuration

Setup Example

Install DataInterchange Client (Typical) on the PC of each user (developer, tester, or administrator).

- For developers, install the local database and set up their computers for ODBC access to the development database.
- For testers, do not install a local database. Set up their computers for ODBC access to the test database, and if manual deployment will be allowed via Export to System, then also set up access to the development database.
- For administrators, do not install a local database. Set up their computers for ODBC access to the production database and if manual deployment will be allowed via Export to System, then also set up access to the test database.

Establish check-in and check-out procedures for the developers. DataInterchange does not provide a check-in, check-out mechanism, so this must be done manually.

Establish procedures for deploying (or promoting) artifacts. If automated deployment is to be used, write the deployment JCL jobs, or scripts.

Running DataInterchange Client

DataInterchange Client is designed to prevent more than one user from working on the same item at the same time. As a result, you need to log on to DataInterchange Client databases every time you start the software, as follows.

◆ To run DataInterchange Client:

1. Select DataInterchange from the DataInterchange item on the Start menu.



NOTE: If you are using a multi-user or client-server setup, your User ID and Password are determined by your system administrator.

You are now ready to connect to the system that displays in the System drop-down list. To change which system will be accessed, select an alternate system name from the list.



TIP: Use the ODBC Administrator in the Windows Control panel to set up your passwords if you do not want to type your passwords every time you access a system.

You can work on items in different systems at the same time. For example, you can edit a map from the Development system while also editing one from the Test system.

Installing EDI standards

After you have configured your DataInterchange Client system, you must install EDI standards before you can map with EDI standards. DataInterchange Client is shipped with a number of common EDI standards on the CD. You can download the most recent EDI standards from the DataInterchange Web site at <http://edi.services.ibm.com/datainterchange>.

◆ To install the EDI standards from the CD:

1. Insert the CD ROM containing the EDI standards file you want to install.
2. Select Open Import File from the File menu of DataInterchange Client.
3. Select the directory Standards on the CD ROM.
4. Choose one of the following subdirectories:
 - X12 ANSI, ASC, and X12
 - VICS VICS
 - AARV Rail
 - UCS UCS
 - EDIFACT UN/EDIFACT

The available EDI standards display.

5. Select the EDI standard you want to install by clicking its plus sign or double-clicking its folder.

The folder opens and each transaction within the EDI standard displays.



NOTE: You need not install an entire EDI standard. If you prefer, you can install only the transactions you need.

6. Click the transactions you want to install, or click the EDI Standard to install all transactions.

The Import button on the tool bar becomes available.

7. Click Import.

An Execution Status dialog box displays showing your import progress.

Your EDI standards are now installed. Their components appear in the windows you see when you click EDI Standards on the Navigator bar. For details, see Chapter 20, “EDI Standards,” on page 189.

Moving to DataInterchange Client

It is extremely important that you have proper external controls and procedures in place to prevent users from updating the same items using both the host and client interfaces, as that can result in loss of some of your updates.

If you are moving Host format maps, EDI standards, and data formats from a previous release, refer to the *Migration Considerations* publication included with the product.

Moving setup profiles and Trading Partner profiles

Setup profiles and Trading Partner profiles can be accessed from DataInterchange Client. Setup profiles include Mailboxes, Activity Logs, Network profiles, and so on.

Client-server mode

If you have client-server access to the DataInterchange Host databases, you can create new profiles or update existing ones using the DataInterchange Client windows interface. Changes you make using DataInterchange Client are immediately reflected in the host databases when you save your changes.

While you are making changes using DataInterchange Client, other client users are locked out from making changes to the same profile. Likewise, if you are making changes using DataInterchange Host, other DataInterchange Host users cannot make changes to the same profile. However, it is possible for a client user and a host user to make changes to the same profile at the same time. As a result, it is extremely important to have external controls in place to prevent this from happening.

Stand-alone mode

In order to make changes to Setup and Trading Partner profiles in stand-alone mode, you must first export, download, and import them to DataInterchange Client from DataInterchange Host. Make sure that another user is not updating the item on the Host at the same time you are updating it on the Client. If that happens, the Client user will overwrite any changes made by the Host user. As a result, it is extremely important to have external controls in place to prevent this from happening.

◆ To update a profile that currently exists on the host:

1. Export the profile from DataInterchange Host in tagged format.
2. Using a file transfer utility, download the profile to the PC where DataInterchange Client is running.
3. Import the profile into DataInterchange Client.
4. Update the profile using DataInterchange Client.
5. Export the profile from DataInterchange Client in tagged format.
6. Using a file transfer utility, upload the file to the DataInterchange Host.
7. Import the profile into the DataInterchange Host.

Control Strings

The DataInterchange Host translator does not use EDI standards, data formats, or maps directly during translation. Instead, it uses a control string, which includes information about the EDI standard, data formats, and map.

On DataInterchange Client, EDI standards, data formats, and maps are compiled into a control string. A control string contains what is needed of these objects to perform EDI translation on the host.

Following are considerations for compiling maps into control strings in client-server and stand-alone modes:

Client-server mode

When you compile a map in client-server mode, the resulting control string is placed directly in the host database.

Stand-alone mode

In stand-alone mode, the compile option creates a control string in the PC database. You must then export the control string in tagged format, upload the file, and import the data into DataInterchange Host.



NOTE: Check your file transfer options; ASCII and CRLF are necessary. Allow for long, variable-length records: a variable-record format (RECFM=V or VB), a record length of 8152 (LRECL=8152), and block size of 8156 (BLKSIZE=8156).

Migrating data between versions and releases

Use the DataInterchange Client Release Migration function to migrate data from one version of DataInterchange Client to another.

If you are currently using DataInterchange Client 3.1, you must perform the release migration to move the data from release 3.1 to release 4.1.



NOTE: Release Migration will not move DataInterchange 3.1 Host format maps, EDI standards, and data formats. You must import these in Host 4.1. Refer to the *Migration Considerations* publication included with the product.

When planning your release migration, consider the following:

- A database administrator should perform the release migration for shared databases.
- An individual user can perform the release migration for a stand-alone database configuration.
- Migrate runtime data that is contained on the host as part of the DataInterchange 4.1 Host installation process. Runtime data not contained on the host can be migrated using the DataInterchange Client release migration procedure below.
- Migrate customization time data using only DataInterchange Client.
- Migrate configuration data only if you want to save custom queries, reports, or system data. If configuration data is migrated, the system data moves from DataInterchange Client 3.1 to DataInterchange Client 4.1. The first time you use DataInterchange Client 4.1 after migrating configuration data, you must edit the system data and adjust Data Source Names and Qualifiers as needed to reflect DataInterchange 4.1 databases.



NOTE: The DataInterchange Client Release Migration option uses unique data structures from the regular DataInterchange export and import formats, and should not be intermixed with that capability. To use the import capability, the data must have been exported from DataInterchange Client using DataInterchange Client Release Migration option.

To perform a release migration, export the data from DataInterchange Client 3.1 using the following procedure.

◆ To export data:

1. Select DI Client Release Migration from the View menu.

The Release Migration wizard displays.



NOTE: All list windows within DataInterchange Client must be closed.

2. Select Export to create a set of files with data from Systems associated with the DataInterchange Client you are using, and then click Next.

3. Select the type of data you are exporting, and then click Next. Your choices include:
 - System Data - only system data will be processed. System data is runtime data and customization time data.
 - Configuration Data - only configuration data will be processed.
 - Both - system data and configuration data will be processed.
4. Select one or more systems to be used by the release migration process you chose, and then click Next. If you chose Configuration Data in step 3, skip to step 6.
5. Select the system from which data will be exported by the release migration process, and then click Next. Your choices include:
 - Customization Time Data - EDI standards, data formats, and maps
 - Runtime Data - Trading Partners, other profiles, translate tables, and control strings
 - Both - Customization time and Runtime data
6. Type the migration path the data will be exported to. An existing path, from a previous DataInterchange Client 4.1 export, can be selected from the drop-down list. Click Next.
7. Click Finish to begin the process. A status box displays during the export.



NOTE: After you import configuration data, you should review you systems definitions to ensure the settings are correct, especially data source names.

◆ To import data:

1. Select DI Client Release Migration from the View menu on DataInterchange Client 4.1.
The Release Migration wizard displays.



NOTE: All list windows within DataInterchange Client must be closed.

2. Select Import to populate the Client you are using with data from another version or release of the product, and then click Next.
3. Select the type of data you are importing, and then click Next. Your choices include:
 - System Data - only system data will be processed. System data is runtime data and customization time data.
 - Configuration Data - only configuration data will be processed.
4. Select the path containing the migration data, and then click Next.
5. Select the system that the selected migration data source will migrate into, and then click Next. This screen displays only if more than one system is defined in the DataInterchange Client.

6. Select the system into which exported data will be imported. Export data will be filtered to populate the selected system. After you make your selection, click Next. Your choices include:
 - Customization Time Data - EDI standards, data formats, and maps
 - Runtime Data - Trading Partners, other profiles, translate tables, and control strings
 - Both - Customization time and Runtime data
7. Confirm the options you have chosen for the import. To change the selected options, click Back; otherwise, click Finish to initiate the import.



TIP: You can also use Release Migration to save and restore data in a system.

Naming convention changes for DataInterchange Client

In the DataInterchange Client interface, some of the terminology has been changed to reflect generally accepted conventions as well as requests from customers. Note the following changes in terminology:

Table 1. Terminology Changes between DataInterchange Host and DataInterchange Client

DataInterchange Host Terminology	DataInterchange Client Terminology
Application Data Formats (ADF)	Data Formats
Application Definition Profile	Application Defaults
Generate a control string	Compile a map into a control string
Multi-Occurrence Structure Mapping	Path Qualified Mapping
Network Operation Profile	Network Commands
Requestor Profiles	Mailboxes
Security Profile	Network Security
System Profile	CICS Performance
Trading Partner Transactions (TPT)	Maps
Translation Tables	Forward and Reverse Translation Tables
Validation Tables	Code Lists

Host functions not available in the DataInterchange Client interface

The following functions must be accessed through the DataInterchange Host interface:

- DataInterchange utility
- Translation and communication options of the transaction store

The DataInterchange Client interface

DataInterchange Client is designed to make working with DataInterchange easier. Drawing on the power and usability of the Microsoft Windows graphical environment, DataInterchange Client makes DataInterchange setup, maintenance, and management easier.

Through the DataInterchange Client interface you can create, update, and manage DataInterchange:

- Setup profiles
- Trading Partner profiles
- Maps
- Data formats
- EDI Standards

You can also import eXtensible Markup Language (XML) Document Type Definitions (DTDs) into DataInterchange Client and create and print reports.

Using windows

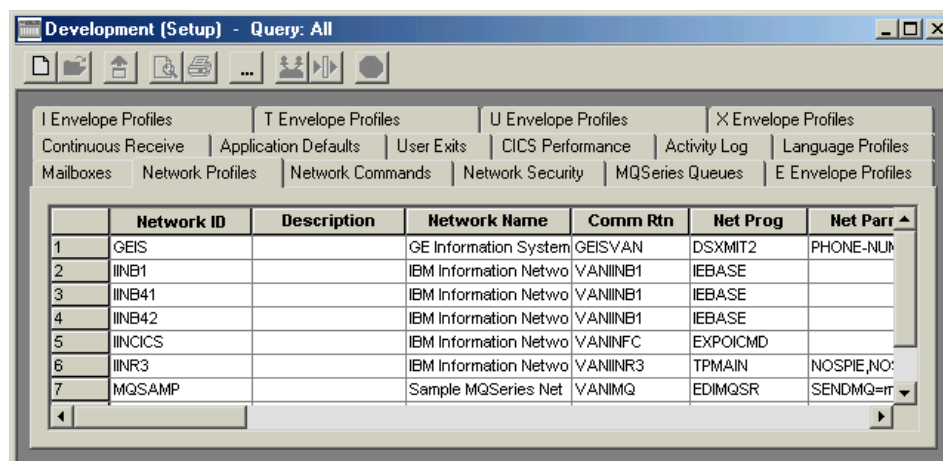
DataInterchange Client displays information in three types of windows:

- List
- Tree
- Editor

This section describes those windows and explains how to control them.

List windows

The purpose of a list window is to display a list of items of a given type, such as a list of trading partners. List windows allow you to choose items on which you want to perform such actions as editing, printing, deleting, and renaming.



	Network ID	Description	Network Name	Comm Rtn	Net Prog	Net Parr
1	GEIS		GE Information System	GEISVAN	DSXMIT2	PHONE-NU
2	IINB1		IBM Information Netwo	VANINB1	IEBASE	
3	IINB41		IBM Information Netwo	VANINB1	IEBASE	
4	IINB42		IBM Information Netwo	VANINB1	IEBASE	
5	IINCICS		IBM Information Netwo	VANINFC	EXPOICMD	
6	IINR3		IBM Information Netwo	VANINR3	TPMAIN	NOSP,NO
7	MGSAMP		Sample MQSeries Net	VANIMQ	EDIMQSR	SENDMQ=rr

The rows you see displayed in a list window tab, as illustrated above, are the result of a query against the database. The column names represent information stored in the database, which you select when you define your list window queries. To change that information, you use an editor window, as described on page 27. The list window also contains the date, time, and user ID of the last update.

Modifying list window information

You can control the information that is displayed in list windows, as well as the way the information displays.

◆ To display additional columns in a list window:

Click the scroll bar on the bottom of the screen to scroll to the right or left.

◆ To select the columns that display on the screen:

1. Click Modify Window Properties (...). Refer to “List Window Tool Bar Buttons” on page 32.

The Object List Window Query dialog box displays.

2. If you want to change the current query, select a query from the Current Query drop-down list.

Columns that are available in that query display in the Selected Columns and/or Available Columns list box. You can select the columns you want to display in the window by completing this procedure.



NOTE: You need not create a new query to change the columns that display in the list window.

3. Select the columns you want to display in the list window from the Available Columns list box.

Columns in the Selected Columns list box already display in the window. Use the < button to move a column to the Available Columns list box so that it does not display.

The > button moves columns back to the Selected Columns box. The << button and >> button move all the columns in one list box to the other.



NOTE: The columns in the Selected Columns list box are the columns that have been set up in the Current Query. To modify the query or create a new one, see Chapter 24, “Queries,” on page 285.

4. When you have finished selecting the columns you want to display in the list window, click OK.

The modified query is executed and the list window displays again with your new column selections.

If you are working in a list window opened through the Navigator bar, your column selection is saved as that window’s default. The next time you open that list window, you will see that column setup with the data displayed as you left it.

You can also change the width of each column in a list window.

◆ To change the width of a column:

1. Move the cursor pointer over the line that divides columns until the cursor changes to a double arrow.
2. Click and hold down the left mouse button.
3. Drag the arrow to either increase or decrease the width of the column and release the mouse button.

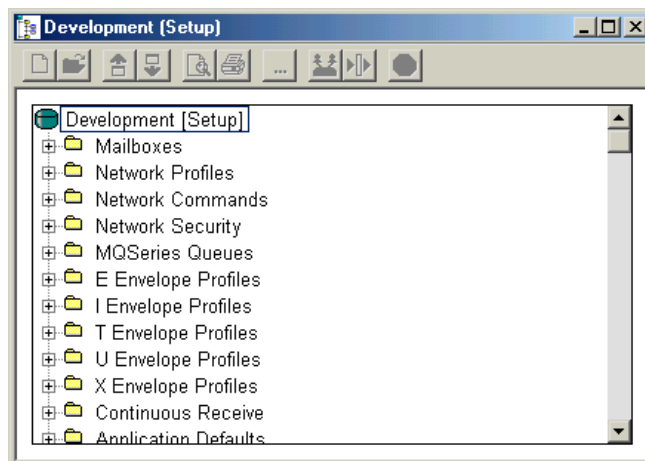
Tree windows

DataInterchange Client also allows you to view lists in Tree View, rather than in the tabbed list window.

◆ To use tree view:

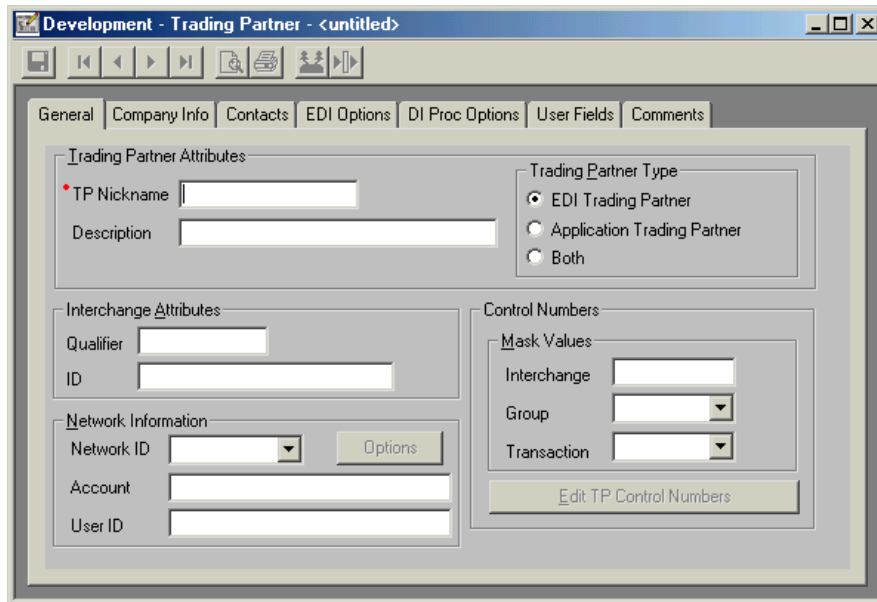
1. Click the plus (+) sign to expand the tree.
2. The queries associated with that branch of the tree display.
3. Double-click a query to run it and display items in the list window.
4. Double-click an item to open its editor window.

To set the default display to Tree view, see “Selecting list window options” on page 42.



Editor windows

Editor windows display when you open an item in a list window. They feature tabs that contain fields, drop-down lists, check boxes, and other Windows controls, as illustrated:



Many of the fields and control names display as column names across the top of list windows. Enter information into editor windows using the keyboard, mouse, and Windows controls.



NOTE: When creating a new item, required fields in editor windows are preceded by a red dot.

List and editor windows work like any other standard window in Microsoft Windows. You can minimize and maximize windows and move them around within the application. For more information on using Windows, see your Microsoft Windows documentation.

When you are finished with a window, close it as follows.

◆ To close a window, do any of the following:

- Click the X in the upper right corner, or press Control plus the F4 key.
- Click the Control Menu in the upper left corner of the window and select Close.
- Select Close from the File menu.

For more information on the Control Menu, see your Windows documentation.

Selecting commands

DataInterchange Client offers three different methods of selecting commands:

- **Menus**
You can perform an action by selecting its command from a menu using the mouse or keyboard.
- **Tool bars**
You can perform an action by clicking its button using the mouse.
- **Shortcuts**
You can perform an action using keyboard or mouse shortcuts.

The following sections describe those methods.

Menus

Menu options offer the most comprehensive method of performing common DataInterchange functions. Using the menu options you can: open new and existing items; create a query or a report; import and export components; save, close, preview, and print items; navigate through lists of items; set viewing preferences; access help; and exit DataInterchange Client.

Most of the options in the menus are self-explanatory. The options that display under the menus (File, Actions, Edit, Navigate, View, Window, and Help) vary depending on what function you are using. Whether or not options are available (that is, whether they appear black instead of gray) often depends on, for example, which tab is in front.

Table 2, “Availability of Menu Options,” on page 28, illustrates when the various menu options are available. Table 3, “Activating Menu Options,” on page 29, illustrates what action you need to perform in order to activate various menu options.

Table 2. Availability of Menu Options

Options on these menus are available when.No window is open	. . . A list window is active	. . . An editor window is active
File Menu	Open Browser Open Import File Open Query List Open Report List Exit	New Open Close Open Browser Open Import File Open Query List Open Report List Print Print Preview Print List Properties Exit	Save Close Open Browser Open Import File Open Query List Open Report List Print Print Preview Exit

Table 2. Availability of Menu Options

Options on these menus are available when.No window is open	. . . A list window is active	. . . An editor window is active
Actions Menu		Copy Rename Delete Export to File Export to Other System Usages Compile (Generate) Create Standard from Data Format Unlock	Export to File Usages Compile (Generate) Create Standard from Data Format
Edit Menu		Create Standard from Data Format	Cut Copy Paste
Navigate Menu			Move First Move Previous Move Next Move Last
View Menu	Status Bar Navigator Bar Preferences Message Log Event Log Systems Customize DI Client Release Migration	Stop Loading Refresh Tool bar Status Bar Navigator Bar Preferences Message Log Event Log Systems Customize	Tool bar Status Bar Navigator Bar Preferences Message Log Event Log Systems Customize
Window Menu		Cascade Tile (List of open windows)	Cascade Tile (List of open windows)
Help Menu	Contents Search for Help on How to Use Help DataInterchange on the Web About DataInterchange	Contents Search for Help on How to Use Help DataInterchange on the Web About DataInterchange	Contents Search for Help on How to Use Help DataInterchange on the Web About DataInterchange

Table 3. Activating Menu Options

To display these menu options. . .	Do this:
Actions Menu	Open any window.
Edit Menu	Open an editor window.
Navigate Menu	Open an editor window.

Table 3. Activating Menu Options

To display these menu options. . .	Do this:
Open option, File menu	Highlight an item or items in the list window.
Save option, File menu	Create a new document and fill in at least one field or open an existing document and make a change.
Print Preview option, File menu	Highlight an item or items in the list window or open an existing document.
Print option, File menu	Highlight an item or items in the list window or open an existing document.
Copy option, Actions menu	Highlight an item or items in the list window. Not all items can be renamed.
Rename option, Actions menu	Highlight an item or items in the list window. Not all items can be renamed.
Delete option, Actions menu	Highlight an item or items in the list window.
Export to File option, Actions menu	Highlight an item or items in the list window. (Note that not all items can be exported.)
Export To Other System option, Actions menu	Highlight an item or items in the list window. (Note that not all items can be exported.)
Usages option, Actions menu	Highlight a map, or maps, or a trading partner or trading partners in its list window, or open the editor window of an existing map or trading partner.
Compile option, Actions menu	Highlight a map or maps or a control string or control strings in their list window from the map editor or the envelope standards.
Create Standard from Data Format	Highlight a data format from a data format list window or open a data format editor.
Unlock option, Actions menu	Highlight an item or items in the list window.
Cut option, Edit menu	Highlight a piece of text.
Copy option, Edit menu	Highlight a piece of text.
Paste option, Edit menu	Cut or copy a piece of text.
Navigate menu options	Select several items from a list window and open an editor window.

Tool bars








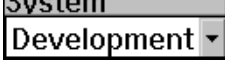

You can use tool bar buttons as shortcuts to perform most of the frequently used DataInterchange Client menu commands. The three DataInterchange Client tool bars are:

- Navigator bar
- List Window tool bar
- Editor Window tool bar

Navigator Bar

To display the Navigator bar at the top of the DataInterchange Client main screen, ensure it is checked in the View menu. Navigator bar buttons open DataInterchange Client's functional areas. Table 4 summarizes the Navigator bar buttons.










Table 4. Navigator bar Buttons

This button. . .	Does this:
	Opens the Setup List window.
	Opens the Trading Partner List window.
	Opens the EDI Standards List window.
	Opens the Data Formats List window.
	Opens the XML List window.
	Opens the Mapping List window.
	Opens the Transaction Store List window.
	Selects the system.
	Opens Help.

List window tool bar

Each work area in DataInterchange Client contains lists of items, which display in list windows. Each list window contains a tool bar. Table 5 illustrates the buttons that display on DataInterchange Client's list window tool bar.










Table 5. List Window Tool Bar Buttons

This button. . .	Does this:
 New	Creates a new item.
 Open	Opens the selected item from the list window.
 Export	Exports the selected item to a file.
 Preview	Previews the print format for the selected item.
 Print	Prints the selected item.
 Properties	Modify Window Properties. Selects or modifies a query that is run to load the list window.
 Usages	Displays which maps are associated with a trading partner or which trading partners are associated with a map. Use this button to create a usage or map rule, as well as to display usages or map rules.
 Compile	Compiles the selected map or envelope standard into a control string or recompiles the selected control string.
 Stop Loading	Stops building the list of items in a list window. This allows you to stop DataInterchange from building long lists when you see the item you are looking for.

Editor window tool bar

Each list window contains a list of components that you create and maintain using editor windows. Each editor window contains a tool bar. Table 6 illustrates the buttons that display on DataInterchange Client's editor window tool bar.

Table 6. Editor Window Tool Bar Buttons

This button. . .	Does this:
 Save	Saves the component. Enabled after any data in the window has been changed.
 Move First	Displays the first component you selected in the editor window, if you selected more than one.
 Move Previous	Displays the previous component you selected in the editor window, if you selected more than one.
 Move Next	Displays the next component in the editor window, if you selected more than one.
 Move Last	Displays the last component you selected in the editor window, if you selected more than one.
 Preview	Previews the print format for the selected component.
 Print	Prints the selected component.
 Usages	Displays which usages and map rules are associated with a trading partner or which usages or map rules are associated with a map.
 Compile	Compiles the selected map into a control string or recompiles the selected control string. Also, compiles the selected envelope standard into an envelope standard control string or recompiles the selected control string.

Shortcuts

DataInterchange Client allows you to select menu commands using standard Windows keyboard or mouse shortcuts. You can also use the keyboard to select items in list and editor windows, as well as navigate through fields in editor windows and grids.

◆ To select a menu command using the keyboard:

1. Press the Alt key.

The cursor moves to the menu bar.

2. Press the key corresponding to the underlined letter, usually the first, of the menu you want to display.

The menu displays.

3. Press the key corresponding to the underlined letter, usually the first, of the command you want to execute.

The command executes.



NOTE: You can also select commands by pressing the F10 key rather than the Alt key. Use the left and right arrow keys to move from menu to menu. Use the up and down arrow keys to scroll through the options on each menu.

The keyboard makes it easy to navigate through items and components in list windows and fields in editor windows, as summarized in Table 7.

Table 7. Keyboard and Mouse Shortcuts

Use this shortcut. . .	To:
Control plus N	New
Control plus O	Open
Control plus P	Print
Alternate plus F4	Close
Tab	Jump from one field to the next in the editor windows and grid editors.
Shift plus Tab	Move back a field in editor windows.
Control plus Tab	Flip from window to window.
Up and down arrows	Choose the item above or below the selected items in list windows.
Enter key	Scroll down the grid editor.
Shift plus down arrow or mouse drag	Select a range of sequential items in a list window or items in an editor window.
Control plus a mouse click	Select discontinuous items in a list window or items in an editor window.
Control plus Enter	Deselect a selected item.

Table 7. Keyboard and Mouse Shortcuts

Use this shortcut. . .	To:
Double-click the left mouse button	Open a window.



NOTE: Because double-clicking is the fastest way to execute a command, this manual uses it as the preferred method in procedures. You can also use the shortcut keys.

Performing common file management tasks

DataInterchange Client is designed to provide consistent actions across the user interface. Both list windows and editor windows follow consistent procedures for performing the common file management tasks of viewing, copying, editing, renaming, deleting, and printing items and components.

Procedures for importing and exporting items and components are also standard across DataInterchange Client windows. For details, see Chapter 4, “Export/Import.”

Viewing an item

◆ To view an item:

1. Click the button on the DataInterchange Client Navigator bar corresponding to the functional area you require.

The list window displays.

2. Click the tab within the list window corresponding to the item you wish to view.

A list displaying those items for the selected tab displays.

3. Double-click the item you wish to view.

The item displays in its editor window.

Copying an item

The copy function allows you to duplicate an item within the DataInterchange system in which you are working. If you want to base a new component on an existing component, for example, copy the existing component under a new name and edit it to the new specifications.

◆ To copy an item:

1. Click the button on the DataInterchange Client Navigator bar corresponding to the functional area you require.

The list window displays.

2. Click the tab within the list window corresponding to the item you wish to view.

A list displaying the items for the selected tab displays.

3. Click the name once of the item you want to copy.

4. Select Copy from the Actions menu.

The Copy Object dialog box displays with one or more fields used to name the component.

5. Type in one or more new values, and click OK.

DataInterchange Client copies the item.

Editing an item

If you want to update data in an item, open the editor window for that item, make your changes, and save them.

◆ To edit an item:

1. Click the button on the DataInterchange Client Navigator bar corresponding to the functional area you require.

The list window displays.

2. Click the tab within the list window corresponding to the item you wish to view.

A list displaying the items for the selected tab displays.

3. Double-click the item you want to edit.

The item's editor window displays.

4. Change values for fields as required.

5. Click Save on the tool bar to save the changes.

Renaming an item

◆ To rename an item:

1. In the appropriate list window, select the item you wish to rename.

2. Select Rename from the Actions menu.

The Rename Object dialog box displays.

3. Type in a new name and click OK.

DataInterchange Client renames your item.



NOTE: Your database system must support Cascade Update for rename to be fully functional. If Cascade Update is not supported by your database system, Rename will only work on simple objects. DB2 does not support Cascade Update.

Deleting an item

◆ To delete an item:

1. In the appropriate list window, select the item you wish to delete.
2. Select Delete from the Actions menu.

A confirmation displays.

3. Click Yes if you want to delete the item.

DataInterchange Client displays a message in the Execution Status window when it has completed the deletion. Click Close to close the Execution Status window.

Printing an item

◆ To print an item:

1. In the appropriate list window, select the component you wish to print.
2. If you want to preview the item, click Print Preview on the tool bar.

DataInterchange Client shows you a preview of the printed document.

3. Click Print on the tool bar.

A Print dialog box that allows you to select printers and other print options displays.

4. Click OK.

DataInterchange Client sends the item to the default or selected printer.



NOTE: You can also double-click an item in its list window to open the editor window, and print from there.

Using editor window grids

Some DataInterchange Client editor windows use a grid system for editing. In addition to entering information in rows, as in a spreadsheet, you also use standard Windows controls, such as drop-down lists and check boxes. Table 8 summarizes the procedures for using the grid editors.

Table 8. Grid Editor Procedures

To . .	Do this:
Enter a new row	Type in the row designated by an *.
Insert a new row	<p>Select the row (by clicking the row number) which you would like to display below the row you want to insert, and click Insert.</p> <p>A new row designated by an * displays.</p> <p>If the selected row is below the current * row, the * row moves below the selected row.</p> <p>If the selected row is above the * row, the * row moves above the selected row.</p>
Edit information in a row	<p>Click the cell that you want to edit. If the cell you selected has a drop-down list, an arrow displays on the right. Click the arrow to view the drop-down list, and then select your entry from the list.</p> <p>If there is no drop-down list, you see a cursor when you click the cell. Type in information.</p> <p>On some drop-down lists, you can get a cursor by double-clicking the cell.</p> <p>If a cell contains numeric information, up and down arrow keys display when you double-click the cell. Use these keys to scroll to the number you want to enter in that cell.</p>
Edit a component	<p>Click the row number to select the entire row, and then click Edit or double-click the row number.</p> <p>The appropriate Editor displays.</p>
Delete a row	Click the row number to select the entire row, and then click Delete.



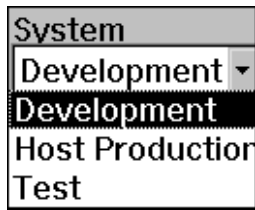
NOTE: You cannot edit all columns in a grid. Columns you cannot edit have a gray background.

Working with multiple systems

DataInterchange allows you to work in as many systems as you like at the same time. This can be very convenient for comparing information in the test and production systems, for example. This section shows you how to select systems and manage their windows.

Selecting a System

If you open list windows by clicking their respective buttons on the tool bar, they automatically open in the default system. DataInterchange Client displays the name of your current default system in the System box on the Navigator bar.

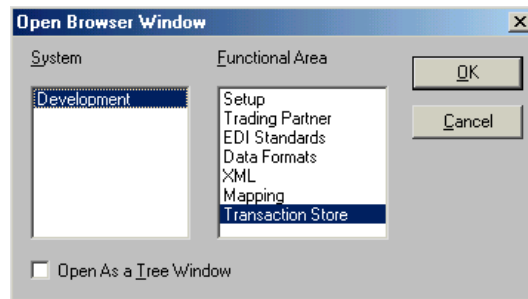


You can select the system by choosing a system from this drop-down list, or by opening items from the Open Browser option in the File menu.

◆ To select a system using the Open Browser option

1. Select Open Browser from the File menu.

The Open Browser window displays.



2. In the System list box on the left, click the system in which you wish to work.
3. In the Functional Area list box on the right, select the area in which you wish to work.



NOTE: Selecting an area in the Functional Area list box is the same as clicking a button on the Navigator tool bar.

4. If you wish to view your work in a tree window, rather than in list windows, click the Display as a Tree View check box.

5. Click OK.

DataInterchange Client opens the list window of the Functional Area you selected in the System you selected. If you select a system that you have not worked in since you started DataInterchange Client, you may be required to log on.

Understanding window title bars

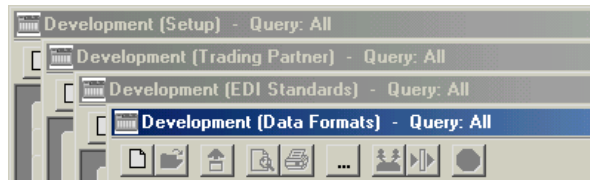
The window title bar displays important information for keeping track of which system you are working in. The name of the system is on the title bar of each DataInterchange Client window. The name of the functional area you are working with displays to the right of the system name in parentheses. The current query displays to the right of the functional area in list windows, the current open item displays to the right of the functional area in editor windows.

Note that you can have multiple windows from multiple systems open at the same time.

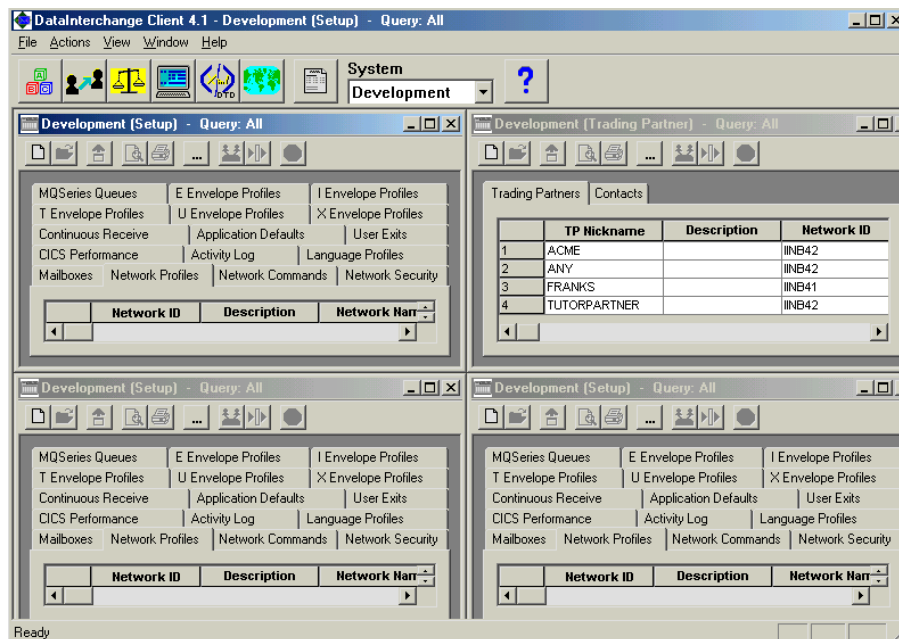


TIP: To avoid confusing which system you are using, set the window background of each system to a different color using the Preferences command on the View menu, as described on page 42.

To cascade windows as shown below, from the Window menu, select Cascade.



To tile windows as shown below, from the Window menu, select Tile.



Setting preferences

DataInterchange Client allows you to set preferences for what you view on the screen and how it is viewed. You can also set preferences for the Message Log.

Setting window preferences

When you start DataInterchange Client, it maximizes the main window to full screen and restores any list windows that you left open upon exit. Change those defaults through the Preferences dialog box.

◆ To set the Main window display option:

1. Select Preferences from the View menu.

The Preferences dialog box displays.

2. Click the Maximize Main Window check box.

Inserting a check sets the default window size so that it displays at maximum size when you start DataInterchange Client.

Removing a check sets the default so that the Main window displays on the screen at the same size you set it before you last exited DataInterchange Client.

3. Click OK.

◆ To set the list window default:

1. Select Preferences from the View menu.

The Preferences dialog box displays.

2. Click the Restore List Windows check box.

Inserting a check sets the default so that list windows left open upon exit display when you restart DataInterchange Client.

Removing a check sets the default so that list windows left open upon exit are closed when you restart DataInterchange Client.

3. Click OK.

Setting message log preferences

DataInterchange Client maintains a Message Log that captures information on errors and problems. You may disable the Message Log and change the length of time that it stores messages.

◆ To disable the Message Log:

1. Select Preferences from the View menu.

The Preferences dialog box displays.

2. Click the Disable Message Logging check box to insert a check.

3. Click OK.

The next time you start DataInterchange Client, the Message Log will not collect data.

◆ To set the length of time the Message Log stores messages:

1. Select Preferences from the View menu.

The Preferences dialog box displays.

2. Check the Delete Messages Older than X Days check box, and type the number of days you wish to store messages.

If you do not want to delete messages from the log, click the check box to remove the check.

3. Click OK.

The Message Log will purge messages older than the number of days you entered when you restart DataInterchange.

Selecting the system color**◆ To set window color preferences:**

1. Select the system whose window background color you wish to change through the Systems list.

2. Select Preferences from the View menu.

The Preferences dialog box displays.

3. Click Change in the Color Options list box.

The standard Windows Colors dialog box displays.

4. Click the color in which you wish the window background to display.

5. Click OK to close the Colors dialog box.

6. Click OK to close the Preferences dialog box.

The background of all windows in the selected system displays in the color you selected.



NOTE: Colors can also be changed by editing the system (View -> Systems).

Selecting list window options

DataInterchange Client displays information in list windows. You can view information using any of the following options:

- Multiline tabs

When you open list windows either by clicking on Navigator bar buttons or using the Open Browser command on the File menu, information displays in a tabbed list window. If there are too many tabs to fit into the list window, the list window displays with multiple lines of tabs.

- **Single line tabs**
When you open list windows either by clicking on Navigator bar buttons or using the Open Browser command on the File menu, information displays in a tabbed list window. If there are too many tabs to fit into the list window, scroll buttons appear on the right side of the tabs. Use these buttons to scroll tabs into view.
- **Tree view**
When you open list windows either by clicking on Navigator bar buttons or using the Open Browser command on the File menu, the list window displays information in a tree view.

◆ **To select the list window view:**

1. Select Preferences from the View menu.
The Preferences dialog box displays.
2. Click the appropriate radio button for your view choice.
3. Click OK.

Customizing field tags

DataInterchange Client allows for the customization of User Field 1 through User Field 10 tags on the Trading Partner User Fields tab page.

◆ **To customize the labels:**

1. Select Customize from the View menu.
The Customize editor displays.
2. Double-click the list entry you want to rename.
An editor page displays.
3. In the Displayed Field Label edit box, type the new name of the User Field.
4. Click Save, and close the window.
5. Repeat the process for the other tags you want to change.

Viewing control and status bars

DataInterchange Client allows you to control which tool bars to view on the screen, if any. You can also choose whether to view the Status bar, which runs along the bottom of the screen and provides system information.



The Navigator bar, tool bar, and Status bar all display on the screen by default. You may hide any or all bars.

◆ **To hide a control or status bar:**

On the View menu, select the bar you wish to hide by clicking it to remove the check mark. That bar no longer displays on the screen. Bars preceded by a check mark on the menu display on the screen.

Getting help

DataInterchange Client includes online, context-sensitive Help that allows you to display information about virtually any aspect of the program. In most cases, getting the help you need is as easy as clicking a Help button or pressing the F1 key.

The Help system contains information about each DataInterchange Client screen and field and explains how to use the operations presented to you. You also have access to the DataInterchange Client glossary of terms, error messages, and DataInterchange reference material. Each option on the Help menu is described briefly below.

Contents

Select this option to display the Contents screen for DataInterchange Client Help. This screen allows you to access all of the major areas of Help by clicking the name of an area.

Search for help on

Select this option to display the Search dialog box. Type a word in the field to display a list of index entries beginning with that word. Select an index entry, and then click Display to view the information related to the index entry you selected.

How to use help

Select this option to display the Windows help topics that describe Windows Help.

DataInterchange on the Web

Select this option to display a submenu with links to different pages at the DataInterchange Web site, including:

- DataInterchange Home Page
- Fact Sheet
- Downloads
- Frequently Asked Questions
- Technical Support

About DataInterchange Client

Select this option to display a dialog box that contains the DataInterchange Client version number, as well as the percentage of system resources and memory currently available.

Accessing online help

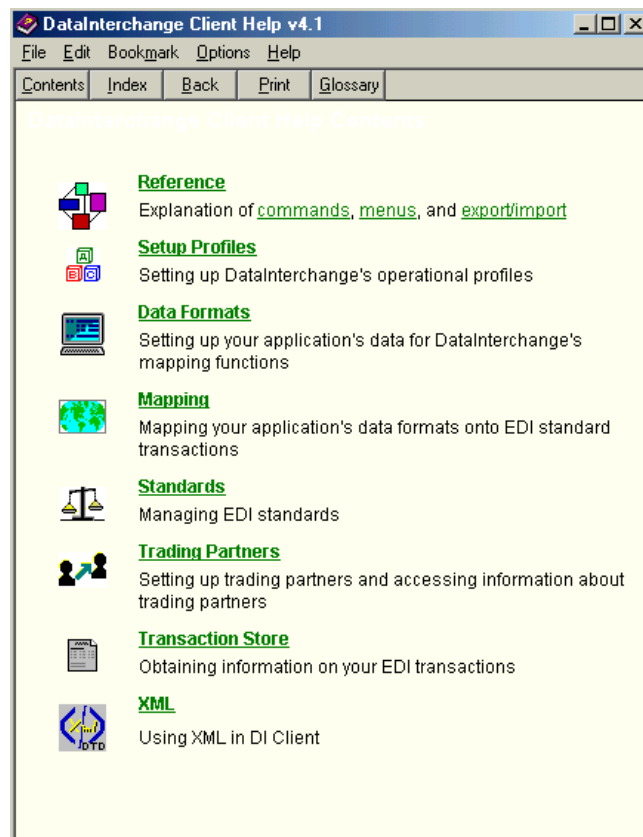
There are three ways to access DataInterchange Client Help:

- **Help Buttons**
Some DataInterchange Client windows and dialog boxes include a Help button. Clicking Help displays a Help window that contains information specific to the current window or dialog box. When you are through reading the information, double-click Control Menu in the upper-left corner of the window to exit Help and continue working.
- **Help Menu**
The DataInterchange Client Help menu provides direct access to the Help Contents, as well as Help topics for major tasks like creating and editing documents. Select an option from the menu to display that portion of Help.
- **F1 Key**
DataInterchange Client's help is context sensitive, which means if you click a window, and then press the F1 key, DataInterchange Client displays the relevant help topic.

Some DataInterchange Client error boxes allow you to receive additional information by pressing the F1 key.

Using DataInterchange Client help

DataInterchange Client Help is designed to provide you with fast access to information when you run into a problem or when you just want to know more about a particular aspect of DataInterchange Client. The starting point of the Help system is the Help Contents screen (below), which shows you the type of information available.



From the DataInterchange Client Help Contents screen, you can get information about menu commands, see field definitions, display error messages, or view the glossary. You can also go directly to a number of specific Help topics on tasks, such as creating a document and communications. You can move freely through the system by clicking links.

Links

Links allow you to jump quickly from one Help topic to another simply by clicking a specially marked word or picture. Using hypertext links, you can browse through various Help topics or move from general information to more specific information.

In DataInterchange Client Help, as in many other Help systems, links display in green and are underlined with either a solid or a broken line. Click a link to display the corresponding Help topic.

Words or phrases underlined with a solid line take you to different topics. Keywords underlined with a broken line pop up a definition in the current topic.

Help screen buttons

The following five buttons appear at the top of the Help screen:

- Contents
This button returns to the Help Contents screen from any other Help screen.
- Search
This button displays the Help Search window, which contains a list of keywords and phrases. Scroll through the list to find a word or phrase you wish to select, or type a word to begin scrolling automatically. Click a keyword, and then click Show Topics, to display a list of related topics. Select a topic and click Go To to display the Help topic.
- Back
This button displays the previous Help topic. Click repeatedly to continue moving backward through previously accessed Help screens.
- Print
This button allows you to print the open Help screen. Click to display a print dialog box that allows you to select printers and other print options.
- Glossary
This button displays the DataInterchange Client glossary.

DataInterchange Client Help includes all the features available through Microsoft Windows Help. See Windows documentation for more information about Windows Help.

Using the Message Log

DataInterchange Client displays messages about errors that occur within DataInterchange Client, such as incorrect operations performed by the user or a failure to complete the current command. After displaying an error message, DataInterchange Client logs it in the Message Log, which you access from the View menu.

Viewing messages

You can view and print the Message Log for your information. Note, however, that the list cannot be edited, and it can only be deleted with the Message Log Purging option available in the application preferences section of DataInterchange Client, which is described on page 41.

◆ To view the Message Log:

1. Select Message Log from the View menu.

The Message Log displays. See Table 9 on page 47 for an explanation of the columns.

2. Double-click the item to view the entire message.

Table 9. Message Log Columns

This field. . .	Contains:
Updated Date/Time	The date and time the message occurred.
Updated User ID	The user who was on the system at the time the message occurred.
Message ID	A message identification number generated by DataInterchange Client.
Message Text	The message text. You only see a little bit of the text. To view the entire message, double-click the text.
Module	The portion of DataInterchange Client code in which the error occurred.
Line	The line number in the code in which the error occurred.

Setup for translating data

Before you can use DataInterchange to translate documents, messages, or files, certain information must be defined within DataInterchange. This information is used to describe how data is formatted in your source and target documents, how it is to be translated, how it is to be sent and received, and other pertinent information. The following outlines the necessary setup steps:

1. Install and establish your environment, including both the DataInterchange Server and the Client.

To install DataInterchange in MVS or CICS, see the *DataInterchange Installation Guide*. To install and set up the Client, see Chapter 1, “Installation and setup.” Also refer to “Configuration alternatives” on page 9 for client/server setup information.

2. Import or create document definitions.

- a. Add EDI standards and envelope standards.

Select the EDI standards you wish to use and import them to your DataInterchange system. To install EDI standards in the Client, see “Installing EDI standards” on page 16.

- b. Define data formats to DataInterchange.

Data formats are used to define the format of your application data to DataInterchange. Several record formats are supported by DataInterchange. One data format can be used to describe input application data that is sent to a trading partner and output application data that is received from a trading partner. See Chapter 18, “Data formats,” for more information.

- c. Import XML DTDs.

See “Extensible Markup Language (XML)” on page 183 for more information.

3. Verify/create a network profile.

See “Network profiles” on page 107 for more information.

4. Define yourself to DataInterchange.

See “Mailbox profiles” on page 99 for more information.

5. Define your trading partners.

A trading partner profile must be created for each trading partner with whom you will do business, unless you are using the minimal trading partner feature of DataInterchange. See “Specifying usages and rules” on page 136. The profile contains information used to identify the trading partner as well as other trading partner specific data. For more information, see Chapter 17, “Trading Partners.”

6. Customize the envelope profile, if needed.

You may need to create envelope profiles to provide customized envelope information. See Chapter 9, “Envelope profiles.”

7. Map your source document definition to your target document definition.

A document definition you defined to DataInterchange now must be mapped to a target document definition. This means associating the elements in your source document definition to the elements in your target document definition. You must create a map for translating your source document definition to your target document definition. See Chapter 21, “Creating a map,” for more information about creating a map.

As part of mapping, you may choose to specify translation tables as needed. Translation tables allow you to substitute one value for another when performing translations.

8. Identify which trading partners will use the maps.

Map rules and send and receive usages associate a map with the trading partner. Create map rules, send usages, or receive usages, depending on the type of map. Map rules and send and receive usages also define information used when performing translations, such as encryption keys, functional acknowledgment information, the type of envelope to use and the name of the envelope profile. Maps may be used by multiple trading partners. For more information, see Chapter 17, “Trading Partners.”

9. Generate a control string.

After you complete a map and associate it with trading partners using map rules, send usages, or receive usages, you must compile a control string before DataInterchange can use the map. The DataInterchange Host uses the control string in its translation processing.

Export/Import

With the export and import functions, you can install EDI standards on DataInterchange Client and transfer your work between computers and DataInterchange systems. In stand alone mode, you use the export and import functions to move data between DataInterchange Host and DataInterchange Client.

You can export and import files to exchange profiles with trading partners that use DataInterchange or with other users that are not connected to your local-area network. For example, a manufacturer can export the mapping of a purchase order to a supplier so that the supplier does not have to recreate the purchase order information.

About export/import

Export takes a DataInterchange item you select from either the DataInterchange Client or Host and writes it either to a file or to another DataInterchange system.

Import reads export files and inserts the data into a DataInterchange system. Export/import files on the PC are identified by the .EIF file extension.

You use the import and export functions when you need to:

- Install EDI standards on DataInterchange Client
- Install DTDs for XML data
- Move profiles, EDI standards, data formats, and maps from DataInterchange Host to DataInterchange Client
- Update DataInterchange Host databases with work done on DataInterchange Client when using the client in stand alone mode
- Share profiles, data formats, maps, and EDI standards with other DataInterchange users



NOTE: Export/import is a critical part of all DataInterchange Client interactions with DataInterchange Host when using the Client in stand alone mode. For information on how export/import works in moving data between DataInterchange Host and DataInterchange Client, see “Moving to DataInterchange Client” on page 17.

Export/import file specifications

The following specifications apply to export/import files:

- If you are moving Host format maps, EDI standards, and data formats from a previous release, refer to the *Migration Considerations* publication included with the product.
- The field position and length of fixed-format records are subject to change due to design changes within DataInterchange between different releases.
- Exporting EDI standards in fixed format is not recommended because of the amount of disk space they can take up. Tagged format is recommended for exporting EDI standards.
- Export only one complete EDI standard per export/import file to keep file sizes and export times manageable.

Exporting

The DataInterchange Client export function has two options:

- You can export to a file
- You can export to another DataInterchange system, for example, from the Test system to the Production system

Export to a file when you want to move an item.

- To DataInterchange Host from DataInterchange Client
- To another DataInterchange Client user
- From one system to another

Export to another DataInterchange system when you want to duplicate items in various DataInterchange systems. For example, if you created a trading partner in the Test system and want to copy to the Production system, use the Export to Other System command.

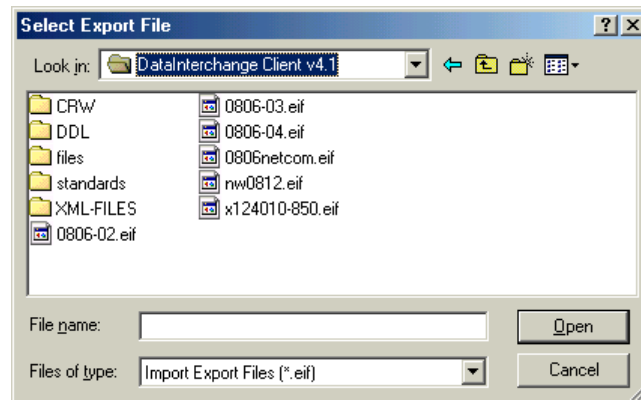
Exporting to an export/import file

When you complete work in stand-alone mode: you export to file; you use an upload utility to upload that file to the host; and then you use the host's import functions to merge that profile into the DataInterchange Host database, so that your changes can take effect.

◆ To export to a file:

1. To export an item to file, click it in its list window.
2. Click Export Selected Documents on the tool bar.

The Select Export File dialog box displays with the File Name field highlighted.



3. Either select or type the name of the file you want to export into. If you type the name, use the .EIF extension. You can also select an existing file you want to export to by selecting the drive and folder where the file is located.

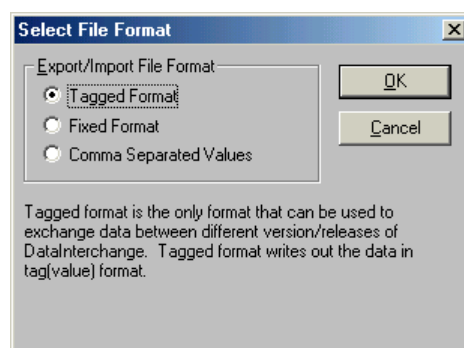
To see a list of all files in a folder, select “All Files (*.*)” from the drop-down list in the List of Files field.



TIP: You may export as many items as you like to a single file. For example, you can set up a file called TPUPDATE.EIF and export all of your updated Trading Partner profiles to that file.

4. Click Open.

If you typed in the name of a new file, the Select File Format dialog box displays.



NOTE: A DTD will be in .eif format when exporting.

a. Choose one of the following file record formats:

- *Tagged Format*
Writes data into a file with tags representing database fields. This format is recommended because empty fields are not referenced in the file; therefore, it creates the smallest files. This format allows you to move data between versions of DataInterchange Client.
- *Fixed Format*
Writes data into a file where each field is represented at its full length in the database. If data in the field does not fill up its whole length, blanks are appended to the data. Use this format when it is the only format your application will accept.
- *Comma-Separated Values Format*
Writes data into a file in which each field is separated by a comma. Character fields appear in quotes; numeric fields do not. Use this format to load data into spreadsheet and other PC applications.

If you type in or select a file that has had data exported to it before, DataInterchange Client exports your data in the format selected when the file was first created. Data is appended to the end of the file.

5. Click OK.

- If you are exporting any of the profiles listed in Table 10, “Items with Associated or Referenced Types” on page 55, DataInterchange Client displays a dialog box asking you which referenced and associated types you want to export with the item you are exporting. Select the types by clicking the check boxes. For explanations of each of these dialog boxes, see “Specifying referenced and associated types” on page 58.

6. When you have finished filling out the dialog box, click OK.

- If you are exporting any of the profiles listed in Table 11, “Items with no Associated or Referenced Types” on page 55, no dialog box displays before export. DataInterchange Client exports the item to the specified file and displays an Execution Status window. When DataInterchange Client is finished exporting the item, an Export Complete message displays in the Execution Status window.

Exporting to other systems

The Export to Other System command is useful for exporting items that have been developed and tested in the Test system to the Production system, where they can be used to exchange data with trading partners.

◆ To export to other systems:

1. To export an item to another system, select the item and choose Export to Other System from the Actions menu.

The Select a System dialog box displays with the available systems listed.

2. Select the system to which you want to export your item and click OK.

A dialog box for your system’s client-server middleware package may be displayed.

3. If the dialog box displays, enter your user ID and Password to gain access to the system you have selected and click OK.

If you are exporting any of the profiles listed in “Items with Associated or Referenced Types” on page 55, DataInterchange Client displays a dialog box asking you which referenced and associated types you want to send with the item you are exporting. Select the types by clicking the check boxes. For explanations of each of these dialog boxes, see “Specifying referenced and associated types” on page 58 of this chapter.

4. When you have finished filling out the dialog box, click OK.

If you are exporting any of the profiles listed in Table 11 on page 55, no dialog box displays before export.

DataInterchange Client exports the selected item.



NOTE: When you export an item to a system which already has an item of that type with the same name, DataInterchange Client displays a warning dialog box. If you want to replace the existing item with the one you are exporting, click OK. Otherwise, you can cancel your export.

Table 10. *Items with Associated or Referenced Types*

Item	Page
Mailbox	58
Network Profile	59
Application Defaults profile	59
Continuous Receive profile	60
Network Security profile	62
Trading Partner profile	62
Data Format Dictionary	64
Data Format	64
EDI Standard Dictionary	64
EDI Standard Transaction	65
Envelope Standard	65
Mapping	66
Control Strings	67

Table 11. *Items with no Associated or Referenced Types*

Item
Activity Log profile
CICS Performance profile

Table 11. Items with no Associated or Referenced Types

Item
Code List
Contacts
E, I, T, U, and X Envelope profile
Envelope Control String
Forward Translation Table
Global variables
Language profile
MQ Series Queues
Network Commands profile
Reverse Translation Table
User Exits profile
XML Dictionary
XML DTDs

Importing from an export/import file

The import function allows you to use an item received from another DataInterchange user.

◆ To import an item:

1. Select Open Import File from the File menu.

The Select Import File dialog box displays with the File Name field highlighted.

2. You can select an existing file from which you want to import by selecting the drive, folder, and file.

To see a list of all files in a folder, select “All Files (*.*)” from the drop-down list in the List of Files field.

3. Click Open.

The tree view of the export/import file you selected displays with folders below representing each type of item that you have exported to that file. To see the contents of a folder, click the plus sign to the left of the folder. The contents of a folder are shown as documents beneath that folder.

4. Double-click the document you want to import.
5. Click Import on the tool bar.

The Select a System dialog box displays with the available systems listed.

6. Select the system to which you want to import your item and click OK.

A dialog box for your system's client-server middleware package may display for you to log in to the system.

7. If a dialog box displays, enter your user ID and password to gain access to the system you have selected and click OK.

DataInterchange Client imports the selected items into the specified system and displays the Execution Status window so that you can monitor the progress of the import. When DataInterchange Client is finished importing the item, an Import Complete message displays in the Execution Status window.



NOTE: When you import an item to a system that already has an item of that type with the same name, DataInterchange Client displays a warning dialog box. If you want to replace the existing file with the one you are importing, click OK. Otherwise, you can cancel your import.

Importing a DTD file

A DTD file must be obtained and then imported into DataInterchange.

◆ To import a DTD:

1. Select Open Import File from the File menu.

The Select Import File dialog box displays with the File Name field highlighted.

2. Either select or type the name of the file from which you want to import. If you type the file name, you should type the .DTD file extension. You can also select an existing file from which you want to import by selecting the drive and folder where the file is located.

To see a list of all DTD files in a folder, select “XML DTD File (*.dtd)” from the drop-down list in the List of Files field.

3. Click Open.

The Import XML DTD window opens with the name of the DTD file shown in the DTD Name field.

4. Select a dictionary from the drop-down list in the Dictionary Name field. Selecting a dictionary is required.

5. You can enter the root element in the Root Element Name field.
6. You can enter a description of the DTD file in the Description field.
7. Click the Import button when you complete the field entries.

Specifying referenced and associated types

Many DataInterchange items do not work in isolation but in relation to other items. Maps, for example, work in relation to EDI standards and usages or map rules, which work in relation to Trading Partner profiles. When you export such items, you may need to export the items to which they are related. The Export/Import dialog boxes allow you to select which related types you want to export along with the primary item.

There are two types of relationships between DataInterchange items:

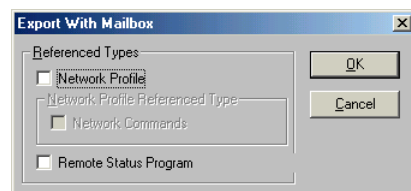
- **Referenced**
Referenced types are items that are referred to by the item you are exporting.
- **Associated**
Associated types, on the other hand, refer to the item you are exporting.



ATTENTION: When importing associated and referenced types, the imported items replace existing ones with the same name without warning.

Procedures for each item with associated or referenced types follow.

Mailbox



◆ To complete the Export Mailbox dialog box:

1. Click the Network Profile check box if you want to include the Network profile referenced in the Mailbox you are exporting.
2. Click the Network Commands check box if you want to include the Network Commands referenced in the Network profile you are exporting with the Mailbox profile.
3. Click the Remote Status Program Profile check box if you want to include the Remote Status User Exit item referenced in the Mailbox profile you are exporting.
4. When you have finished selecting the referenced types you wish to export, click OK.

DataInterchange Client begins to export the item.

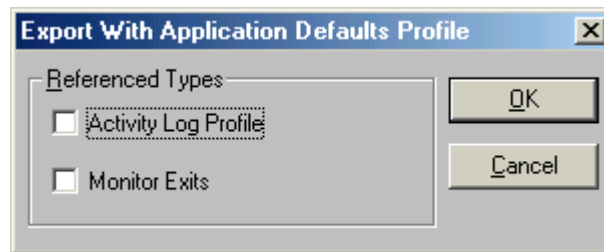
Network profile



◆ To complete the Export With Network Profile dialog box:

1. Click the Network Commands check box if you want to include the Network command referenced in the Network profile you are exporting.
2. When you have finished selecting the referenced types you wish to export, click OK.
DataInterchange Client begins to export the item.

Application defaults profile



◆ To complete the Export With Application Defaults Profile dialog box:

1. Click the Activity Log Profile check box if you want to include the Activity Log referenced in the Application Defaults profile you are exporting.
2. Click the Monitor Exits check box if you want to include the Monitor User Exit referenced in the monitor user exit you are exporting.
3. When you have finished selecting the referenced types you wish to export, click OK.
DataInterchange Client begins to export the item.

Continuous receive profile

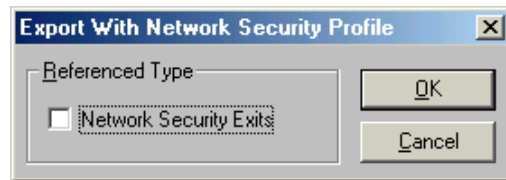
◆ To complete the Export with Continuous Receive Profile dialog box:

1. Click the Activity Log Profile check box if you want to include the Activity Log referenced in the Continuous Receive profile you are exporting.
2. Click the Mailbox Profile check box if you want to include the Mailbox profile referenced in the Continuous Receive profile you are exporting.
 - a. Click the Network Profile or Remote Status Program check boxes if you want to include Network profile or Remote Status Program user exit types referenced in the Mailbox profile you are exporting with the Continuous Receive profile.
 - b. Click the Network Commands check box if you want to include the Network Commands referenced by the Network profile you are exporting with the Continuous Receive profile.
3. Click the Trading Partner Profile check box if you want to include the Trading Partner referenced in the Continuous Receive profile you are exporting.
 - a. Click the Trading Partner Contacts, Network Profile or Network Security Profile check boxes if you want to include the appropriate items that are referenced in the Trading Partner profile you are exporting with the Continuous Receive profile.

- 1) Click the Network Commands check box to include the Network Commands referenced in the Network profile you are exporting with the Continuous Receive profile.
 - 2) Click the Network Security Exits check box if you want to include the Security User Exits referenced in the Network Security profile you are exporting with the Continuous Receive profile. Security User Exits include Authentication, Encryption, Filtering, and Compression.
 - b.** If you include the referenced Trading Partner Profile, you may choose to include its associated types, Receive Usages, Send Usages, or Rules.
 - 1) Click the option button corresponding to the usages or map rules you want to include.
 - To export all usages or rules, click All.
 - To export no usages or rules, click None.
 - To select which usages or rules you want to export, click Select.
- During the export, the appropriate usages or rules dialog box displays.
- 2) Select the usages or rules you want to include with the export and click OK.
- 4.** Click the Activity Log Profile check box if you want to include the Activity Log profile referenced in the Continuous Receive profile you are exporting.
 - 5.** Click the Mailbox Profile check box if you want to include the Mailbox profile referenced in the Continuous Receive profile you are exporting.
 - a.** Click the Network Profile or Remote Status Program check boxes if you want to include Network profile or Remote Status Program user exit types referenced in the Mailbox profile you are exporting with the Continuous Receive profile.
 - b.** Click the Network Commands check box if you want to include the Network Commands referenced by the Network profile you are exporting with the Continuous Receive profile.
 - 6.** When you have finished selecting the referenced and associated types you wish to export, click OK.

DataInterchange Client begins to export the item.

Network Security profile

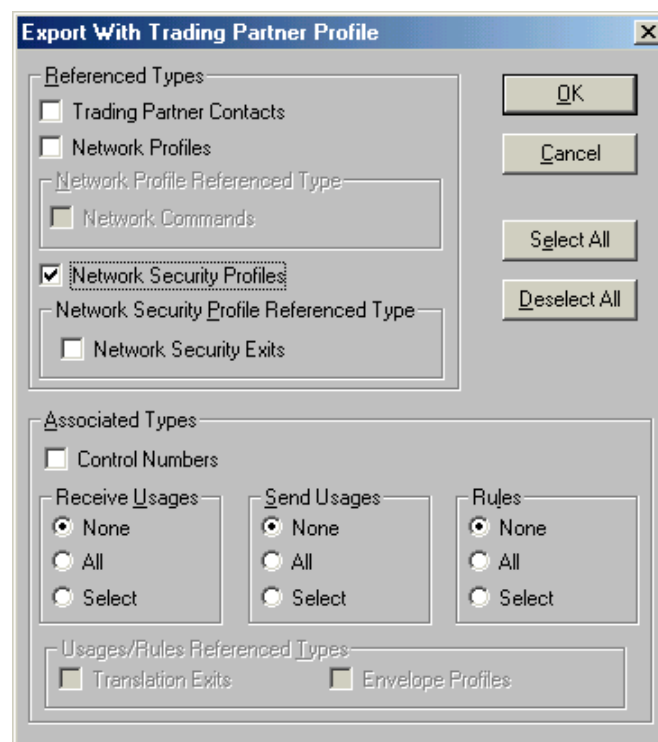


◆ To complete the Export with Network Security Profile dialog box:

1. Click the Network Security Exits check box if you want to include the Security user exits (Authentication, Encryption, Filtering, and Compression) referenced in the Network Security profile you are exporting.
2. When you have finished selecting the referenced type you wish to export, click OK.

DataInterchange Client begins to export the item.

Trading Partner profile



◆ To complete the export Trading Partner Profile dialog box:

1. Click the Trading Partner Contacts check box if you want to include the Trading Partner Contacts referenced in the Trading Partner profile you are exporting.

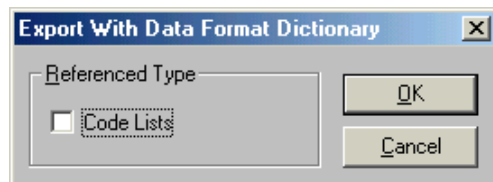
2. Click the Network Profile check box if you want to include the Network profile referenced in the Trading Partner Profile you are exporting.
 - a. Click the Network Commands check box if you want to include the Network Commands profile referenced in the Network Profile you are exporting with the Trading Partner profile.
3. Click the Network Security Profile check box if you want to include the Network Security profile referenced in the Trading Partner Profile you are exporting.
 - a. Click the Network Security Exits check box if you want to include the Network Security User Exits profile referenced in the Network Profile you are exporting with the Trading Partner profile.
4. Click the Control Numbers check box if you want to include the control number pairings.
5. You may choose to include associated types, Receive Usages, Send Usages, and Rules with the Trading Partner profile you are exporting.
 - a. Click the option button corresponding to the usages or rules you want to export.
 - To export all usages or rules, click All.
 - To export no usages or rules, click None.
 - To select which usages or rules you want to export, click Select.

During the export, the appropriate usages or rules dialog box displays.

 - b. Select the usages or rules you want to export and click OK.
6. You may choose to include Usages or Rules referenced types, Translation Exits or Envelope Profiles.
7. When you have finished selecting the referenced and associated types you wish to export, click OK.

DataInterchange Client begins to export the item.

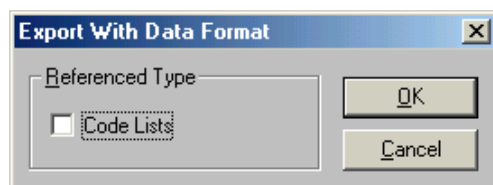
Data format dictionary



◆ To complete the **Export with Data Format Dictionary** dialog box:

1. Click the Code Lists check box if you want to include the Code List referenced in the Data Format Dictionary you are exporting.
2. When you have finished selecting the referenced type you wish to export, click OK.
DataInterchange Client begins to export the item.

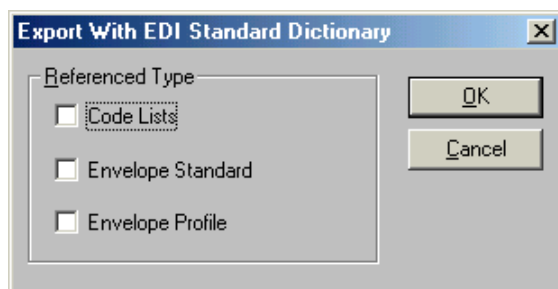
Data format



◆ To complete the **Export with Data Format** dialog box:

1. Click the Code Lists check box if you want to include the Code List referenced in the Data Format you are exporting.
2. When you have finished selecting the referenced type you wish to export, click OK.
DataInterchange Client begins to export the item.

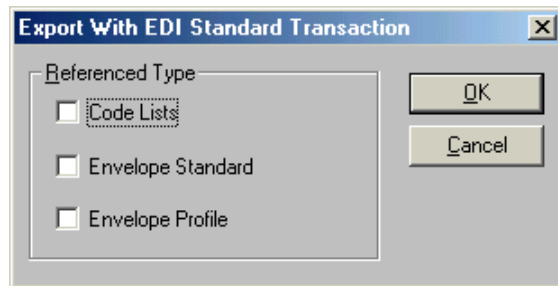
EDI Standard dictionary



◆ **To complete the Export with EDI Standard Dictionary dialog box:**

1. Click the appropriate check box to include that referenced item in the EDI Standard Dictionary you are exporting.
2. When you have finished selecting the referenced types you wish to export, click OK.
DataInterchange Client begins to export the item.

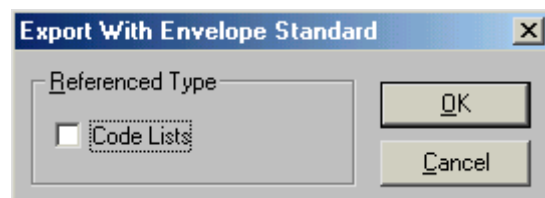
EDI Standard transaction



◆ **To complete the Export with EDI Standard Transaction dialog box:**

1. Click the appropriate check box to include that referenced item in the EDI Standard Transaction you are exporting.
2. When you have finished selecting the referenced types you wish to export, click OK.
DataInterchange Client begins to export the item.

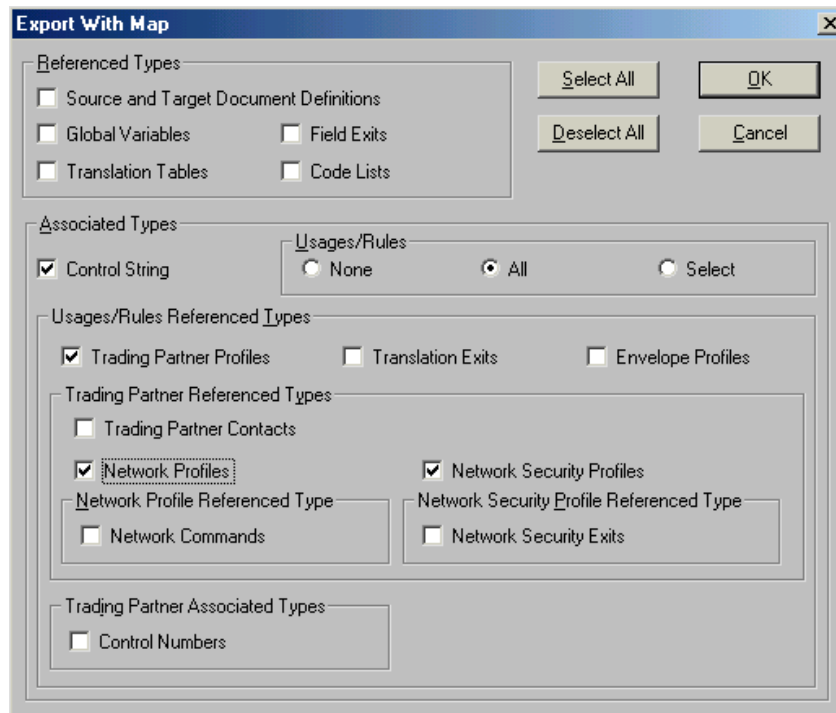
Envelope standard



◆ **To complete the Export with Envelope Standard dialog box:**

1. Click the appropriate check box to include that referenced item in the envelope standard you are exporting.
2. When you have finished selecting the referenced types you wish to export, click OK.
DataInterchange Client begins to export the item.

Mapping



◆ To complete the export Mapping dialog box:

1. Click the check boxes of the following referenced types if you want to include them with the map you are exporting: Source and Target Document Definitions, Global Variables, Translation Tables, Field Exits, and Code Lists.
2. You may choose to include the associated types, Control String and Usages/Rules, with the map you are exporting.
 - a. If you want to include the Control String associated with the map you are exporting, click the Control String check box.
 - b. If you want to include the usages or rules associated with the map you are exporting, click one of the three option:
 - 1) Click the option button corresponding to the usages or rules you want to export.
 - To export all usages or rules, click All.
 - To export no usages or rules, click None.
 - To select which usages or rules you want to export, click Select.

During the export, the appropriate export usages or rules dialog box displays.

- 2) Select the usages or rules you want to send with the export and click OK.

- c. If you include usages or rules associated with your map, you may choose to send Trading Partner Profiles, Translation Exit Routines, and Envelope Profiles referenced in the Usages/Rules by clicking the respective check boxes.
 - 1) If you include the Trading Partner profile referenced in your map, you may choose to send its referenced types, Trading Partner Contacts, Network Profiles, and Network Security Profiles, by clicking the Trading Partner Contacts, Network Profiles, and Network Security Profiles check boxes, respectively. You may choose to send its associated types by clicking the Control Numbers check box.
 - 2) If you include the Network Profile referenced in your map, you may choose to send its referenced type, Network Commands, by clicking the Network Commands check box.
 - 3) If you include the Network Security Profile referenced in your map, you may choose to send its referenced type, Network Security User Exits, by clicking the Network Security Exits check box.
- 3. When you have finished selecting the referenced and associated types you wish to export, click OK.

DataInterchange Client begins to export the item.

Control Strings

The screenshot shows the 'Control Strings' dialog box. It has a title bar with the text 'Control Strings' and a close button. The dialog is divided into several sections. The 'Referenced Types' section contains four checkboxes: 'Global Variables', 'Field Exits', 'Translation Tables', and 'Code Lists'. To the right of this section are buttons for 'Select All', 'Deselect All', 'OK', and 'Cancel'. The 'Associated Types' section contains a group box labeled 'Usages/Rules' with three radio buttons: 'None', 'All', and 'Select'. Below this is another group box labeled 'Usages/Rules Referenced Types' containing three checkboxes: 'Trading Partner Profiles', 'Translation Exits', and 'Envelope Profiles'. Below that is a group box labeled 'Trading Partner Referenced Types' containing three checkboxes: 'Trading Partner Contacts', 'Network Profiles', and 'Network Security Profiles'. Below that is a group box labeled 'Network Profile Referenced Type' containing a checkbox for 'Network Commands'. Below that is a group box labeled 'Network Security Profile Referenced Type' containing a checkbox for 'Network Security Exits'. At the bottom is a group box labeled 'Trading Partner Associated Types' containing a checkbox for 'Control Numbers'.



NOTE: Control strings have the following restrictions:

- They can be imported.
- They cannot be exported in fixed or comma-delimited formats.

◆ To complete the Export with Control Strings dialog box:

1. Click the check boxes of the following referenced types if you want to include them with the Control String you are exporting: Global Variables, Translation Tables, Field Exits, and Code Lists.
2. You may choose to include the Usages/Rules associated with the Control String you are exporting.
 - a. If you want to include usages or rules with the Control String you are exporting, click one of the three option buttons:
 - To include all usages or rules, click All.
 - To include no usages or rules, click None.
 - To select which usages or rules you want to export, click Select.

During the export, the appropriate usages or rules dialog box displays.

Select the usages or rules you want to export with the map control string, and click OK.

- b. If you include the usages or rules associated with your Control Strings export, you may choose to send their referenced types, Trading Partner Profiles, Translation User Exit (Pre-Translation and Post-Translation User Exits), Envelope Profiles by clicking the appropriate check boxes.
 - 1) If you include the Trading Partner profile referenced in the Control String you are exporting, you may choose to send its referenced types, Trading Partner Contacts, Network Profiles, and Network Security Profiles, by clicking the appropriate check boxes. You may choose to send its associated types by clicking the Control Numbers check box.
 - 2) If you include the Network Profiles referenced in the Trading Partner profile you are exporting, you may choose to send its referenced type, Network Commands, by clicking the Network Commands check box.
 - 3) If you include the Network Security Profiles referenced in the Trading Partner profile you are exporting, you may choose to send its referenced type by clicking the Network Security Profiles check box.
3. When you have finished selecting the referenced and associated types you wish to export, click OK.

DataInterchange Client begins to export the item.

PART 2. Setup

Activity Log profiles

The Activity Log profile allows you to set up message files that instruct DataInterchange to log a record of host activities, including:

- Comments on the status of an event, such as whether it is queued, sending, receiving, delivered, or completed
- Notes showing when a user gains access to a profile using the DataInterchange Host interface
- Detailed descriptions of program and database errors

By default, DataInterchange logs all activity messages in a single Activity Log profile, called EDIFFS. You need not change that profile or add another. It is more convenient to sort out activity messages, however, if you create separate Activity Log profiles for different purposes.

About Activity Logs

Activity Log profiles allow DataInterchange to log messages about DataInterchange activities in a file. This section provides an overview of the purpose of Activity Log profiles and how you set them up.

Purpose

All DataInterchange Host messages are placed in a single DB2 table named EDILOG. This table is logically partitioned into separate logs by the column ELAPPLID, which is referred to in DataInterchange as the log name. By default, DataInterchange Utility messages are inserted using the log name specified in the EDIFFS Activity Log profile. DataInterchange Facility messages are inserted using the log name specified in the EDIIMP Activity Log profile. Although you need not create any other Activity Log profiles, you may find that it is easier to sort messages if you store them under different log names created for different purposes.

For example, most companies run at least two different systems, Test and Production. They use the Test system while setting up a trading partner in order to make sure that such things as maps and profiles are working correctly. Once they are satisfied that their DataInterchange system correctly processes the trading partner's data, they move that trading partner to the Production system.

If all activity messages that DataInterchange logged were stored based on the default Activity Log profile, it would be difficult to separate messages related to the Test system from messages related to the Production system. Consequently, many companies at least create separate Activity Log profiles for their Test and Production systems. That allows them to track errors more easily when testing a new trading partner.

Another reason companies may use separate Activity Logs is to separate messages by application. For example, Purchasing may want to receive a record of all activity from its application. To provide that record, you would set up an Activity Log profile for the purchasing application. All activity bound to and from the purchasing system which DataInterchange logged would then be sorted to a file identified by the Activity Log profile for the purchasing application.

Setup overview

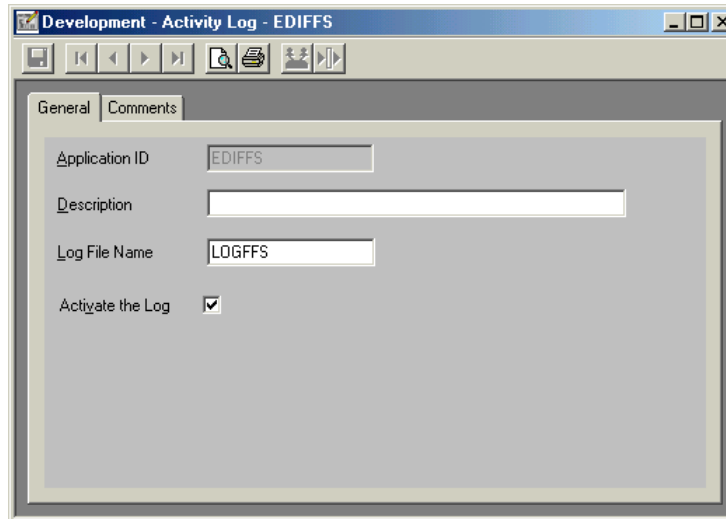
You set up and maintain Activity Log profiles through the Activity Log List window, which you access by clicking Setup on the DataInterchange Client Navigator bar. The Setup window, which contains tabs for DataInterchange Client's setup profiles, displays. Click the Activity Log tab, and the Activity Log List window displays.

	Application ID	Description	Lock	Updated Date/Time	Updated User ID
1	EDIFFS		No	10/22/2001 10:51:18 AM	admin
2	EDIMP		No	10/22/2001 10:51:18 AM	admin

This window displays a list of existing Activity Log profiles. Each row contains information about an Activity Log profile, each column contains data stored in the profile. Information in the columns displays in fields, drop-down lists, and check boxes in the Activity Log Editor window. The profile list window, however, also contains the date, time and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to "Creating a query" on page 287.

To view a profile or to add or change the information in these fields, double-click the row of the profile you wish to work with. The Activity Log Editor window displays, with the General tab in front.



The Activity Log Editor window contains two tabs: General and Comments. Use the General tab to enter and change information contained in the Activity Log profile. Use the Comments tab to type any comments you wish about the selected Activity Log profile.

Following are detailed procedures for creating new Activity Log profiles. For information on viewing, copying, editing, renaming, deleting, and printing profiles, see “Performing common file management tasks” on page 35. For information on exporting profiles, see “Exporting” on page 52.

Creating Activity Log profiles

The first step in creating Activity Log profiles is to decide whether you need to create separate files to log DataInterchange activity messages for particular systems or applications. Most companies at least create separate activity logs for their test and production systems. Some also create separate logs for different applications, such as purchasing or accounts receivable.

Create a new Activity Log profile to your DataInterchange installation when you want to separate DataInterchange messages by DataInterchange system, business application, or other criteria.

◆ To create an Activity Log profile:

1. Click Setup on the DataInterchange Client Navigator bar.

The Setup window displays.


2. Click the Activity Log tab.

A list of the existing Activity Log profiles displays.

3. Click New on the tool bar.

The Activity Log profile Editor window displays with the General tab open and all fields blank.

4. Complete the fields on the General tab. Required fields are preceded by a red dot.

Click  for field descriptions.

5. Click the Comments tab and type any comments you have about the Activity Log profile into the Comments field.

6. Select Save from the File menu to save the profile.

Application Defaults profiles

An Application Defaults profile allows you to identify your business applications, such as purchasing and accounts receivable, to DataInterchange and set specific DataInterchange processing defaults for an application. You need not create an Application Defaults profile for each application; you can use one profile for all applications, as long as you want DataInterchange to translate messages for each application in the same manner.

Processing defaults specified in an Application Defaults profile include:

- The name of the Activity Log used to store processing messages
- Whether you log DataInterchange transaction data to the Transaction Store
- Whether to gather management reporting statistics for an application

About Application Defaults

The Application Defaults profile identifies your business applications to DataInterchange and controls certain processing features. An Application Defaults profile mainly determines whether copies of transactions are stored in DataInterchange's Transaction Store database and the log files in which processing messages are stored.

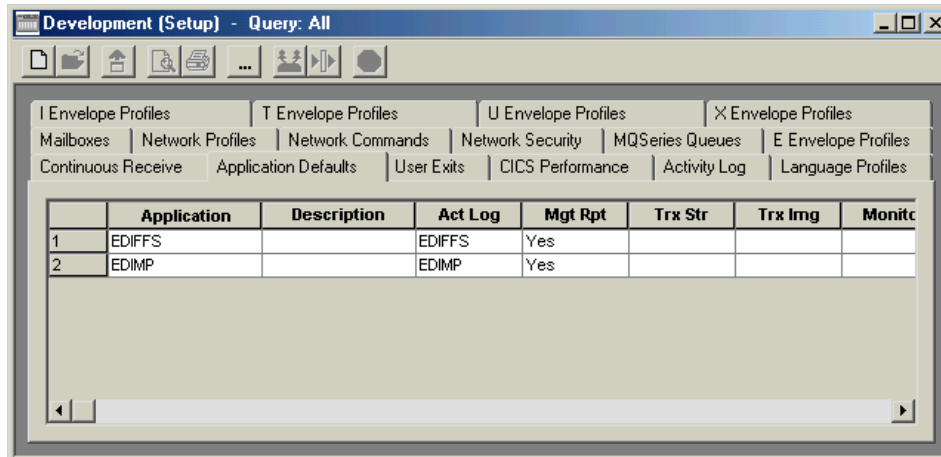
When an application invokes DataInterchange, it provides an Application ID. DataInterchange then searches for an Application Defaults profile to match the Application ID. If DataInterchange cannot find the matching Application Defaults profile, it searches for an Activity Log profile set up for the application. If it cannot find a specific Activity Log, it uses the DataInterchange default Activity Log, EDIFFS.

Should you decide, for example, not to send images of functional acknowledgments for the invoicing application to the Transaction Store database, you would set up an Application Defaults profile for your company's invoicing system with such instructions.

For more information on the Transaction Store, see the *DataInterchange Administrator's Guide*. For more information on processing messages, see Chapter 5, "Activity Log profiles," on page 71.

Setup overview

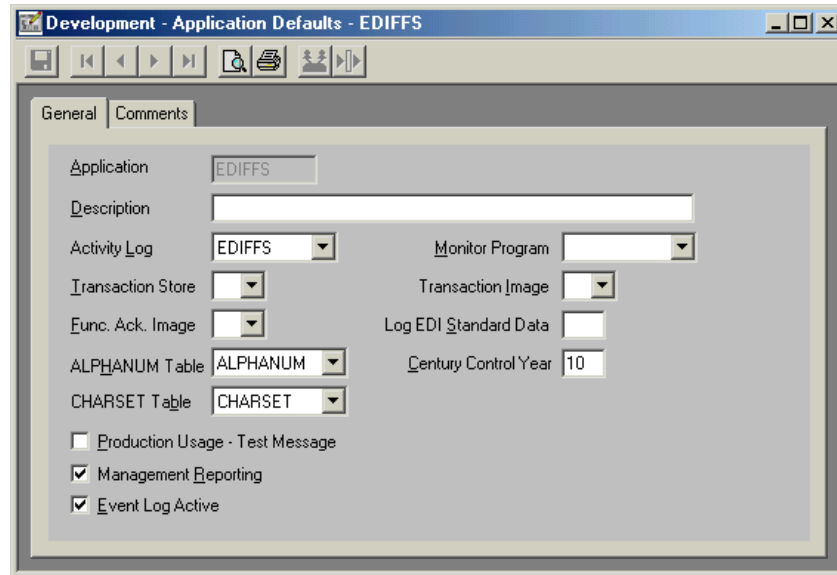
You set up and maintain Application Defaults profiles through the Application Defaults List window, which you access by clicking Setup on the DataInterchange Client Navigator bar. The Setup window, which contains tabs for DataInterchange Client's setup profiles, displays. Click the Application Defaults tab, and the Application Defaults List window displays.



This window displays a list of existing Application Defaults. Each row contains information about an Application Defaults profile; each column contains data stored in that profile. Information in the columns displays in fields, drop-down lists, and check boxes in the Application Defaults Editor window. The profile list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to "Creating a query" on page 287.

To view a profile or to add or change the information in these fields, double-click the profile you wish to work with. The Application Defaults Editor window displays, with the General tab in front.



The Application Defaults Editor window contains two tabs: General and Comments. Use the General tab to enter and change information contained in the Application Defaults profile. Use the Comments tab to type any comments you wish about the selected Application Defaults profile.

Following are detailed procedures for creating new Application Defaults profiles. For information on viewing, copying, editing, renaming, deleting, and printing profiles, see “Performing common file management tasks” on page 35. For information on exporting profiles, see “Exporting” on page 52.

Creating Application Defaults profiles

The first step in creating Application Defaults profiles is to identify the business applications you are linking to DataInterchange. Then you must decide how you want DataInterchange to handle the logging of each application’s EDI transactions and DataInterchange activity logs.

Create a new Application Defaults profile when you want to treat a particular business application’s processing defaults differently from either the DataInterchange default values or other applications you have previously set up.

→To create an Application Defaults profile:

1. Click Setup on the DataInterchange Client Navigator bar.


The Setup window displays.

2. Click the Application Defaults tab.

A list of the existing Application Defaults profiles displays.

3. Click New on the tool bar.

The Application Defaults Editor window displays with the General tab open and all fields blank.

4. Complete the fields on the General tab. Required fields are preceded by a red dot.
Click  for field descriptions.
5. Click the Comments tab and type any comments you have about the Application Defaults profile into the Comments field.
6. Click Save on the tool bar to save the profile.

Continuous Receive profiles

A Continuous Receive profile allows you to initiate event-driven processes using DataInterchange/CICS and Expedite/CICS. (Expedite/CICS is the communication package for connecting to IBM Global Services.) Event-driven processes means that the receipt of a message from a trading partner triggers the translation process. Under typical processing, your system polls your network mailbox periodically, and any messages are processed in batches.

Through Continuous Receive, you can automate the process of receiving and processing files sent to your mailbox by trading partners. DataInterchange Client allows you to set up a Continuous Receive profile to:

- Receive and deenvelope standard data
- Translate the standard data to a business application's format
- Automatically start a response application, which receives the translated business application data into a temporary storage queue
- Automatically receive network acknowledgments



NOTE: The continuous receive feature is used only with receive maps. It can not be used with send maps or data transformation maps.

About Continuous Receive profiles

Continuous Receive profiles allow you to automate receiving and processing of messages using DataInterchange/CICS. In other words, Continuous Receive profiles allow you to initiate event-driven processes, in which the receipt of a message from a trading partner initiates the translation process, as opposed to periodic polling of a network mailbox to check for messages.

The Continuous Receive process can also be used to perform event-driven EDI using DataInterchange/CICS and MQSeries message queues. For more information, see “Using MQSeries with Continuous Receive” on page 82.

You need to add a Continuous Receive profile for each unique continuous receive session you want to run. For example, you can add a Continuous Receive profile for each network mailbox, or one for each transaction type, or one for a specific trading partner. You can also create Continuous Receive profiles to define different ways of processing transactions, including:

- Translating the data and delivering it to the receiving business application
- Saving untranslated data in the Transaction Store database
- Providing the output in C (control) and D (data) records or as raw data
- Starting a response transaction or application after DataInterchange has finished its processing
- Automatically receiving network acknowledgments

For information about starting or stopping continuous receive requests, or for information about continuous receive cleanup, see the *DataInterchange Programmer's Reference*.

Setup overview

You set up and maintain Continuous Receive profiles through the Continuous Receive List window, which you access by clicking Setup on the DataInterchange Client Navigator bar. The Setup window, which contains tabs for DataInterchange Client's setup profiles, displays. Click the Continuous Receive tab, and the Continuous Receive List window displays.

	Cont Recv ID	Description	Active	Mailbox ID	TP Nickname	Print
1	TST001	Test file	No		ANY	M
2	TST002	Test file	No		TUTORPARTNER	M
3	TST003	Test file	No		TUTORPARTNER	TI
4	TST004	Test file	No		TUTORPARTNER	M

This window displays a list of existing Continuous Receive profiles. Each row contains information about a Continuous Receive profile; each column contains data stored in the profile. Information in the columns displays in fields, drop-down lists, and check boxes in the Continuous Receive Editor window. The profile list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to “Creating a query” on page 287.

To view a profile or to add or change the information in these fields, double-click the row of the profile you wish to work with. The Continuous Receive Editor window displays, with the General tab in front.

The screenshot shows the 'Development - Continuous Receive - TST002' window. It features a toolbar with icons for file operations and a tabbed interface with 'General' and 'Comments' tabs. The 'General' tab contains the following fields and options:

- Cont. Recv. ID:** TST002
- Description:** Test file
- TP Nickname:** TUTORPARTNER (dropdown)
- Mailbox ID:** (dropdown)
- Print Type:** MQ (dropdown)
- Print:** (text field)
- Exception Type:** VS (dropdown)
- Exception:** (text field)
- Response Type:** PG (dropdown)
- Response:** (text field)
- Msg. User Class:** (text field)
- Purge Interval:** (text field)
- FA TS Queue:** (text field)
- Additional Records:** 1 (dropdown)
- Application ID:** (text field)
- Language:** (dropdown)
- User Field:** (text field)

At the bottom, there are three groups of checkboxes:

- ☐ Active, ☐ Translate, ☐ Duplicate Envelopes, ☐ Pageable Translation
- ☐ Raw Data, ☐ Syncpoints, ☐ Network Acks Only
- ☐ Delay FAs, ☐ Deenvelope only, ☐ SAP Status Tracking

The Continuous Receive Editor window contains two tabs: General and Comments. Use the General tab to enter and change information contained in the Continuous Receive profile. Use the Comments tab to type any comments you wish about the selected Continuous Receive profile.

Following are detailed procedures for creating new Continuous Receive profiles. For information on viewing, copying, editing, renaming, deleting, and printing profiles, “Performing common file management tasks” on page 35. For information on exporting profiles, see “Exporting” on page 52.


Creating Continuous Receive profiles

The first step in creating Continuous Receive profiles is to decide which trading partners, network IDs, or message types you want to set up for Continuous Receive. Base your decision on business needs.

If your business runs an assembly line that requires a continuous stream of parts from suppliers, you may want to set up a Continuous Receive profile for materials release transactions. DataInterchange will then process materials releases immediately, and your manufacturing applications will contain up-to-date information on materials inventory.

Create a new Continuous Receive profile for a trading partner, network ID, or transaction set when you need to receive and process transactions sent to your mailbox immediately.

◆ To create a Continuous Receive profile:

1. Click Setup on the DataInterchange Client Navigator bar.
The Setup window displays.
2. Click the Continuous Receive tab.
A list of the existing Continuous Receive profiles displays.
3. Click New on the tool bar.
The Continuous Receive Editor window displays with the General tab open and all fields blank.
4. Complete the fields on the General tab. Required fields are preceded by a red dot.
Click  for field descriptions.
5. Click the Comments tab and type any comments you have about the Continuous Receive profile into the Comments field.
6. Click Save on the tool bar to save the profile.

Using MQSeries with Continuous Receive

The Continuous Receive process can be used to perform event-driven EDI using MQSeries message queues. To do this, perform the following steps:

1. Your CICS and MQSeries administrators must enable the CICS region for MQSeries support. As part of this process, they must define an MQSeries trigger event queue, which the MQSeries supplied transaction CKTI monitors within the CICS region. For example, assume CICS1.TRIGGER is the name of this queue.
2. Your MQSeries administrator must define an MQSeries process via the DEFINE PROCESS command to provide MQSeries with the information about the DataInterchange transaction EDIQ. A sample of the DEFINE command follows:

```
DEFINE PROCESS(EDIPROCESS) APPLICID(EDIQ) APPLTYPE(CICS)
```

3. Using the MQSeries DEFINE QLOCAL command, your MQSeries administrator must define the MQSeries message queue that will receive envelopes. A sample of the command follows:

```
DEFINE QLOCAL(EDIRECEIVE) INITQ(CICS1.TRIGGER) PROCESS(EDIPROCESS)
      TRIGGER TRIGTYPE(FIRST) TRIGDATA('CRPROF=MQ1 MQPROF=RECV1')
```

The TRIGDATA field is critical. Within this field are two keyword-value combinations. The CRPROF keyword identifies the name of the DataInterchange Continuous Receive profile to use when data is received on the message queue. The MQPROF keyword identifies a DataInterchange MQSeries Queue profile associated with this message queue. Both of these keyword-value combinations are mandatory. The order in which the two keyword-value combinations are specified is not important.

In this example, only one MQSeries message queue will be used for event driven EDI. You can define any number of queues in your environment. If DataInterchange and MQSeries information is set up correctly, DataInterchange will process any number of queues.

The MQSeries definitions are now complete.

4. Within DataInterchange, you must define an MQSeries Queue profile. The MQSeries Queue profile associates the physical MQSeries message queue name with a logical name used within DataInterchange. The MQSeries Queue profile also includes information that is used when processing the message queue. In the example, the name of the MQSeries Queue profile is RECV1. When creating RECV1, the full queue name will be set to EDIRECEIVE.

For more information, see “MQSeries Queue profiles” on page 103.

5. You must also define a Continuous Receive profile. In this example, the profile name is MQ1. The TP Nickname, Mailbox ID, MSG User Class, and Network Acks Only fields should be left blank. They are ignored when the Continuous Receive profile is used for MQSeries processing.

DataInterchange automatically processes all data written to the MQSeries message queue as soon as MQSeries dispatches the trigger event messages.

CICS Performance profiles

The CICS Performance profile specifies whether DataInterchange should use the host system's advanced CICS performance capabilities. DataInterchange Client allows you to enter the information that DataInterchange needs to increase performance under CICS.

About CICS Performance profiles

CICS Performance profiles allow you to define CICS performance characteristics for specific CICS systems in your DataInterchange host environment. This section provides an overview of the purpose of CICS Performance profiles and how you set them up.

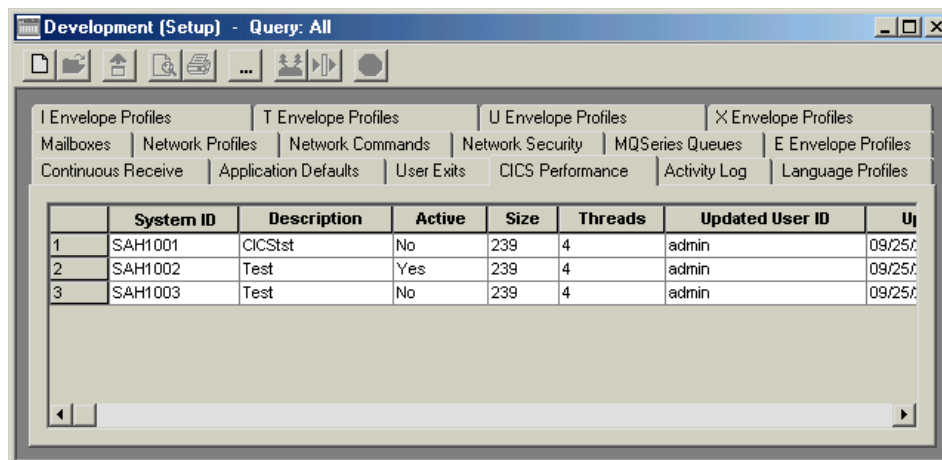
Purpose

As an online environment, CICS is designed to execute commands as quickly as possible. DataInterchange Client allows you to set up CICS Performance profiles that provide information required by DataInterchange to optimize performance in a CICS environment.

Once you have set up the profile to make CICS performance enhancements active, DataInterchange uses core memory to hold data. When the translator requires that data, it can retrieve it from core memory, rather than the database, which increases system performance.

Setup overview

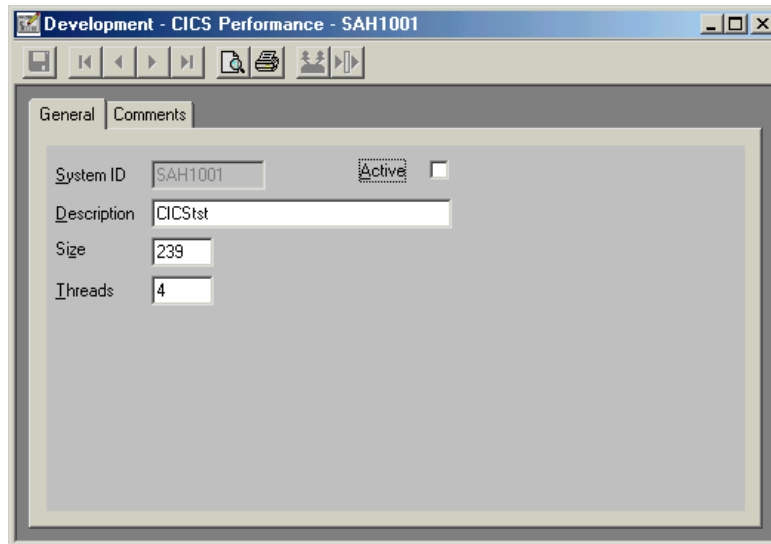
You set up and maintain CICS Performance profiles through the CICS Performance List window, which you access by clicking Setup on the DataInterchange Client Navigator bar. The Setup window, which contains tabs for DataInterchange Client's setup profiles, displays. Click the CICS Performance tab, and the CICS Performance List window displays.



This window displays a list of existing CICS Performance profiles. Each row contains information about a CICS Performance profile; each column contains data stored in the profile. Information in the columns displays in fields, drop-down lists, and check boxes in the CICS Performance Editor window. The profile list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to “Creating a query” on page 287.

To view a profile or to add or change the information in these fields, double-click the row of the profile you wish to work with. The CICS Performance Editor window displays, with the General tab in front.




The CICS Performance Editor window contains two tabs: General and Comments. Use the General tab to enter and change information contained in the CICS Performance profile. Use the Comments tab to type any comments you wish about the selected CICS Performance profile.

Following are detailed procedures for creating new CICS Performance profiles. For information on viewing, copying, editing, renaming, deleting, and printing profiles, see “Performing common file management tasks” on page 35. For information on exporting profiles, see “Exporting” on page 52.

Creating CICS performance profiles

The first step in creating CICS Performance Profiles is determining the name of the CICS performance regions that will use the DataInterchange performance feature.

◆ To create a CICS Performance profile:

1. Click Setup on the DataInterchange Client Navigator bar.
The Setup window displays.
2. Click the CICS Performance tab.
A list of the existing CICS Performance profiles displays.
3. Click New on the tool bar.
The CICS Performance Editor window displays with the General tab open.
4. Complete the fields on the General tab. Required fields are preceded by a red dot.
Click  for field descriptions.
5. Click the Comments tab and type any comments you have about the CICS Performance profile into the Comments field.
6. Click Save on the tool bar to save the profile.

Envelope profiles

The envelope profiles have one field for each element in the envelope standard. The profiles provide literal or constant data for building header or trailer segments for transaction sets, messages, functional groups, and interchanges. You supply only the values that need to be populated and for which a value is not provided by another source. See the individual profile for the envelope standard you are using.

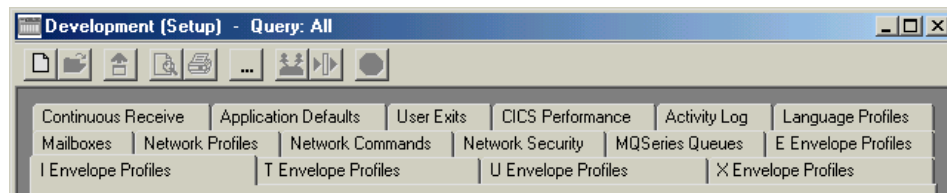
About Envelope profiles

Each envelope profile has a name and a description. The remaining fields represent the data elements in the envelope standard. The field names are designed to make cross referencing easy. For example, the field UNB03 is the third data element in the UNB segment.

DataInterchange supports generic envelope profiles. Refer to the *DataInterchange Administrator's Guide* for additional information. A generic profile name can consist of 1-to-6-characters (base name). When a generic envelope profile is accessed by a send usage, receive usage, or map rule, DataInterchange appends the envelope profile suffix from the trading partner profile to the base name to determine which envelope profile to access during enveloping.

Setup overview

You set up and maintain envelope profiles through the envelope list windows, which you access by clicking Setup on the DataInterchange Client Navigator bar. The Setup window, which contains DataInterchange Client's setup profiles, displays.



Each envelope profile (E, I, T, U, and X) has its own tab. Click the specific envelope profile tab, and that Envelope Profile List window displays.

The window displays a list of existing envelope profiles for that specific tab. Each row contains information about an envelope profile; each column contains data stored in that profile. Information in the columns displays in fields, drop-down lists, and check boxes in the Envelope Profile Editor window. The envelope profile list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to “Creating a query” on page 287.

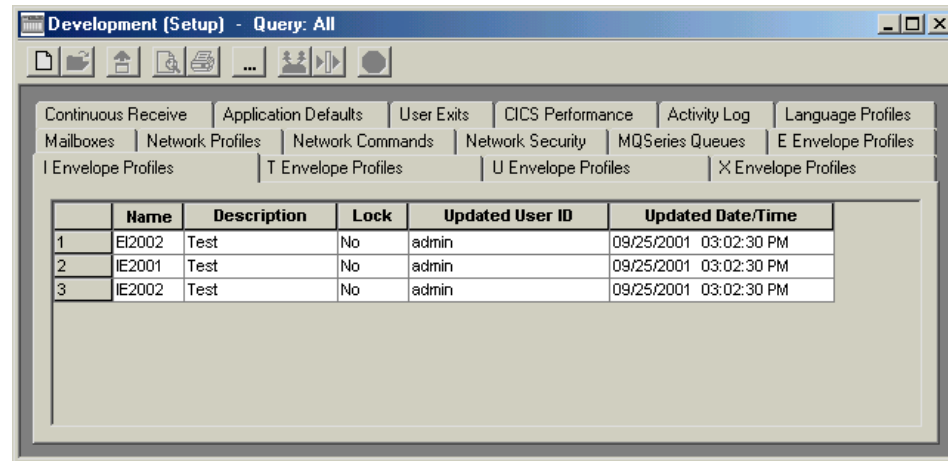
To view a profile or to add or change the information in these fields, double-click the row of the profile you wish to work with. The specific envelope profile editor window displays, with the General tab in front. For example, if you clicked E Envelope, the General tab for E Envelope profiles displays.

Creating Envelope profiles

◆ To create Envelope Profiles

1. Click Setup on the DataInterchange Client Navigator bar.

The Setup window displays.

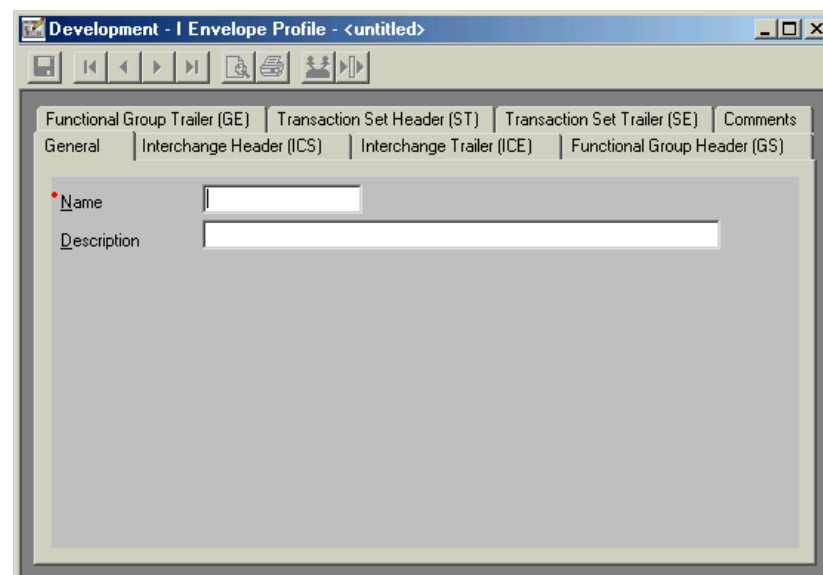


2. Click the tab for the envelope profile (E, I, T, U, or X) that you want to create.


A list of the existing profiles for that specific envelope profile type displays.

3. Click New on the tool bar.

The Envelope Profile Editor window displays with the General tab open.



4. Complete the fields on the General tab, which are the same for all envelope profiles. The required field is preceded by a red dot.

Click  for field descriptions.

5. To complete the specific envelope profile you are working on, refer to the section listed below for the type of profile you are creating:

UN/EDIFACT (E) Envelope profile	92
ICS (I) Envelope profile	92
UN/TDI (T) Envelope profile	93
UCS (U) Envelope profile	93
X12 (X) Envelope profile	93



NOTE: Each envelope profile window has a comments tab.


UN/EDIFACT (E) Envelope profile

After completing the General tab:

1. Click each of the tabs, in turn, completing the fields as needed.

Tabs

Interchange Header (UNB)
Interchange Trailer (UNZ)
Functional Group Header (UNG)
Functional Group Trailer (UNE)
Message Header (UNH)
Message Trailer (UNT)

Click  for field names and descriptions.

2. Click Save on the tool bar to save the profile.


ICS (I) Envelope profile

After completing the General tab:

1. Click each of the tabs, in turn, completing the fields as needed.

Tabs

Interchange Header (ICS)
Interchange Trailer (ICE)
Functional Group Header (GS)
Functional Group Trailer (GE)
Transaction Set Header (ST)
Transaction Set Trailer (SE)

Click  for field names and descriptions.

2. Click Save on the tool bar to save the profile.


UN/TDI (T) Envelope profile

After completing the General tab:

1. Click each of the tabs, in turn, completing the fields as needed.

Tabs

Interchange Header (STX)
Interchange Trailer (END)
Message Header (MHD)
Message Trailer (MTR)

Click  for field names and descriptions.

2. Click Save on the tool bar to save the profile.


UCS (U) Envelope profile

After completing the General tab:

1. Click each of the tabs, in turn, completing the fields as needed.

Tabs

Interchange Header (BG)
Interchange Trailer (EG)
Functional Group Header (GS)
Functional Group Trailer (GE)
Transaction Set Header (ST)
Transaction Set Trailer (SE)

Click  for field names and descriptions.

2. Click Save on the tool bar to save the profile.


X12 (X) Envelope profile

After completing the General tab:

1. Click each of the tabs, in turn, completing the fields as needed.

Tabs

Interchange Header (ISA)
Interchange Trailer (IEA)
Functional Group Header (GS)
Functional Group Trailer (GE)
Transaction Set Header (ST)
Transaction Set Trailer (SE)

Click  for field names and descriptions.

2. Click Save on the tool bar to save the profile.

Language profiles

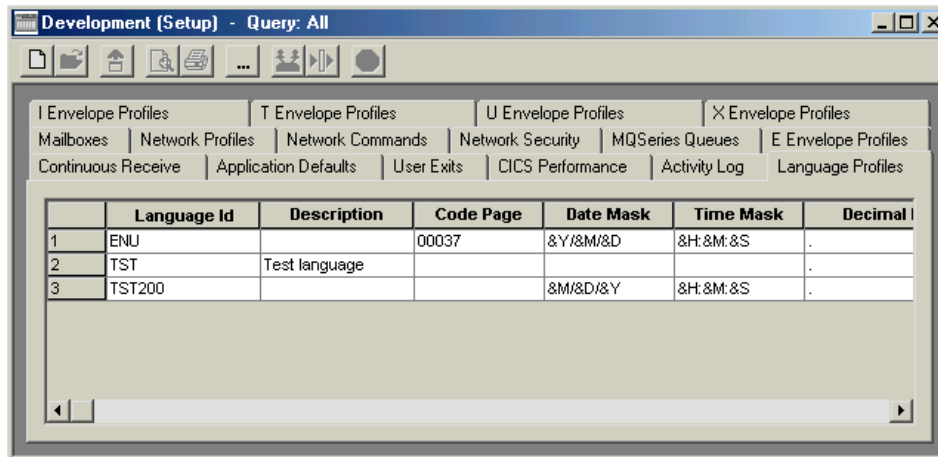
The Language profile is used to handle language and character set issues on the server which is used to perform translations. It does not effect the PC on which DI Client is running. A Language profile contains language variables, such as the formats you use for the date and time. Initially, it contains a member for the language version you install. For example, ENU is the profile for US English. You can add additional members or tailor some parts of the member that DataInterchange supplies.

About Language profiles

Language profiles are used when you need unique language variables for translation. The Language profile is used by the DataInterchange translator. It is not used in DataInterchange Client.

Setup overview

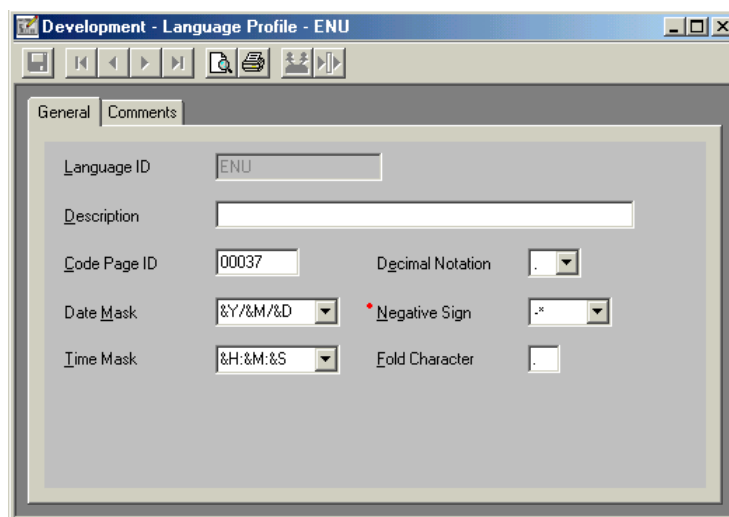
You set up and maintain Language profiles through the Language Profiles List window, which you access by clicking Setup on the DataInterchange Client Navigator bar. The Setup window, which contains DataInterchange Client's setup profiles, displays. Click the Language Profiles tab, and the Language Profile List window displays.



This window displays a list of existing Language profiles. Each row contains information about a Language profile; each column contains data stored in the profile. Information in the columns displays in fields, drop-down lists, and check boxes in the Language Profile Editor window. The profile list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to "Creating a query" on page 287.

To view a profile or to add or change the information in these fields, double-click the row of the profile you wish to work with. The Language Profile Editor window displays, with the General tab in front.



The Language Profile Editor window contains two tabs: General and Comments. Use the General tab to enter and change information contained in the Language profile. Use the Comments tab to type any comments you wish about the selected Language profile.

Following are detailed procedures for creating new Language profiles. For information on viewing, copying, editing, renaming, deleting, and printing profiles, see “Performing common file management tasks” on page 35. For information on exporting profiles, see “Exporting” on page 52.

Mailbox profiles

A Mailbox profile allows you to identify to DataInterchange the groups in your organization that receive documents. Mailbox profiles, therefore, are set up for mailbox users, not the network mailboxes themselves. More than one user can use the same network mailbox, but each user must have his own mailbox profile.

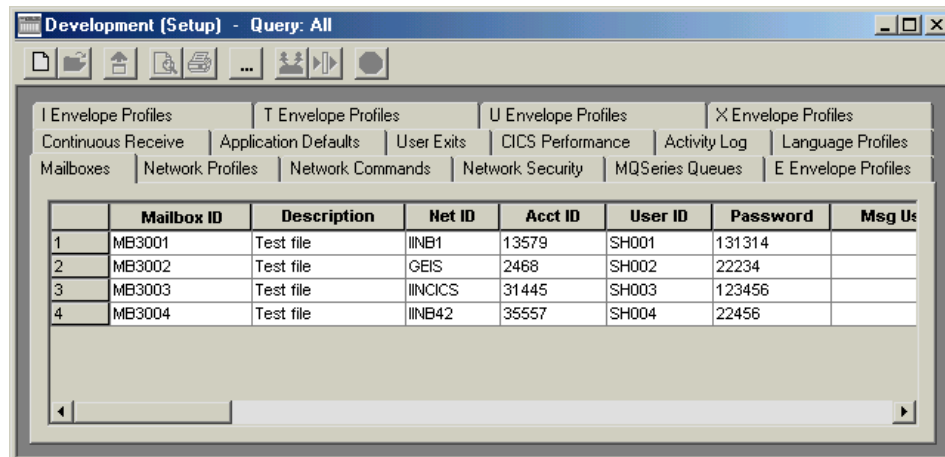
About mailbox profiles

Mailbox profiles contain the information that DataInterchange needs to identify the individuals and groups in your organization that receive documents to be translated. Each individual or group requires its own Mailbox profile.

A Mailbox profile may identify the owner of a network mailbox, but it does not only identify mailbox owners. Many individuals or groups can use the same network mailbox. In order to process documents to meet the requirements of a particular mailbox user, DataInterchange allows you to set up a custom Mailbox profile for each user.

Setup overview

You set up and maintain Mailbox profiles through the Mailbox List window, which you access by clicking Setup on the DataInterchange Client Navigator bar. The Setup window, which contains DataInterchange Client's setup profiles, displays. Click the Mailboxes tab, and the Mailbox List window displays.



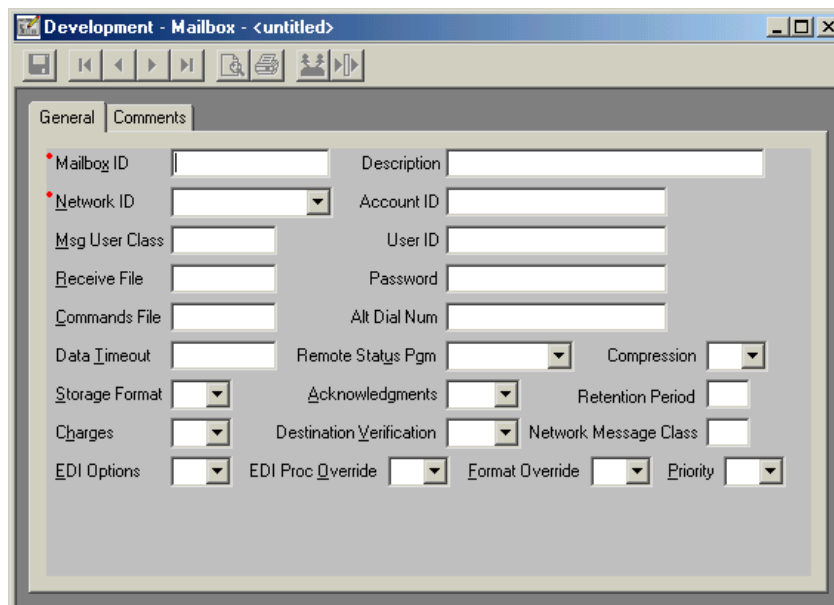
The screenshot shows a window titled "Development (Setup) - Query: All". It has a menu bar with options like "I Envelope Profiles", "T Envelope Profiles", "U Envelope Profiles", and "X Envelope Profiles". Below the menu bar is a toolbar with icons for file operations and navigation. The main area displays a table with the following data:

	Mailbox ID	Description	Net ID	Acct ID	User ID	Password	Msg Us
1	MB3001	Test file	IINB1	13579	SH001	131314	
2	MB3002	Test file	GEIS	2468	SH002	22234	
3	MB3003	Test file	IINCICS	31445	SH003	123456	
4	MB3004	Test file	IINB42	35557	SH004	22456	

This window displays a list of existing Mailbox profiles. Each row contains information about a Mailbox profile; each column contains data stored in that profile. Information in the columns displays in fields, drop-down lists, and check boxes in the Mailbox Editor window. The profile list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to "Creating a query" on page 287.

To view a profile or to add or change the information in these fields, double-click the row of the profile you wish to work with. The Mailbox Editor window displays, with the General tab in front.



The screenshot shows a window titled "Development - Mailbox - <untitled>". It has a menu bar with options like "General" and "Comments". Below the menu bar is a toolbar with icons for file operations and navigation. The main area displays a form with the following fields:

Mailbox ID	Description
Network ID	Account ID
Msg User Class	User ID
Receive File	Password
Commands File	Alt Dial Num
Data Timeout	Remote Status Pgm
Storage Format	Compression
Charges	Acknowledgments
EDI Options	Retention Period
EDI Proc Override	Destination Verification
Format Override	Network Message Class
Priority	

The Mailbox Editor window contains two tabs: General and Comments. Use the General tab to enter and change information contained in the Mailbox profile. Use the Comments tab to type any comments you wish about the selected Mailbox profile.

Following are detailed procedures for creating new Mailbox profiles. For information on viewing, copying, editing, renaming, deleting, and printing profiles, see “Performing common file management tasks” on page 35. For information on exporting profiles, see “Exporting” on page 52.

Creating Mailbox profiles

The first step in creating Mailbox profiles is determining:

- What individuals or groups require DataInterchange services.
- Which network mailboxes those individuals or groups will use.

For example, you may receive a request from a user or group within your organization that requires the DataInterchange services to connect to a trading partner. That user or group requires a Mailbox profile.

You must decide which network mailbox will best meet that user’s needs. That decision may depend on the trading partner’s network communications capabilities and which network they can use to communicate with your organization.

A Mailbox profile may be tied to a Network profile. A new trading partner relationship may require that you create a new Network profile, as well as a new Mailbox profile. You may also find that you can add a trading partner to existing Mailbox profiles, as long as the user requiring a new trading partner connection already has a Mailbox profile and the new trading partner can communicate under that profile’s specifications.

You create a new Mailbox profile when you receive a request for DataInterchange services from a user that either is not already connected to DataInterchange or requires different communication specifications to meet the needs of a new trading partner relationship.

◆ To create a Mailbox profile:

1. Click Setup on the DataInterchange Client Navigator bar.

The Setup window displays.


2. Click the Mailboxes tab.

A list of the existing Mailbox profiles displays.

3. Click New on the tool bar.

The Mailbox Editor window displays with the General tab open and all fields blank.

4. Complete the fields on the General tab. Required fields are preceded by a red dot in DataInterchange Client.

Click  for field descriptions.

5. Click the Comments tab and type any comments you have about the Mailbox profile into the Comments field.

6. Click Save on the tool bar to save the profile.

MQSeries Queue profiles

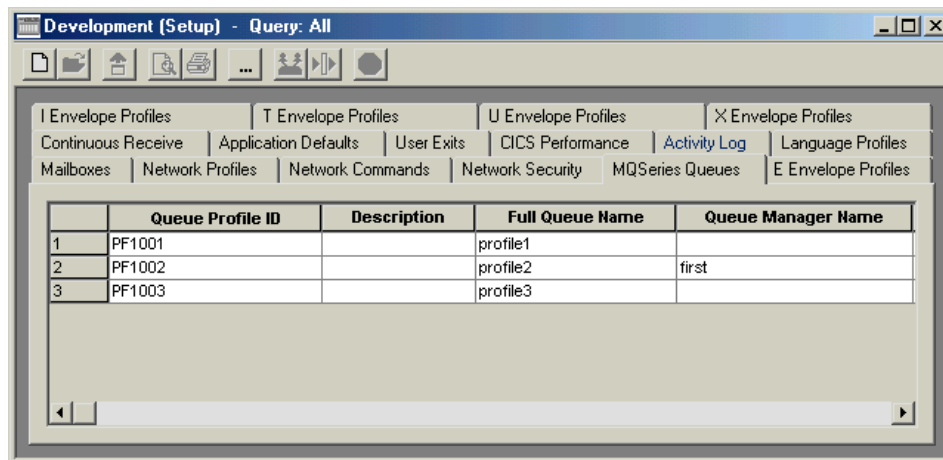
DataInterchange Client's MQSeries profile is used to associate logical names with real MQSeries message queues. To use the MQSeries support in DataInterchange, you must have MQSeries for MVS/ESA Version 1 Release 2 or later, or CICS, installed, and have MQSeries running in order for DataInterchange to access the message queues.

About MQSeries Queue profiles

The MQSeries Queue Profile provides a way to associate processing options DataInterchange will use whenever the MQSeries profile is supplied. MQSeries message queues can be used in place of most sequential files, and they can be the source to obtain a document or the target when a document will be written; and in CICS, MQSeries Queues can be used in event-driven processes. You can use MQSeries Queue profiles as part of a logical network where enveloped data is sent to and received from MQSeries message queues.

Setup overview

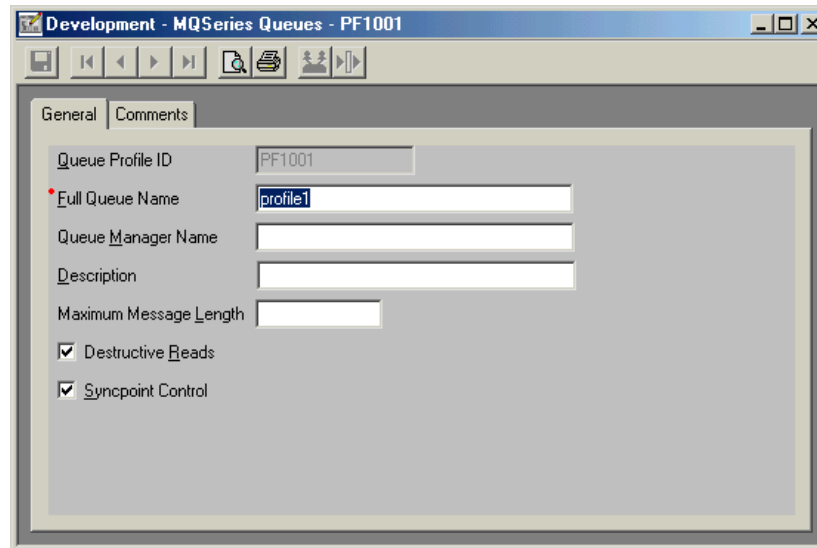
You set up and maintain MQSeries Queue profiles through the MQSeries Queues List window, which you access by clicking Setup on the DataInterchange Client Navigator bar. The Setup window, with tabs for DataInterchange Client's setup profiles, displays. Click the MQSeries Queues tab, and the MQSeries Queues List window displays.



This window displays a list of existing MQSeries Queue profiles. Each row contains information about a MQSeries Queue profile; each column contains data stored in the profile. Information in the columns displays in fields, drop-down lists, and check boxes in the MQSeries Queues Editor window. The profile list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to "Creating a query" on page 287.

To view a profile or to add or change the information in these fields, double-click the row of the profile you wish to work with. The MQSeries Queue Editor window displays, with the General tab in front.



The MQSeries Queues Editor window contains two tabs: General and Comments. Use the General tab to enter and change information contained in the MQSeries Queue profile. Use the Comments tab to type any comments you wish about the selected MQSeries Queue profile.

Following are detailed procedures for creating new MQSeries Queue profiles. For information on viewing, copying, editing, renaming, deleting, and printing profiles, see “Performing common file management tasks” on page 35. For information on exporting profiles, see “Exporting” on page 52.

Creating MQSeries Queue profiles

Create a new MQSeries Queue profile for each queue you will use.

◆ To create a MQSeries Queue profile:

1. Click Setup on the DataInterchange Client Navigator bar.

The Setup window displays.


2. Click the MQSeries Queues tab.

A list of the existing MQSeries Queue profiles displays.

3. Click New on the tool bar.

The MQSeries Queues Editor window displays with the General tab open and all fields blank.

4. Complete the fields on the General tab.

Click  for field descriptions.

5. Click the Comments tab and type any comments you have about the MQSeries Queue profile in the Comments field.

6. Select Save from the File menu to save the profile.

Network profiles

Network profiles define for DataInterchange the characteristics of the networks you use for communications with trading partners. DataInterchange Client is shipped with the network profiles required to communicate with several major networks (Table 12 on page 108), and if you use those networks, you need not create any new network profiles.

If you are not using any of those networks, you need to create a new network profile. You may also bypass networks altogether and make a direct point-to-point connection between DataInterchange and your trading partner's system.

Network profiles are also used to assist in identifying MQSeries message queues used to send and receive documents.



ATTENTION: Network profiles may need to have a corresponding set of Network Commands profiles to define the commands required to communicate with the network. To create Network Commands profiles, see Chapter 14, “Network Commands profiles,” on page 113. For further details in communicating with networks, see the *DataInterchange Programmer's Reference*.

About Network profiles

DataInterchange supports two means of communicating with trading partners: network communications and direct communications. Network communications move data from DataInterchange to a mailbox supported by a network service provider or a company. Direct communications are just that: communication links between DataInterchange and a trading partner's system.

Create a Network profile when using MQSeries to exchange EDI data with your trading partners. For MQSeries communication with trading partners, the Network profile identifies the MQSeries Queue profiles associated with the message queues used to send and receive documents. For more information, see "MQSeries Queue profiles" on page 103.

DataInterchange reads network parameters through Network profiles. Any network you use must have a profile defined for DataInterchange. To make network communications easier, DataInterchange Client is shipped with profiles for IBM Global Services, and the General Electric Information Services (GEIS) as described in Table 12.

Table 12. Network Profiles Shipped with DataInterchange Client

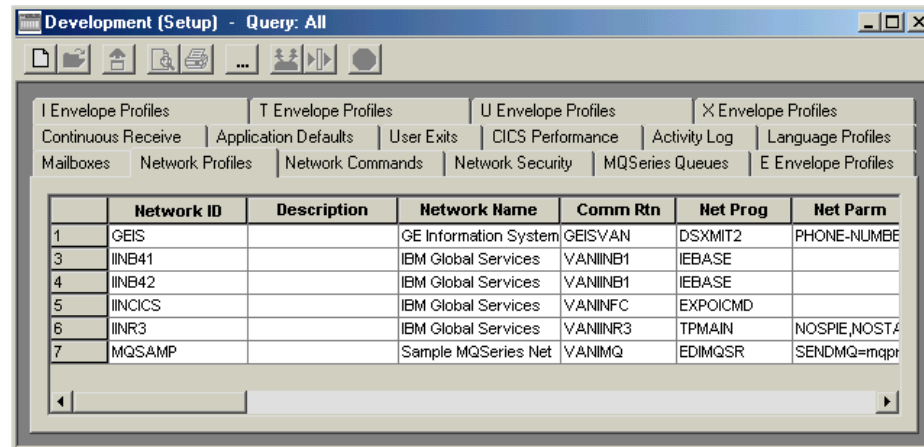
Profile	Communication Routine	Network Program	Used With:
GEIS	GEISVAN	DSXMIT2	General Electric Information Services Company
IINB41	VANIINB1	IEBASE	Expedite Base/MVS Version 4 Release 1
IINB42	VANIINB1	IEBASE	Expedite Base/MVS Version 4 Release 2 and higher
IINCICS	VANINFC	EXPOICMD	Expedite/CICS
MQSAMP	VANIMQ	EDIMQSR	Sample MQSeries Network

If you are not using IBM Global Services or the General Electric Information Services (GEIS) network, or if you want to use a point-to-point connection that bypasses the mailbox, or use MQSeries queues to send and receive data, you must create a network profile.

DataInterchange can communicate with any generalized network or make point-to-point communications directly with a trading partner's system. For more information on communicating with other networks and making point-to-point connections, see the *DataInterchange Programmer's Reference*.

Setup overview

You set up and maintain network profiles through the Network Profiles List window, which you access by clicking Setup on the DataInterchange Client Navigator bar. The Setup window, which contains DataInterchange Client's setup profiles, displays. Click the Network Profiles tab, and the Network Profiles List window displays.



	Network ID	Description	Network Name	Comm Rtn	Net Prog	Net Parm
1	GEIS		GE Information System	GEISVAN	DSXMIT2	PHONE-NUMBE
3	IINB41		IBM Global Services	VANINB1	IEBASE	
4	IINB42		IBM Global Services	VANINB1	IEBASE	
5	IINCIS		IBM Global Services	VANINFC	EXPOICMD	
6	IINR3		IBM Global Services	VANINR3	TPMAIN	NOSPIE,NOSTA
7	MQSAMP		Sample MQSeries Net	VANIMQ	EDIMQSR	SENDMQ=mcpr

This window displays a list of networks your company can use to communicate with trading partners. Each row contains information about a Network profile; each column contains data stored in that profile. Information in the columns displays in fields, drop-down lists, and check boxes in the Network Profiles Editor window. The profile list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to “Creating a query” on page 287.

To view a profile or to add or change the information in these fields, double-click the row of the profile you wish to work with. The Network Profiles Editor window displays, with the General tab in front.

The Network Profiles Editor window contains two tabs: General and Comments. Use the General tab to enter and change information contained in Network profiles. Use the Comments tab to type any comments you wish about the selected Network profile.

Following are detailed procedures for creating Network profiles. For information on viewing, copying, editing, renaming, deleting, and printing profiles, “Performing common file management tasks” on page 35. For information on exporting profiles, see “Exporting” on page 52.

Creating Network profiles

The first step to working with Network profiles is determining which networks your trading partners use, or which MQSeries message queues documents will be received from or sent to. You need to create Network profiles for any networks not listed in Table 12 on page 108. You also need to create a network profile to identify MQSeries message queues that you will receive documents from or send documents to.

Once you know which Network profiles you need to create, gather the information required to complete the profile. Review the fields that display in the Network profile. You need the name of a technical contact at the network for which you are creating a profile to supply some of the information required.

Create a new Network profile to communicate with a network not listed in Table 12 on page 108.

◆ To create a network profile:

1. Click Setup on the DataInterchange Client Navigator bar.

The Setup window displays.


2. Click the Network Profiles tab.

A list of the existing Network profiles displays.

3. Click New on the tool bar.

The Network Profiles Editor window displays with the General tab open and all the fields blank.

4. Complete the fields on the General tab. Required fields are preceded by a red dot.

Click  for field descriptions.

5. Click the Comments tab and type any comments you have about the selected Network profile into the Comments field.

6. Click Save on the tool bar to save the profile.

Network Commands profiles

A Network Commands profile are used by DataInterchange to construct the commands required by a network to the network; it is used to prepare the network commands file that DataInterchange sends to your network interface program.

You only need to create Network Commands profiles if you are setting up DataInterchange to connect to networks for which DataInterchange does not include Network profiles. DataInterchange is shipped with Network profiles and Network Commands profiles for IBM Global Services and the General Electric Information Services network.



ATTENTION: Each Network Commands profile is associated with a Network profile. To create Network profiles, see Chapter 13, “Network profiles,” on page 107. For further details in communicating with networks, see the *DataInterchange Programmer's Reference*.

About Network Commands

Every network requires certain commands to accomplish tasks like sending and receiving data. Those commands require information from DataInterchange, such as a User ID or Password. Network Commands profiles define the commands required by the network and maps the DataInterchange data they require to the network command file.

For example, a network may require a command called SENDX12 to send X12 files to the network. In order to send X12 data to that network, you must first define a Network profile for that network, and then define the Network Commands profiles that instruct DataInterchange how to create the SENDX12 command. The SENDX12 command may require that DataInterchange pass a user ID to the network. The Network Commands profiles for the SENDX12 command tells DataInterchange where to map the data in DataInterchange's User ID field to the network input file so that you can send X12 data to the network.

The Network Commands profiles map data within DataInterchange into commands fed to a network message program, which, in turn, communicates with a network interface program. The Network Commands profiles are used to prepare commands to pass to the network's communications routines by specifying where DataInterchange can locate the various bits of data required by the network.

Creating network commands sometimes requires making changes to the DataInterchange code itself. If you find that you cannot create all the network commands you need through the Network Commands profile, contact your DataInterchange technical support representative.



NOTE: One or more Network Commands profiles are used to construct each network command.

Setup overview

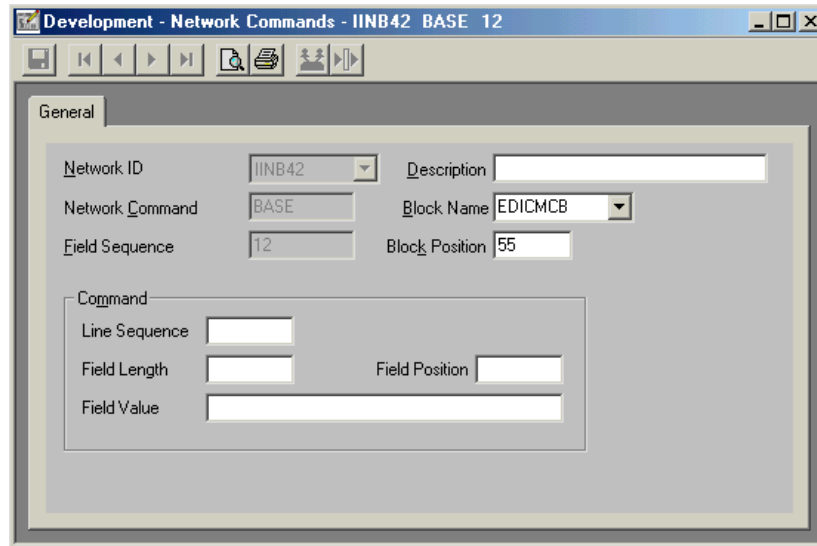
You set up and maintain Network Commands profiles through the Network Commands List window, which you access by clicking Setup on the DataInterchange Client Navigator bar. The Setup window, which contains DataInterchange Client's setup profiles, displays. Click the Network Commands tab, and the Network Commands List window displays.

	Network ID	Description	Net Cmd	Fld Seq	Blk Name	Blk Pos	Line S
1	IINB42		BASE	12	EDICMCB	55	
2	IINB42		CALLREC		EDINPDB	60	
3	IINB42		FSUPPORT	7	EDIREQDB	64	

This window displays a list of Network Commands profiles. Each row contains information about a Network Commands profile; each column contains data stored in the profile. Information in the columns displays in fields, drop-down lists, and check boxes in the Network Commands Editor window. The profile list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to “Creating a query” on page 287.

To view a profile or to add or change the information in these fields, double-click the row of the profile you wish to work with. The Network Commands Editor window displays.



The Network Commands Editor window contains a General tab. Use the General tab to enter and change information contained in Network Commands profiles.

Following are detailed procedures for creating new Network Commands profiles. For information on viewing, copying, editing, renaming, deleting, and printing profiles, see “Performing common file management tasks” on page 35. For information on exporting profiles, see “Exporting” on page 52.

Creating Network Commands profiles

The first step in creating Network Commands profiles is to determine what commands are required by the network with which you are going to communicate. You need to talk to a technical support representative to find out what the network requires for communications.

You need to obtain specifications for the network’s interface program so that you can write an interface to your network messaging application. You also need to obtain a list of the commands and command specifications that the network requires to perform specific tasks, such as sending mail to or receiving mail from a network mailbox.

Once you have that information, you set up a Network profile, as described in Chapter 13, “Network profiles,” on page 107. Then you set up one or more Network Commands profiles for each network command.

You create a new Network Commands profiles when you are setting up a new network that is not already supported by DataInterchange. You also need to set up a new Network Commands profile when you are adding a new command to an existing network.

◆ **To create a Network Commands profile:**

1. Click Setup on the DataInterchange Client Navigator bar.

The Setup window displays.


2. Click the Network Commands tab.

A list of the existing Network Commands profiles displays.

3. Click New on the tool bar or select New from the File menu.

The Network Commands Editor window displays with the General tab open and all the fields blank.

4. Complete the fields on the General tab.

Click  for field descriptions.

5. Select Save from the File menu to save the profile.

Network Security profiles

The Network Security profile allows DataInterchange to support security applications. You can protect data transfers by adding encryption and authentication processing through security applications shipped with DataInterchange or through third-party security applications.

DataInterchange allows you to set up filters that screen data for special characters. The system also supports file compression.



ATTENTION: A third-party or in-house security application entered into a Network Security profile must have a corresponding entry in a User Exit profile. (Security applications shipped with DataInterchange do not.) For information on User Exit profiles, see Chapter 16 on page 121.

About Network Security profiles

Network Security profiles allow you to add encryption, authentication, compression, and filtering applications to your DataInterchange system. The Network Security profile provides DataInterchange with the information it needs to pass messages to security applications for processing before sending data on the outbound side and before translating data on the inbound side.

Security services are often called “cryptographic services” and are defined below:

- **Encryption**
Refers to a process of scrambling a message before it is sent so that it cannot be read if intercepted by an unauthorized person during transmission. The receiver must have the appropriate key to decipher an encrypted message. Both the sender and receiver must use compatible encryption applications.
- **Authentication**
Refers to a process of verifying the identity of the sender and certifying the validity of the contents. When messages are authenticated, the receiver knows that the sender is who he says he is and that the contents of the message have not been corrupted during transmission. Both the sender and receiver must use compatible authentication applications.

■ Compression

Refers to a process of shortening the length of data. By compressing files, you shorten transmission times and reduce disk usage when storing files.

■ Filtering

Refers to a process of screening data before it is transmitted. You filter data to make sure that it does not contain characters that perform special functions on the network. If a network is sensitive to a particular character, you must run filtering when you use encryption services to make sure that the encrypted file does not contain the invalid character.

For more information on using security with DataInterchange, see the *DataInterchange Programmer's Reference*.



NOTE: DataInterchange provides an authentication program named IBMNSPA (load module EDITRAA). IBMNSPA does not require a corresponding User Programs profile.

DataInterchange provides an encryption interface program named IBMNSPE (load module EDITREE). IBMNSPE does not require a corresponding User Programs profile.

You can set up Network Security profiles to handle either encryption or authentication, or both. Filtering and compression work only in conjunction with encryption; you cannot set them up as separate processes. You can set up as many Network Security profiles as you need to handle security arrangements made with trading partners.



NOTE: To associate the same Network Security profile with many trading partners, type the appropriate Network Security profile ID in the Network Security Profile ID field of the Trading Partner profile.

Setup overview

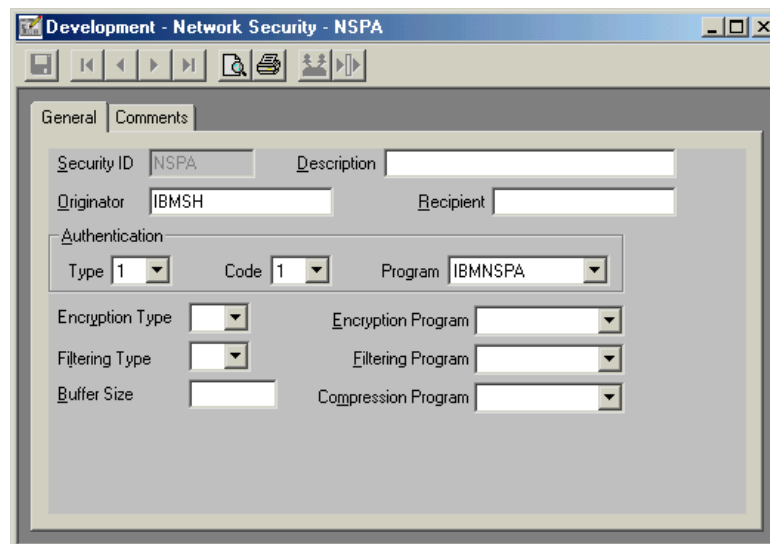
You set up and maintain Network Security profiles through the Network Security List window, which you access by clicking Setup on the DataInterchange Client Navigator bar. The Setup window, which contains DataInterchange Client's setup profiles, displays. Click the Network Security tab and the Network Security List window displays.

	Security ID	Description	Originator	Recipient	Auth Type	Auth Cod
1	IBMNSP					
2	NSPA		IBMSH		1	1

This window displays a list of existing Network Security profiles. Each row contains information about a Network Security profile; each column contains data stored in the profile. Information in the columns displays in fields, drop-down lists, and check boxes in the Network Security Editor window. The profile list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to “Creating a query” on page 287.

To view a profile or to add or change the information in these fields, double-click the row of the profile you wish to work with. The Network Security Editor window displays, with the General tab in front.



The Network Security Editor window contains two tabs: General and Comments. Use the General tab to enter and change information contained in Network Security profiles. Use the Comments tab to type any comments you wish about the selected Network Security profile.

Following are detailed procedures for creating new Network Security profiles. For information on viewing, copying, editing, renaming, deleting, and printing profiles, see “Performing common file management tasks” on page 35. For information on exporting profiles, see “Exporting” on page 52.

Creating Network Security profiles

The first step in creating Network Security profiles is determining whether you want to add security services to your messages. You must make this decision in conjunction with trading partners, as a trading partner must use compatible security software to send and receive encrypted and authenticated messages.

Once you have decided to use security services (also called cryptographic services), you need to decide which security applications to use. You may use the programs supplied with DataInterchange for both encryption and authentication or third-party security software.



ATTENTION: If you use an in-house or third-party security program, you must first set up a User Exit profile for the program. For details, see “User Exit profiles” on page 121.

For more information on using security with DataInterchange, see the *DataInterchange Programmer's Reference*.

You create a new Network Security profile when you want to use security services when exchanging files with a trading partner. You may need to set up different Network Security profiles to meet different trading partner requirements.

For example, say you have set up a Network Security profile that supports both encryption and authentication. If another trading partner that wants to use security services only wants to use encryption, then you need to create a new Network Security profile for that trading partner. You can use the same Network Security profile for trading partners that have the same security requirements.

◆ To create a Network Security profile:

1. Click Setup on the DataInterchange Client Navigator bar.

The Setup window displays.


2. Click the Network Security tab.

A list of the existing Network Security profiles displays.

3. Click New on the tool bar.

The Network Security Editor window displays with the General tab open and all the fields blank.

4. Complete the fields on the General tab. Required fields are preceded by a red dot.

Click  for field descriptions.

5. Click the Comments tab and type any comments you have about the selected Network Security profile into the Comments field.

6. Click Save on the tool bar to save the profile.

User Exit profiles

A User Exit profile provides DataInterchange with the information it needs to call other programs. For example, if you want to send and receive encrypted messages, you need to call a separate program to perform cryptographic services. DataInterchange identifies that security program through a User Exit profile.



NOTE: Some utilities supplied with DataInterchange do not require User Exit profiles. These are so indicated in the manuals.

About User Exit profiles

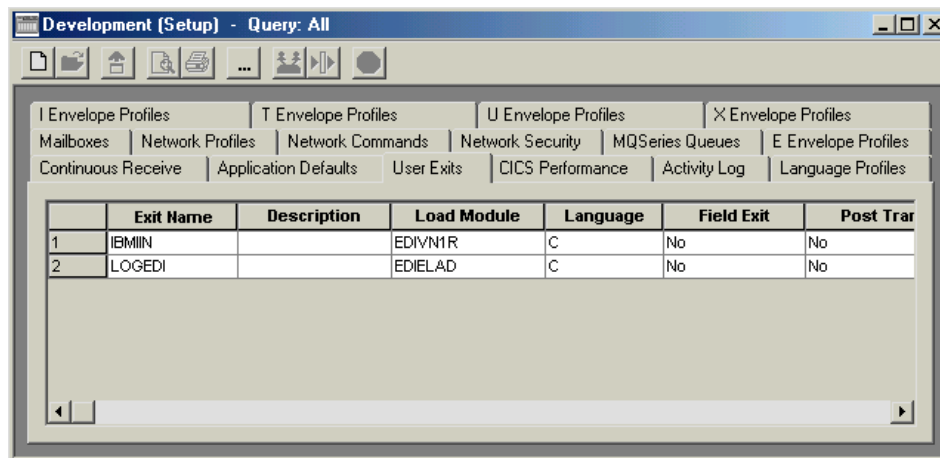
User Exit profiles contain the information that DataInterchange needs to identify separate programs that DataInterchange calls to provide supplemental processing. You need a User Exit profile for each external program or exit routine you use.

DataInterchange supports programs for:

- Network communications
- Security
- Data Mapping
- Network message handling
- Point-to-Point network communications

Setup overview

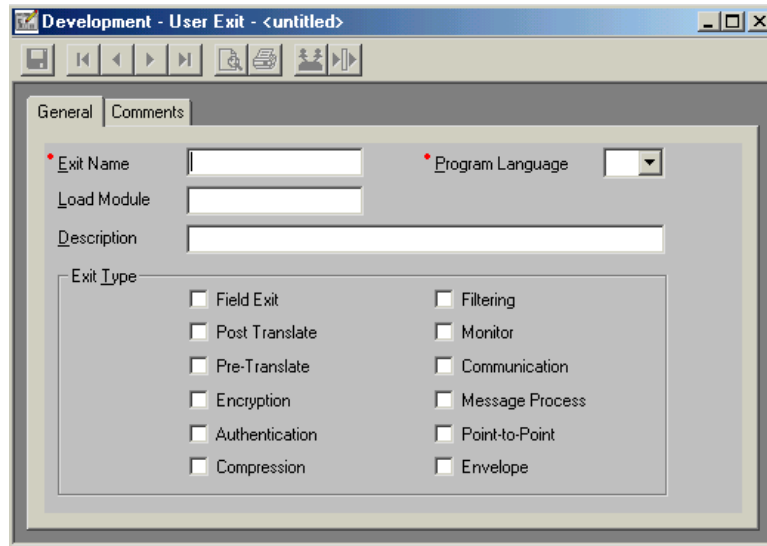
You set up and maintain User Exit profiles through the User Exit List window, which you access by clicking Setup on the DataInterchange Client Navigator bar. The Setup window, which contains DataInterchange Client's setup profiles, displays. Click the User Exits tab, and the User Exit List window displays.



This window displays a list of existing User Exit profiles. Each row contains information about a User Exit profile; each column contains data stored in the profile. Information in the columns displays in fields, drop-down lists, and check boxes in the User Exit Editor window. The profile list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to “Creating a query” on page 287.

To view a profile or to add or change the information in these fields, double-click the row of the profile you wish to work with. The User Exit Editor window displays, with the General tab in front.



The User Exit Editor window contains two tabs: General and Comments. Use the General tab to enter and change information contained in the User Exit profile. Use the Comments tab to type any comments you wish about the selected User Exit profile.

Following are detailed procedures for creating new User Exit profiles. For information on viewing, copying, editing, renaming, deleting, and printing profiles, see “Performing common file management tasks” on page 35. For information on exporting profiles, see “Exporting” on page 52.

Creating User Exit profiles

The first step in creating User Exit profiles is identifying the external program or utility that you need DataInterchange to call. Remember that some utilities shipped with DataInterchange do not require User Exit profiles.

You create a new User Exit profile whenever you start using another external program for supplemental processing.

◆ To create a User Exit profile:

1. Click Setup on the DataInterchange Client Navigator bar.


The Setup window displays.

2. Click the User Exits tab.

A list of the existing User Exit profiles displays.

3. Click New on the tool bar or select New from the File menu.

The User Exit Editor window displays with the General tab open and all the fields blank.

4. Complete the fields on the General tab. Required fields are preceded by a red dot.
Click  for field descriptions.
5. Click the Comments tab and type any comments you have about the selected User Exit profile into the Comments field.
6. Select Save from the File menu to save the new User Exit profile.

PART 3. Trading Partners

Trading Partners

Managing Trading Partner profiles is one of the most essential tasks. Whenever your organization begins exchanging documents with a new trading partner, you need to create a new Trading Partner profile. Trading Partner profiles identify your trading partners to DataInterchange and specify the details of how to exchange documents with each trading partner.

About Trading Partner profiles

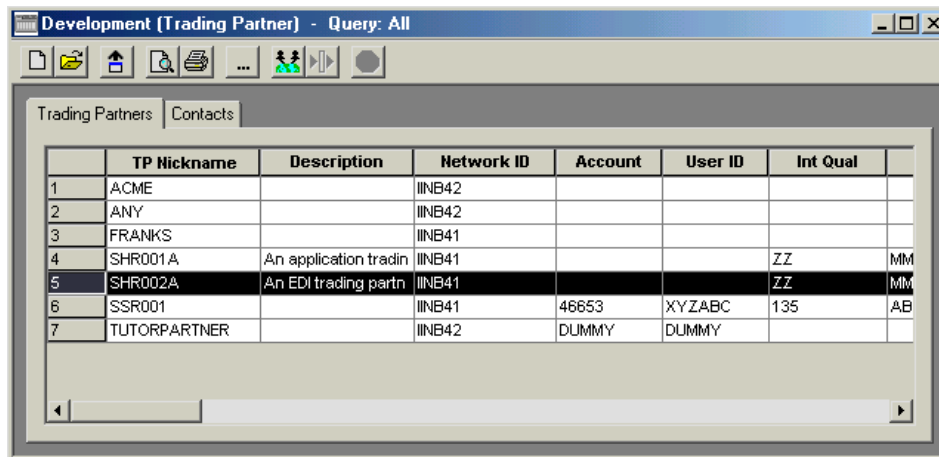
Trading Partner profiles allow DataInterchange to maintain information about your company's trading partners and to define the essential features of your connections. This section provides an overview of the purpose of Trading Partner profiles and how you set them up.

Purpose

DataInterchange Client maintains essential information on your trading partners through Trading Partner profiles. For example, the Trading Partner profile for a trading partner called JB Smith & Company would contain basic information such as company name, address, telephone number, and so on, as well as information used in translation, such as Trading Partner Nickname, Network ID, Account and User ID information, and Control Numbers.

Setup overview

You set up and maintain Trading Partner profiles through the Trading Partner List window, which you access by clicking Trading Partner on the DataInterchange Client Navigator bar.



	TP Nickname	Description	Network ID	Account	User ID	Int Qual	
1	ACME		IINB42				
2	ANY		IINB42				
3	FRANKS		IINB41				
4	SHR001A	An application tradin	IINB41			ZZ	MM
5	SHR002A	An EDI trading partn	IINB41			ZZ	MM
6	SSR001		IINB41	46653	XYZABC	135	AB
7	TUTORPARTNER		IINB42	DUMMY	DUMMY		

The Trading Partner List window contains two tabs, Trading Partners and Contacts. To work with a Trading Partner or Contact, click the appropriate tab to display a list of current Trading Partner or Contact profiles. Each tab displays a list of existing profiles. Each row contains information about a profile; each column contains data stored in the profile. Information in the columns displays in fields, drop-down lists, and check boxes in the Trading Partner or Contact Editor window. The profile list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to “Creating a query” on page 287.



NOTE: When you have used DataInterchange Client for a while, your list of Trading Partner profiles may grow to the thousands. To shorten the length of time it takes to display the list, create a query that filters the list so that it displays only the set of trading partners you wish to view, and only the columns you really need to see.

To view a profile or to add or change the information in these fields, double-click the row of the profile you wish to work with, or click the Trading Partner profile you want to work with and then click Open in the window's tool bar. The profile's editor window displays, with the General tab in front.

The Trading Partner Editor window contains seven tabs, each of which displays specific information about only the selected Trading Partner profile. You enter specific information by clicking the tabs and filling in the fields. Once you save any changes, DataInterchange Client displays this information in the Trading Partner profiles list.



NOTE: To customize the field tags on the User Fields tab page, see “Customizing field tags” on page 43.

To view the Contacts Editor, click the Contacts tab in the Trading Partner List window. Note that the Contacts Editor is not the same as the Contacts tab on the Trading Partner Editor window. The Contacts Editor allows you to compile and maintain a list of people and organizations you contact, as described on page 141. The Contacts tab on the Trading Partner Editor lists the contacts associated with the trading partner. The tab can be used to associate additional contacts with the trading partner or remove existing associations, as shown on page 130.

Following are detailed procedures for creating new Trading Partner and Contact profiles. For information on viewing, copying, editing, renaming, deleting, and printing profiles, see “Performing common file management tasks” on page 35. For information on exporting profiles, see “Exporting” on page 52.

Creating Trading Partner profiles

Define a trading partner with which you will exchange documents.

◆ To create Trading Partner profiles:

1. Click Trading Partner on the DataInterchange Client Navigator bar.


The Trading Partners list window displays. Click the Trading Partners tab if it is not already in front. This displays a listing of existing Trading Partner profiles.

2. Click New on the tool bar.

The Trading Partner Editor window displays, with the General tab in front.

3. Complete the TP Nickname field. This is the only required field on this screen, as indicated by the red dot. This field names the trading partner profile.

4. Enter information in the other fields on the General Tab as desired.


Click  for field descriptions.

5. Click Options. One of two dialog boxes displays, depending on the network you use. For more information on networks, see Chapter 5 of the *DataInterchange Administrator's Guide*.

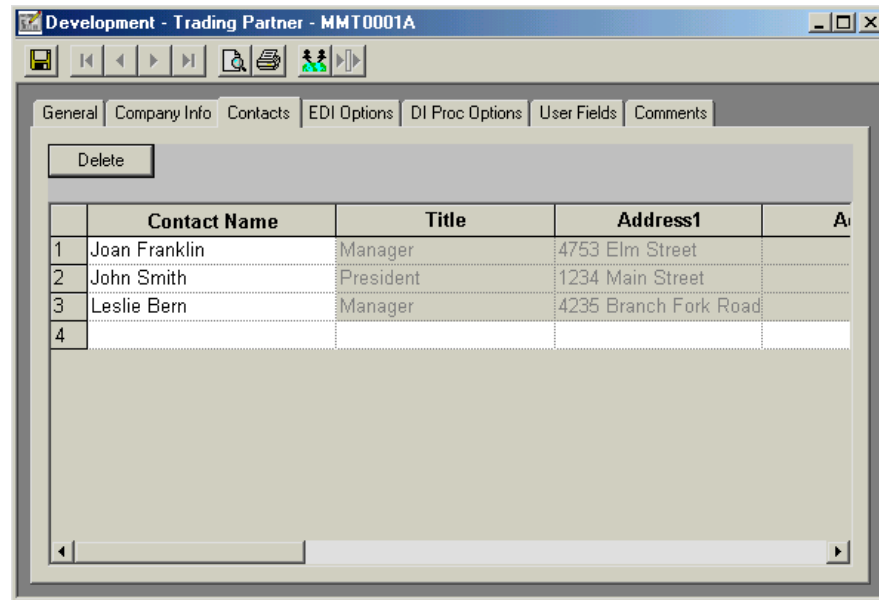


NOTE: The Options button only becomes available after you make a selection from the Network ID drop-down list. The IBM Global Services Addressing dialog box displays.

The network you use to communicate with a trading partner has been set up through a Network profile. A network displays in the list only after you create a Network profile for it. For details, see Chapter 13, "Network profiles," on page 107.

Click  for field descriptions.

6. Click Edit TP Control Numbers if you want to view, create, or edit the control numbers pair information. The Control Numbers List window displays with the following information: TP nickname, receiver ID, receiver qualifier, transaction ID, interchange control number, group control number, and transaction control number.
 - a. To view the information, click the left and right arrows on the scroll bar at the bottom of the Control Numbers List window.
 - b. To change information about control number pairs, double click the selection on the list window, and make any changes on the General tab page. Click Save.
 - c. To add control number pairs, click New. Fill in the required fields of Receiver Attributes and any optional fields you want to add, then click Save.
7. Enter information in the fields on the Company Info Tab, as desired.
8. Click the Contacts tab, and select any contacts you want to associate with this trading partner from the Contact Name drop-down list.




	Contact Name	Title	Address1	Address2
1	Joan Franklin	Manager	4753 Elm Street	
2	John Smith	President	1234 Main Street	
3	Leslie Bern	Manager	4235 Branch Fork Road	
4				

The column in the grid fills in with data from the Contact profile you selected. Your contacts must be added before you can select them within your trading partner profile.

To add a Contact to the list, see “Creating contacts” on page 141.

9. Click the Comments tab and type any comments you have about the selected Trading Partner into the Comments field.
10. The fields on the EDI Options tab, DI Proc Options tab, and the User Fields tab are optional.

Click  for field descriptions.



NOTE: User Fields tab fields can be customized by the user. You can store anything you consider relevant to the trading partner in the 10 fields on this tab.

11. Click Save on the tool bar.

DataInterchange Client saves the new Trading Partner profile.

Understanding processes and rules

For DataInterchange to transform a document from the source format to the target format, a map is needed. Maps tell DataInterchange how elements in the source document correspond to the elements in the target document. The overall process of determining which map to use to transform a given document is known as map resolution. The simplest form of map resolution is to always use the same set of routing and transformation instructions for a particular source document. This is common when exchanging documents between two internal applications, and is usually referred to as Application Integration or Enterprise Application Integration (EAI). However, this is typically not sufficient for Business to Business (B2B) requirements.

In a B2B scenario, routing and transformation requirements for a given type of source document also typically depend upon the sender and receiver of the document. For example, a purchase order that came from Company X's SAP system in IDOC format may need to be translated into cXML if it is destined for Company A because that is what Company A expects. If that same purchase order is instead destined for Company B, the format must be transformed into an EDIX12 850. In this B2B scenario, the type of transformation depends on both the document type (purchase order in SAP IDOC format) and the receiver (Company A or Company B).

Additionally, the destination is usually specified within the content of the document itself, not in a special header or separate document prepared specifically for DataInterchange. The content of the document must be parsed to determine its destination and the type of transformation. This process of examining the contents of a document to determine how to route and transform it is usually referred to as Content Based Routing and Transformation. It is one of the supported map resolution processes within DataInterchange, and is the most commonly used method for B2B applications. Routing and transformation instructions are usually referred to as rules, and within DataInterchange are also known as usages because they determine when to use a specific map.

To continue the previous example, in order to specify to DataInterchange how to properly transform the SAP IDOC purchase order for Company A and Company B, we must create some rules (or usages). The simplest way to do this within DataInterchange is to enter two rules: one that specifies if a SAP IDOC formatted purchase order is sent to Company A, then transform it to cXML, and one that specifies if a SAP IDOC formatted purchase order is sent to Company B, then transform it to an EDI X12 850. Since this is a very simple example, it is most appropriate to use a very simple set of rules. Many customers will have B2B environments that are much more complicated than this example of one document (the purchase order) and two trading partners (Company A and Company B). These customers will require a more complex set of rules to support their environment.

Now we can examine a hypothetical, but realistic example of a large scale B2B environment.

Company X has 10,000 trading partners instead of two, and instead of just one document they need to exchange 10 different types of documents (catalog update, purchasing forecast, purchase order, purchase order acknowledgment, purchase order change, purchase order change acknowledgment, advanced ship notice, receiving advice, invoice and remittance advice). Consider how many rules Company X would have to create to link those 10,000 trading partners to the 10 different documents. If there is a rule for each trading partner, for each document, then the formula is number of rules = number of documents x the number of trading partners, or $10 \times 10,000 = 100,000$. Entering and maintaining 100,000 rules can be an onerous task. This total can be dramatically reduced because many of those 10,000 trading partners exchange exactly the same documents and require the same transformations. Only the routing differs between them.

These 10,000 trading partners can be broadly classified into two categories: those that require their documents to be in cXML format and those that require them in EDI format. If we can create rules between the categories and the documents, rather than between the trading partners and the documents, then we only have 10 documents x 2 categories = 20 rules. Twenty is a far more manageable number than 100,000, so DataInterchange allows you to categorize trading partners by what is known as a process. DataInterchange also allows you to create a rule between a process (or category of trading partner) and a map, as well as between a trading partner and a map.

The term "process" refers to a collection of maps to be used to interact with a set of trading partners. Such a set of maps is commonly referred to as a public process because it is a sequence of public (external to your company) document exchanges which implement some business process. Examples of such a business process are production purchasing or health care claim processing. The sequence of steps that take place inside your company to implement the business process are known as a private process. Processes in DataInterchange can be either public or private processes.

Versioning is another important aspect of processes and is accomplished in DataInterchange via naming conventions. Suppose that after a number of years of using your cXML process, you want to update the process to take advantage of some new features in the latest versions of the cXML documents with a few of your old trading partners and the new trading partners you add. You do not need to exploit the new features in the latest versions of the cXML documents with the vast majority of your current cXML trading partners, so you do not want to alter the way you transact business with them. So, you create a new version of the cXML process named cXML-V2 and switch the specific current customers that you need on the new version to the cXML-V2 process and add all new trading partners to the new version, as well.

Finally, suppose that some particular trading partner has a special requirement for a single document; for example, they cannot use the EDI 850 the way you normally send it. You create a map that transforms your SAP IDOC purchase order into an EDI 850 just the way they require it. Is this a new version of your purchasing process? Not really, it is just a single trading partner that had a specific requirement on a single document. Consequently, instead of creating a new version of your EDI process, you can override the map used when sending a purchase order to only that trading partner by creating a rule that links the trading partner to the specialized map.



NOTE: Trading partner rules, including generic rules, are considered to be of higher precedence than process rules. Therefore, process rules will only be used if no trading partner rules apply.

Understanding minimal trading partners

Though processes address the problem of minimizing the number of rules required to correctly transform a given set of documents to be exchanged with a given set of trading partners, it does not address the issue of reducing the number of trading partner profiles required. For example, in the previous example Company X has 10,000 trading partners. Implementing processes reduced the required number of rules from 100,000 to 20, but did not reduce the number of trading partners in the trading partner profile. There were still 10,000 trading partners to be entered and maintained.

The most fundamental purpose of the trading partner profile is to identify the network address that a message to a trading partner should be routed to, or the network address that appears as the sender of a message from that trading partner. If the DataInterchange user is willing to provide

the network addresses of the sender and receiver for each outbound document passed into DataInterchange for processing, and accept network addresses as the identifiers for the sender and receiver of inbound documents, then the trading partner profile is not required. So, using the Minimal Trading Partners feature of DataInterchange, a user with many trading partners can significantly reduce the number of trading partner profiles, and consequently, rules needed for a given set of trading partners and maps.

When implementing Minimal Trading Partners, the network addresses are stored in your applications instead of DataInterchange; thus, you may need to modify the application before you begin using Minimal Trading Partners. The network address has two parts referred to in the Trading Partner Profile as the interchange ID and qualifier. The user's application must pass in the values to be used for the interchange ID and qualifier on the C record if C and D records are being used, or in predefined elements within the document to be transformed.

If you are using the Minimal Trading Partners feature, you can still create profiles for trading partners and use them as you normally would. Indeed, this is part of the architecture of the Minimal Trading Partners feature because it is how you override the standard behavior for trading partners that have special requirements or need special processing that cannot be handled by the Minimal Trading Partner feature. The Minimal Trading Partners feature and the trading partner profile are, therefore, complimentary rather than mutually exclusive.

Minimal trading partners scenario

Consider once again Company X, which has 10,000 trading partners. Company X must create its own common trading partner profile to specify the delimiters to be used when sending to these trading partners. This profile is the default trading partner profile, including the delimiters, to be used when Company X sends a message or transaction to a trading partner not defined in the profile. This trading partner profile will also be the default trading partner profile used when Company X receives a message or transaction from a trading partner not defined in the profile. So the absolute minimum number of trading partner profiles that could ever be defined is one. That one profile, the default trading partner profile, would be used for all trading partners.

The name of the default trading partner profile is ANY because it can be used for any trading partner that does not have a trading partner profile of its own. The ANY trading partner profile is shipped with DataInterchange and does not have to be created, although it should be updated to reflect the preferences in your environment.

If one of Company X's trading partners (Company A) requires a set of delimiters different than the other 9,999 trading partners, Company X must create a trading partner profile for Company A specifying the unique delimiters to be used. Company X does not have to create and maintain trading partner profiles for the other 9,999 trading partners because they are willing to accept the delimiters defined in the default Trading Partner Profile. When Company X sends to Company A, the profile created for Company A will override the set of delimiters in the default Trading Partner Profile, which is used for the other 9,999 trading partners.

With the possibility that some trading partners will appear in the Trading Partner Profile and some will not, it is necessary to classify trading partners with respect to whether they appear in the Trading Partner Profile. In DataInterchange, trading partners are considered to be either KNOWN or UNKNOWN. A known trading partner is one that has a trading partner profile. Referring again to the scenario, when Company A was included in the trading partner profile, it became known to DataInterchange. The other 9,999 trading partners remain unknown to

DataInterchange. The set of all trading partners, whether known or unknown, is referred to in DataInterchange as ANY trading partner. These classifications become keywords in DataInterchange that can be used to refer to these classes of trading partners in messages, rules, and usages.

Understanding generic rules (usages)

An alternative to using processes to reduce the number of rules you must create and maintain is to use generic rules (or usages). Generic rules are particularly useful for reducing the number of rules when you are using the Minimal Trading Partners feature since you cannot use processes with it. Generic rules allow you to use the keywords ANY and KNOWN in place of a trading partner name on a trading partner-based rule. These rules have the disadvantage in that you cannot create arbitrary classes of trading partners like NON-PROD-PROCESS-V1 for the trading partners associated with the first version of your non-production purchasing process. Trading partners are simply either KNOWN or UNKNOWN, and rules can be created that apply to either specific trading partners, all KNOWN trading partners (the ones in the Trading Partner Profile), or ANY trading partner (all trading partners, both KNOWN and UNKNOWN). The reason you cannot use processes with the Minimal Trading Partners feature is because when an unknown trading partner sends you a message, there is no way for DataInterchange to know what process that trading partner is associated with because DataInterchange knows nothing about the trading partner.

Trading partner-based rules relate trading partners to maps. At the most fundamental level, a rule is the combination of a map name, a sending trading partner name, and a receiving trading partner name. In addition to the fundamentals, rules allow you to specify a set of translation options that apply to the relationship. An example of a translation option is Acceptable Error Level, which affects whether the translator accepts or rejects a given message.

To continue the scenario of Company X and its 10,000 trading partners, consider how many usages you must create to link those trading partners to a single map. Since you know that Company A required a set of delimiters different than the other 9,999 trading partners, you must include Company B and its unique delimiters in the trading partner profile. So far, because there is no difference in the translation options to be used for any of the 10,000 trading partners, only one usage would be required for all. In English documentation, the usage relationship required would read: When Company X sends this type of data to ANY trading partner (either the KNOWN trading partner Company A in the trading partner profile or the 9,999 unknown trading partners not in the trading partner profile), use this map to translate the data. You would code this in the send usage by specifying ANY as the sending trading partner and the keyword ANY for the receiving trading partner. This type of usage is known in DataInterchange as a generic-generic rule because both trading partners are generic (the keyword ANY).

Suppose yet another company (Company B) requires an Acceptable Error Level that is different than the other 9,999 trading partners. To be able to refer to Company B, you must add Company B to the trading partner profile even though all their trading partner profile options are from the default options. Finally, you must add a usage and specify the Acceptable Error Level to be used when Company X sends to Company B. To code this in the send usage, specify ANY as the sending trading partner and Company B as the receiving trading partner. This type of usage is known in DataInterchange as a generic-specific usage, because the sending trading partner is generic (the keyword ANY) while the receiving trading partner (Company B) is specific. You now have two usages that specify two different sets of translation options that could be used when Company A sends to Company C: the anybody-to-anybody usage and the anybody-to-Company C usage. To determine which one to use, the translator arranges all candidate usages into a hierarchy and uses the one that most applies; in this case, anybody to Company C.

Translator hierarchy

The hierarchy used by the translator has four general classes of combinations, listed in order of precedence.

- Specific sending trading partner, specific receiving trading partner (specific-specific)
DataInterchange translates this as meaning: Use this map when this sender trades this document with this receiver.
- Generic sending trading partner, specific receiving trading partner (generic-specific)
DataInterchange translates this as meaning: Use this map whenever any known or unknown sending trading partner trades this document with this specific receiving trading partner.
- Specific sending trading partner, generic receiving trading partner (specific-generic)
DataInterchange translates this as meaning: Use this map whenever this sending trading partner trades this document with any known or unknown trading partner.



NOTE: The external trading partner always takes precedence over the internal trading partner, so which of the two that takes precedence depends on whether the external trading partner (Company A in the previous examples) is the sender or the receiver of the document.

- Generic sending trading partner, generic receiving trading partner (generic-generic)
DataInterchange translates this as meaning: Use this map whenever any known or unknown trading partner trades this document with any other known or unknown trading partner.

Specifying usages and rules

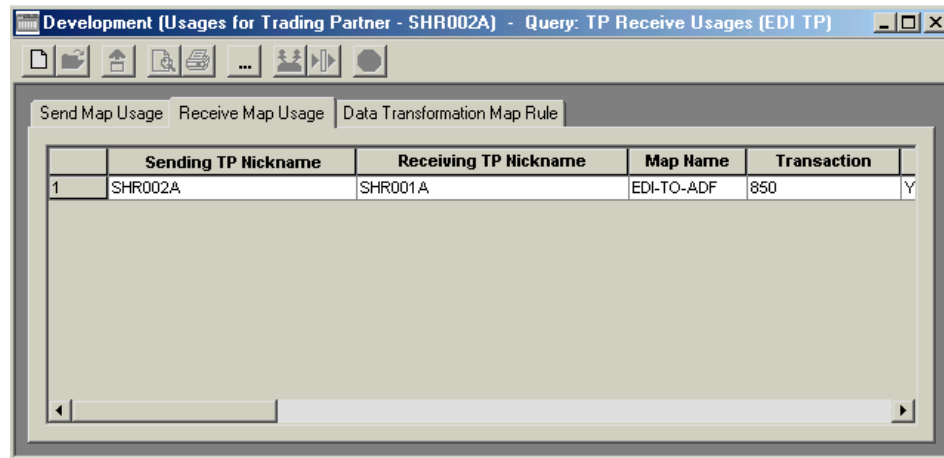
You can view and create send usages, receive usages, and data transformation map rules through the Trading Partner List window.

Viewing usages and rules

◆ To view a usage or rule:

1. In the Trading Partner List window, click the trading partner for which you want to view usages or rules.
2. Click View Usages on the tool bar.

DataInterchange Client runs a query that displays the Usages and Rules List window, which contains three tabs. Click the Send Map Usage tab to display a list of Send Usages associated with the trading partner. Click the Receive Map Usage tab to display a list of Receive Usages associated with the trading partner. Click the Data Transformation Map Rule tab to display rules associated with trading partner.



Creating usages and rules

Create a usage or rule when you need to create a new association between a map and a trading partner.

◆ To create a new usage:



1. In the Trading Partner List window, click the trading partner for which you want to create a usage or rule.
2. Click View Usages on the tool bar.

The Usages and Rules List window displays.

Send map usage

- a. Click the Send Map Usage tab.
- b. Click New on the toolbar.



The Send Map Usage Editor window displays with the General tab open and all the fields blank.

- c. Complete the fields on the General tab. Required fields are preceded by a red dot.
Click  for field descriptions.
- d. The fields on the Exit Routines tab, Envelope Attributes tab, and the DI Options tab are optional.
Click  for field descriptions.

Receive map usage

- a. Click the Receive Map Usage tab.
- b. Click New on the toolbar.

The Receive Map Usage Editor window displays with the General tab open and all the fields blank.



- c. Complete the fields on the General tab. Required fields are preceded by a red dot.
Click  for field descriptions.
- d. The fields on the Attributes tab and the DI Options tab are optional.
Click  for field descriptions.

Data Transformation map rule

- a. Click the Data Transformation map rule tab.
- b. Click New on the toolbar.

The Data Transformation Map Rule Editor window displays with the General tab open and all the fields blank.

The screenshot shows the 'Development - Data Transformation Map Rule - <untitled>' window. The 'General' tab is selected, showing fields for 'Map Name' (required), 'Dictionary Name', 'Document Name', and 'Description'. The 'Usage Indicator' section has radio buttons for 'Test' (selected), 'Production', and 'Information'. The 'Associated With' section has radio buttons for 'Process' (selected) and 'Trading Partners', with a 'Process ID' field. Below are 'Sending' and 'Receiving' dropdown menus. The 'Properties' section has checkboxes for 'Acknowledgment Expected', 'Log Inbound Application Data', 'Log Outbound Application Data', 'Activate', and 'Group level FA only'. The 'Output File' section has 'Name' and 'Type' fields, and an 'Acknowledgment Type' dropdown.

- c. Complete the fields on the General tab. Required fields are preceded by a red dot.
Click  for field descriptions.
 - d. The fields on the Exit Routines tab, Envelope Attributes tab, and the DI Options tab are optional.
Click  for field descriptions.
3. When you have finished entering all values required in the usage or rule, click Save on the tool bar to save the usage or rule.

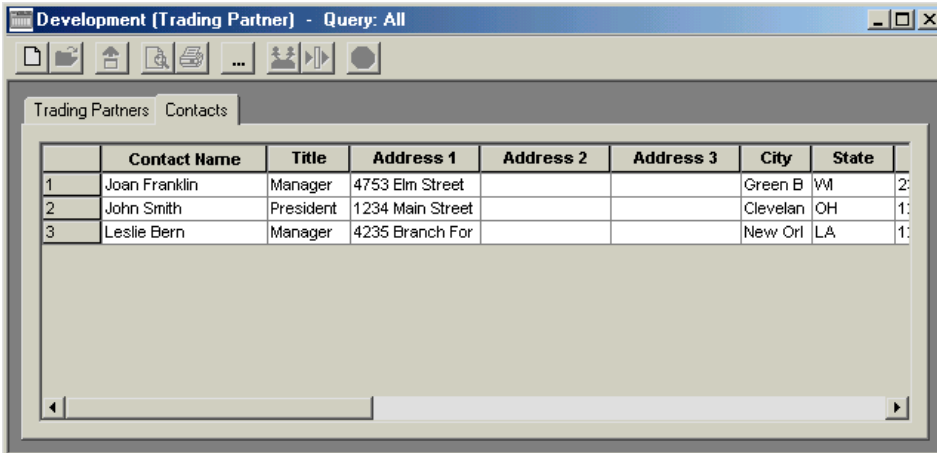
Creating contacts

DataInterchange Client uses Contacts to maintain information on the various contacts for your trading partners. For example, a contact can be the person who manages the business for a trading partner, an analyst at the trading partner's company, or a third-party administrator that manages operations on behalf of a trading partner.

Although it is useful to associate contacts with particular trading partners, it is not necessary. If you like, you may use the Contacts tab as your DataInterchange address book to store the names and addresses of all of your DataInterchange contacts, regardless of their association with trading partners.

Note that contacts have no role in DataInterchange's translation functions. They are provided in DataInterchange Client as a convenience.

You set up and maintain information on your trading partner contacts in the same way you set up and maintain information on trading partners. To view a Contact, double-click the appropriate row in the Contacts tab to display the Contact Editor window.



	Contact Name	Title	Address 1	Address 2	Address 3	City	State	
1	Joan Franklin	Manager	4753 Elm Street			Green B	WI	2
2	John Smith	President	1234 Main Street			Cleveland	OH	1
3	Leslie Bern	Manager	4235 Branch For			New Ori	LA	1

The Contact Editor window contains two tabs, each of which displays information only about the contact selected from the contacts list. You enter information about contacts by clicking the tabs and filling in the fields. DataInterchange Client places the information in the Contact list. For information on viewing, copying, editing, renaming, deleting, and printing Contacts, see "Performing common file management tasks" on page 35. For information on exporting Contacts, see "Exporting" on page 52.



NOTE: The Contact List window displays all contacts that satisfy your query. The Contacts tab within the Trading Partner Editor window displays only those contacts associated with that particular trading partner.

Adding a contact

◆ To add a contact:

1. Click Trading Partner on the DataInterchange Client Navigator bar.
The Trading Partners List window displays.
2. Click the Contacts tab if it is not already in front.
The Contacts List window displays, displaying a listing of existing contacts.
3. Click New on the tool bar.
The Contacts Editor window displays, with the General tab displayed.

The screenshot shows the 'Contacts Editor' window with the 'General' tab selected. The form contains the following fields:

- Name (required, indicated by a red asterisk)
- Title
- E-Mail Address
- Mailing Address section:
 - Address 1
 - Address 2
 - Address 3
 - City
 - State
 - Postal Code
 - Country
- Phone Numbers section:
 - Phone
 - FAX
 - Other

4. Type in a Name, the only required field, and any other information you want to maintain on the contact.
5. Click the Comments tab, and type any comments you have about the selected contact into the Comments field.
6. Click Save on the tool bar.

DataInterchange Client saves the new contact.

PART 4. Mapping

Data formats

The term “data format” defines the layout of your application data. It is a document definition. The word “data”, of course, refers to the information itself. The word “format” refers to the physical layout of information in the file, such as field names and lengths.

DataInterchange needs a description of the data format for each business application that generates data that will be translated, or uses data that has been translated. An application’s data must be described to DataInterchange so that it can be used as a source or target for translation.

DataInterchange Client’s Data Format editor allows you to describe your application’s data to DataInterchange. Once you have created a data format, you then create a map between the application’s data format and the source or target document. Creating a data format, then, is the first step in the mapping process if your map uses or creates data to be used by your application.

You usually need to create a data format for each different business document that is used or created by DataInterchange. A single data format can be mapped to multiple documents.

If you use EDI or XML to exchange invoices, for example, you need to create a data format for your invoicing system so that DataInterchange understands how your invoicing system structures an invoice. From that data format, you create a map so you can translate your application’s data to the 810 transaction in the X12 standard, or the XML format that your trading partner uses. For details on mapping, see Chapter 21, “Creating a map,” on page 207.

This chapter assumes that you know the layout of the application data you are using.

Creating a data format

This section provides the basic steps you follow to create a data format for an application. The broad steps are:

1. Understand how your application data is structured.

Get a copy of your application's record layout and study its format.

2. Fill out a data format work sheet.

This form helps you to enter information on your data format into DataInterchange Client's data format editors. Refer to the sample on page 151 and the worksheet example on page 152.

3. Use the DataInterchange Client data format editors to create the data format.

DataInterchange Client has data format editors for Data Format Dictionaries, Record ID Info, Data Formats, Loops, Records, Structures, and Fields.

Each step is detailed in the sections that follow.

Understanding how your application data is structured

Understanding how your application data is structured requires three steps:

1. Obtain a useful copy of your application data layout.
2. Optionally, structure that data so that DataInterchange Client can use it.
3. Determine the data components used by your application.

Obtain application data layout

First, you need to obtain a copy of each application's record layout. The record layout can come directly from the program code listings or any other documentation that shows the beginning and ending position of each field in the record. For each record, the information should show:

- Physical attributes of each field
- Content of each field
- Position of each field within the record
- Length of each field
- Data type of each field
- Relationship between the records

Many users will be able to obtain COBOL copybooks that specify that information for their applications.

Structure application data

The next step in understanding your application data is structuring application data in a format that DataInterchange Client will accept. DataInterchange can accept two types of data record formats: *raw data records* and *control and data records*, as described below.

■ Raw data

Each record in raw data format identifies itself by containing a unique record identifier (a record ID). Raw data can be either comma-separated or fixed-position. For fixed-position data, the identifier starts in the same position and extends for the same length in each record. In comma-separated data, each value is separated by a comma. Comma-separated data is used only with data transformation maps. Raw data that is fixed-position can be used in data transformation maps, send maps and receive maps. The Record ID is actually a field in the record. As long as records contain identifiable record IDs, you can use application data without modification.

In the illustration below, which is an example of fixed-position data, several records of application data contain the record ID in the first three positions. In this example, HDR is the record ID of the header record, NAM is the name, and so on.

```
HDR0123456 092091C321
NAMSmithson, Patricia Jeanne
SSN5555555555
PRVCity Regional Clinic
PID05050505-X505
ADR555 Cedar Road
ADRAny City IL
ADR61001-1101
TOT1555.00
```

Figure 4. Application data with record identifier

When your data has no record identifier for DataInterchange to associate with each record, you have two choices. You can modify your application data to contain a record identifier or you can modify it to use control (C) and data (D) records.

■ Control and Data (C and D) Format

If no record ID clearly specifies the type of information contained in a record, that record structure can be indicated by using control and data records. Also use C and D records when you need to use multiple data formats in a single file or you need to use overrides offered in the C record that are not offered in raw data. You also can use this type of format to override fields within service segments (such as ISA, GS, UNB, and UNH).

C and D format is only supported by send and receive maps.

There are a number of ways to structure C and D records. Figure 5 on page 148, for example, shows an example of a C and D record in which:

- A C or D is in the first column (byte)
- Record name is in the next 16 columns
- Application data starts in column 18

This example also shows the use of a Z record to indicate the end of the document. For more about Z records, see the *DataInterchange Programmer's Reference*.

1	2	3	4	5	6
123456789012345678901234567890123456789012345678901234567890					
CSPSTT16	AFTSU09	IL			
DPOHDR	P0123456	092091C321			
DP0NOTE	INCOMPLETE INVOICE INFORMATION SLOWS REIMBURSEMENT				
DP0NOTE	PATIENTS WITH MULTIPLE CLAIMS NEED COMPLETE HISTORY				
DINVITEM	005500550055				
DITEMDESC	SURGICAL PROCEDURE				
DITEMDIAGN	CARPAL TUNNEL PAIN				
DINVITEM	005500550055				
DITEMDESC	SURGICAL PROCEDURE				
DITEMDIAGN	LIGAMENT INFLAMMATION				
DNAME	SMITHSON, PATRICIA JEANNE				
Z					
CSPSTT16	AFTSU09	IL			

Figure 5. Sample Application Records with C and D Records



NOTE: For more information on converting data to C and D records, see “Reformatting Data into Control (C) and Data (D) Records” in Chapter 7 of the *DataInterchange Administrator's Guide*.

Determine application data components

You must determine the data components used by your application so that DataInterchange can identify the data and accurately pass it to the translator. A “data component” is a grouping of related data fields, such as the field names that make up line items of a health-care claim or the ship-to address of a purchase order. DataInterchange needs information on:

- Which fields in your application data form the various components DataInterchange uses
- The order in which components occur
- The number of times each component occurs

That information allows DataInterchange to identify the data you pass to the translator from your application. When you use the data format editors to create a data format for an application's data, you show relationships between the various components and show the number of times each component can occur in a transaction.

DataInterchange Client uses four components to express the characteristics of the application's data for mapping:

- Fields
- Structures
- Records
- Loops

Each of these components is described below.

Fields. Fields are fundamental pieces of data, such as prices or item numbers or first names. Raw data that is fixed position can be used in data transformation maps, send maps and receive maps. In COBOL records, they are stored in a single variable.

Structures. A structure is a group of related data fields, which is probably unique to your company. When multiple fields always display together, you can designate the group as a structure and give it a structure name. Structures not only contain fields, but can also contain other structures.

For example, a purchase order line item contains price, quantity, and product ID fields. Those three fields always display together. Further, they are used not only on purchase orders but also on invoices. So you could create a structure called a line-item structure that consists of price, quantity, and product ID fields. A purchase order line item, then, may consist of a line item structure plus a requested ship date field, while an invoice line item may consist of a line item structure plus a requested payment date field. You may even repeat that structure within a purchase order line item record.



NOTE: Structures cannot be used for comma-separated data.

Records. A record is a set of related fields as they are defined in an application's data. Every record in your application must be defined in your data format, assuming you want to map all records. Records contain fields and structures. An example of a record that identifies a patient in a hospital's claims-management system is described in the following example.

You are a health care provider and you want to send your claim information in a single health-care claims EDI standard transaction to the insurer or payer of the claim. You submit claims each day for all patients with the same insurer or payer. You would look at your application's data and determine which records are required to send the data for a patient claim, say a patient record followed by claim line-item records. The patient is identified as R20, and the claim information is identified as R42 and R73, as shown in Figure 6.

R20	01623792001ALBERT	MICKEY	XF07261925S31PO BOX 11
R42	0101623792001	CO19200002510{0005{21000000150{0001{155000002500000A	
R42	0201623792001	PR1 00006100{0000{ 00000000{0000{ 000000000000	
R73	01623792001	48521PATIENT IS RECOVERING FROM SURGERY AS NEEDED	
R20	01623792001	LOUISE	MOONIE XF07261925S31PO BOX 11
R42	0302623792001	OA1 00006100{0002{8900000100{0206{115000005000000E	
R73	01623792001	48521PATIENT SHOULD RETURN IN 2 MONTHS FOR LAB RESULT	

Figure 6. Sample Record

In this example, there are multiple patient records and each patient record contains that patient's claim information. Because this group of records is related and repeats, you would create a loop consisting of the patient record and the repeating claim line-item records. Because loops can repeat, and the health care claims EDI standard transaction defines a repeating patient claim loop, you could send claims for multiple patients in a single health care claims EDI standard transaction to a single insurer or payer.

Loops. A loop is a group of records that repeat. Loops can occur within other loops; this is referred to as a nested loop. Loops are used for repeating such things as line items, as in the following example:

You want to include claims for multiple patients in a single health-care claims transaction. You would look at your application's data and determine which records are required to send the data for a patient claim, say a patient record followed by claim line-item records.

To pay multiple patients in the same transaction you would create a loop consisting of a patient record and a repeating claim line-item record. Because loops can repeat, you could send claims for multiple patients by repeating the patient claim loop for every patient.

When deciding how best to structure your application's data into data components, keep the following component hierarchy in mind:

- Loops can contain other loops or records.
- Records can contain structures or fields.
- Structures can contain other structures or fields.
- Fields are fundamental units of data.

The example of a data format worksheet in the following section will help you to determine how to structure your application's data into data format components.

Filling out a data format worksheet

A data format worksheet can make it easier to decide how to structure application data. This example shows a purchase order record definition and how its data format worksheet looks.

01	SHIPTO		
05	RECORD-IDENTIFIER		
10	PURCHASE-ORDER-NUMBER	PIC X(08).	
10	RECORD-ID	PIC X(02).	
05	COMPANY-NAME	PIC X(30).	
05	COMPANY-DUNS	PIC X(10).	
05	COMPANY-ADDRESS		
10	STREET	PIC X(30).	
10	CITY	PIC X(15).	
10	STATE	PIC X(02).	
10	ZIPCODE	PIC 9(09).	
05	COMPANY-PHONE	PIC 9(10).	
01	DETAIL		
05	RECORD-IDENTIFIER		
10	PURCHASE-ORDER-NUMBER	PIC X(08).	
10	RECORD-ID	PIC X(02).	
05	ITEM-NUMBER	PIC X(10).	
05	ORDER-QUANTITY	PIC 9(09).	
05	ORDER-UNITS	PIC X(01).	
05	UNIT-PRICE	PIC 9(09)V99.	
05	UNIT-DISCOUNT	PIC 9(09)V99.	
05	EXTENSION	PIC 9(09)V99.	
*THERE ARE UP TO 3 DETAIL DESCRIPTION RECORDS PER DETAIL RECORD.			
01	DETAIL-DESCRIPTION.		
05	RECORD-IDENTIFIER		
10	PURCHASE-ORDER-NUMBER	PIC X(08).	
10	RECORD-ID	PIC X(02).	
10	DESCRIPTION	PIC X(30).	
01	TOTAL		
05	RECORD-IDENTIFIER		
10	PURCHASE-ORDER-NUMBER	PIC X(08).	
10	RECORD-ID	PIC X(02).	
05	TOTAL-AMOUNT	PIC 9(09)V99.	

Figure 7. Sample Working Storage Record Definition of a Purchase Order

Table 60. Data Format Component Relationship Worksheet Example

Field Length (FIELD Only)							
Field Data Type (FIELD Only)							
Record ID (REC Only)							
Max Use or Occurs							
Parent Type	Parent Name	Child Type	Child Name				
data format	SAMPLE-PO	REC	SHIP-TO	1	SH		
		LOOP	DETAIL-LOOP	1000			
		REC	TOTAL	1	TT		
REC	SHIP-TO	STRUCT	RECORD-IDENTIFIER	1			
		FIELD	COMPANY-NAME			CH	30
		FIELD	COMPANY-DUNS			CH	10
		STRUCT	COMPANY-ADDRESS	1			
		FIELD	COMPANY-PHONE			NO	10
STRUCT	RECORD-IDENTIFIER	FIELD	PURCHASE-ORDER-NUMBER			CH	8
		FIELD	RECORD-ID			CH	2
STRUCT	COMPANY-ADDRESS	FIELD	STREET			CH	30
		FIELD	CITY			CH	15
		FIELD	STATE			CH	2
		FIELD	ZIPCODE			NO	9
LOOP	DETAIL-LOOP	REC	DETAIL	1	DE		
		REC	DETAIL-DESCRIPTION	3	DD		
REC	DETAIL	STRUCT	RECORD-IDENTIFIER	1			
		FIELD	ITEM-NUMBER			CH	10
		FIELD	ORDER-QUANTITY			NO	9
		FIELD	ORDER-UNITS			CH	1
		FIELD	UNIT-PRICE			N2	11
		FIELD	UNIT-DISCOUNT			N2	11
		FIELD	EXTENTION			N2	11
REC	DETAIL-DESCRIPTION	STRUCT	RECORD-IDENTIFIER	1			
		FIELD	DESCRIPTION			CH	30
REC	TOTAL	STRUCT	RECORD-IDENTIFIER	1			
		FIELD	TOTAL-AMOUNT			N2	11

Table 60. Data Format Component Relationship Worksheet

[illegible]

Use the DataInterchange Client data format component editors

Once you have filled out the data format worksheet, you are ready to create the data format on DataInterchange Client. To create the data format and its components, use the Data Format Editors, as described in the following section.

Using the data format editors

You use the data format editors to create the various components that make up a data format. In using the editors to create a data format, first decide whether to work from the top down or from the bottom up. If you work from the top down, after you create a Data Format Dictionary, you create the larger components first, then work down to the smaller. If you work from the bottom up, after you create a Data Format Dictionary, you create smaller components first, and then work up to larger.

Either way you choose to work, you must create a Data Format Dictionary as your first step. You cannot create components for a nonexistent dictionary and store them elsewhere until the dictionary is created. Once you create the dictionary, you can then create the smaller components such as structures and fields before creating the larger components. In practice, it is easier to create the larger components, such as loops and records first, and then work down to the smaller components until your data format is complete.

This section accomplishes two goals. First, it provides a generic description of the procedures for using the Data Format Editors. At the same time, it steps you through the process of using the editors to create a new data format. This section takes a top-down approach to creating a data format.

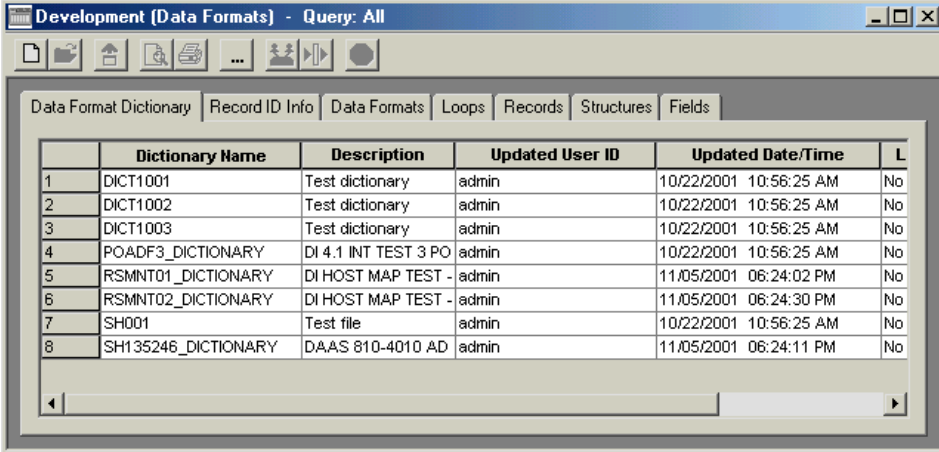
Accessing data format editors

You use the Data Format List window to gain access to the data format component editors. Each tab contains a list of components. Click the tab corresponding to the component you wish to work with. From that list, you can select the specific component you wish to work with and open its editor window by clicking the component and then clicking Open.

◆ To access a data format editor:

1. Click Data Format on the DataInterchange Client Navigator bar.

The Data Format List window, which contains tabs for the data format components, displays.



	Dictionary Name	Description	Updated User ID	Updated Date/Time	L
1	DICT1001	Test dictionary	admin	10/22/2001 10:56:25 AM	No
2	DICT1002	Test dictionary	admin	10/22/2001 10:56:25 AM	No
3	DICT1003	Test dictionary	admin	10/22/2001 10:56:25 AM	No
4	POADF3_DICTONARY	DI 4.1 INT TEST 3 PO	admin	10/22/2001 10:56:25 AM	No
5	RSMNT01_DICTONARY	DI HOST MAP TEST -	admin	11/05/2001 06:24:02 PM	No
6	RSMNT02_DICTONARY	DI HOST MAP TEST -	admin	11/05/2001 06:24:30 PM	No
7	SH001	Test file	admin	10/22/2001 10:56:25 AM	No
8	SH135246_DICTONARY	DAAS 810-4010 AD	admin	11/05/2001 06:24:11 PM	No

- Click the tab of the data format component you wish to work with.

The list window for that component displays.

This window displays a list of existing components of the type you selected. Each row contains a single component; each column contains data stored in the component. Information in the columns displays in fields in the respective editor windows. The list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click *Modify Window Properties (...)*. To create new queries, refer to “Creating a query” on page 287.

- To view an existing component or to add or change its information, double-click the row of the item you wish to work with.

The editor window displays, with the General tab or details tab in front, depending on which component type you opened. You add information or make changes to the data format component through its tabs, as described in the following sections.



NOTE: All of the seven component editors have a General tab and a Comments tab. The Data Format, Loops, Records, and Structures tabs also contain a Details tab, which allows you to set up the specifications for those components. The Data Formats Editor has an Overview tab, which provides a visual representation of the entire data format.

Following are detailed procedures for creating data format components. For information on viewing, copying, editing, renaming, deleting, and printing data format components, see “Performing common file management tasks” on page 35. For information on using the grid editors that display in some data format editors, see “Using editor window grids” on page 38. For information on exporting data format components, see “Exporting” on page 52.

The data format component editors are described in the following sections in the order in which you use them when creating a data format from scratch following a top-down approach.

Using the Data Format Dictionary editor

A Data Format Dictionary essentially is a name within which other components are grouped. Following are detailed instructions for creating a new data format dictionary. For information on viewing, copying, editing, renaming, deleting, and printing data format dictionaries, see “Performing common file management tasks” on page 35. For information on exporting data format dictionaries, see “Exporting” on page 52.

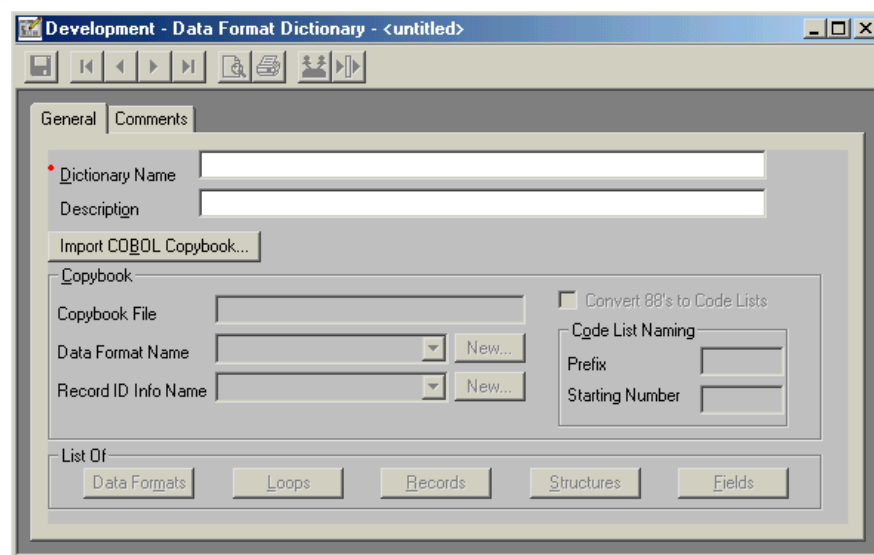
Creating a Data Format Dictionary

Create a new Data Format Dictionary when you set up your first data format for a particular application. You can then reuse data format components for that application when you create subsequent data formats if they are created within that dictionary. You can also import COBOL copybooks using the Data Format Dictionary editor.

◆ To create a new data format dictionary:

1. At the Data Format Dictionary List window, click New Document on the tool bar.

The Data Format Dictionary Editor window displays with the General tab in front and the fields blank.



2. Type a name in the Dictionary Name field. This is a required field, as indicated by the red dot.

The first and last characters can be A-Z and 0-9. All middle characters can be A-Z, 0-9, _, and -. You cannot type spaces within the name. The name can be 30 characters long.

If you wish, you may enter a more complete description of the Data Format Dictionary in the Description field.



NOTE: Once you save the dictionary, the Data Formats, Loops, Records, Structures, and Fields buttons become available in the List Of group box. Click those buttons to display list windows that contain the components associated with this dictionary. When you first create a dictionary, the lists are empty.

3. Click Save on the tool bar to save the dictionary.

After you have saved your dictionary, the Dictionary Name becomes read-only.

Importing a COBOL Copybook

Importing allows you to take a COBOL source file and use it to create DataInterchange fields, structures, and records, and then save them in this Data Format Dictionary.

◆ To import a COBOL Copybook:

1. Use the above procedure to create a dictionary to store the new records and fields. The dictionary must be saved before you proceed. After you save your dictionary, the Dictionary Name becomes read-only and the Import COBOL Copybook button is enabled.

2. Click Import COBOL Copybook.

The Select COBOL Copybook File dialog box displays.

3. Enter the correct path and file name, and click Open.
4. Click Close when the status dialog box indicates the import is complete.

The path and file name of the copybook file display in the Copybook File field, and the Record ID info drop-down list and New button are enabled.

5. Select an existing data format in this dictionary from the drop-down list, or click New.

If a valid data format is selected, the Record ID Info and its New button are disabled. If you click New, the Data Formats Editor displays. Refer to “Using the Data Format Editor” on page 161. Be sure you save the new data format before returning to the Data Format Dictionary Editor.

6. Select the existing record ID info object from the drop-down list, or click New. This is a mandatory field. The record ID info object provides information on how the record is identified to DataInterchange.

If you click New, the Data Format Record ID Information Editor displays. Refer to “Using the Data Format Record ID Information Editor” on page 159. Be sure you save the new record ID information before returning to the Data Format Dictionary Editor.

7. Check the Convert 88s to Codelists box, if you want to convert 88 entries to code lists. One code list will be created for each set of 88-level statements that are associated with the COBOL data name. The names assigned to the code lists are determined by values in the Prefix and Starting Number fields. Checking this box enables the Prefix and Starting Number fields.
 - **Prefix** - Defaults to the first five characters of the data format name, and is added when the check box is selected for the first time.
 - **Starting Number** - Defaults at 1, has a range of 0 to 999.
8. Click Save on the tool bar to save the COBOL objects in the dictionary.



NOTE: Once you save the dictionary, the Data Formats, Loops, Records, Structures, and Fields buttons become available in the List Of group box. Click those buttons to display list windows that contain the components associated with this dictionary. When you first create a dictionary, the lists are empty.

9. Click Records to display a list of data format records belonging to the data format dictionary. You will need to set a record ID for each record added by the import COBOL copybook process.
 - a. Edit each record by double-clicking it. This will open the Data Format Record Editor, which will display the selected record. Select the General tab and set the record ID for the record. Refer to “Using the Data Format Record Editor” on page 166.
 - b. Click Save on the tool bar.
 - c. Repeat for every record on the list.
 - d. Close the list.
10. Back in the Data Format Dictionary Editor, click Data Formats to obtain a list of data formats belonging to the dictionary. You may need to update data format information.
 - a. Edit each Data Format by double-clicking it. This will open the Data Format Editor, which will display the selected data format. Refer to “Using the data format editors” on page 154.
 - b. Click the Raw Data tab if you need to make changes to the raw data information.
 - c. Click the Details tab to add loops that may apply. Refer to “Creating a loop” on page 164.
 - d. Click Save on the tool bar.
 - e. Repeat for every data format on the list that was added by the import COBOL copybook process.
 - f. Close the Data Format List window.
11. Close the Data Format Dictionary Editor by selecting Close from the File menu.

Using the Data Format Record ID Information Editor

The Data Format Record ID Information profile allows DataInterchange to identify records in application data. The profile indicates whether the data format will use raw data format records or C and D format records. When your records are structured using raw data format, the Data Format Record ID Information profile also specifies the location and length of the Record ID.

Unlike Data Format Dictionaries and other components, Record ID Information profiles are global for a system; they are not tied to a specific Data Format Dictionary but can be used in any Data Format Dictionary in the system. If your company structures all application Record ID Information in the same way or always uses C and D records, then you only need to create one Record ID Information profile for use with any data format.

Following are detailed instructions for creating a new data format record ID info profile. For information on viewing, copying, editing, renaming, deleting, and printing data format record ID info profiles, see “Performing common file management tasks” on page 35.

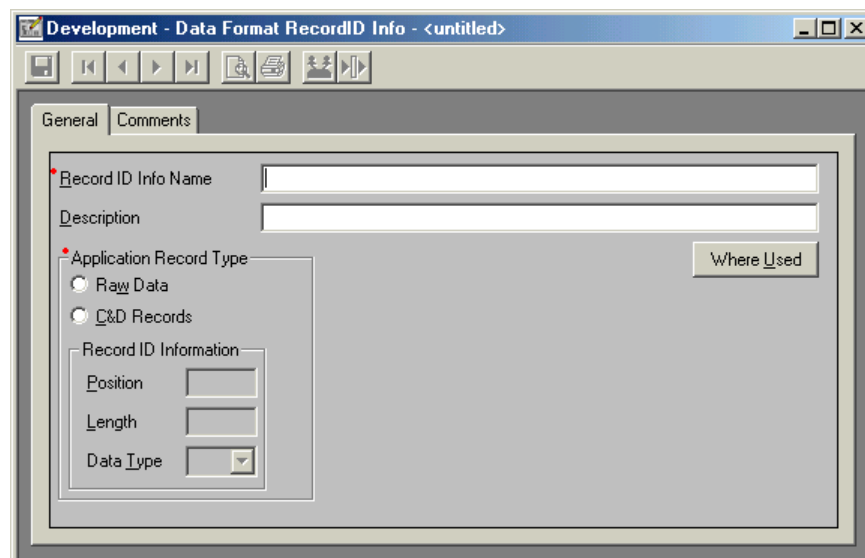
Creating a data format record ID information profile

Create a new Data Format Record ID Information profile when you set up your first data format or when you set up an data format whose record IDs are structured differently than your existing data formats. You can use the same Data Format Record ID Information profile for any data format whose record IDs have the same structure or that are set up as C and D records.

◆ To create a new data format record ID information profile:

1. At the Data Format Record ID Information List window, click New Document on the tool bar.

The Data Format Record ID Information Editor window displays with the General tab in front and the fields blank.



2. Type a name in the Record ID Info Name field. This is a required field, as indicated by the red dot.

The first and last characters can be A-Z and 0-9. All middle characters can be A-Z, 0-9, _, and -. You cannot type spaces within the name. The name can be 30 characters long.

If you wish, you may enter a more complete description of the Data Format Record ID Information profile in the Description field.

3. Indicate whether data formats that specify this Data Format Record ID Info profile will be use raw data format records or C and D format records.

- If your application uses raw data format:

- a. Type the position of the first character of the Record ID in the Position field.

- b. Type the length of the Record ID in the Length field.

- c. Select the Data Type from the drop-down list. Data Types are listed in Table 62, "Data Types for Data Formats," on page 177.

- If your application uses C and D format:

The Position, Length, and Data Type fields automatically fill with the correct values.

4. Click Save on the tool bar to save the profile.

After you have saved your record ID information, the Record ID Info and field becomes read-only.

◆ **To find other data format components that use the current Record ID information profile:**

1. Click Where Used.

A list window displays containing Data Format, Data Format Loop, and Data Format Record tabs.

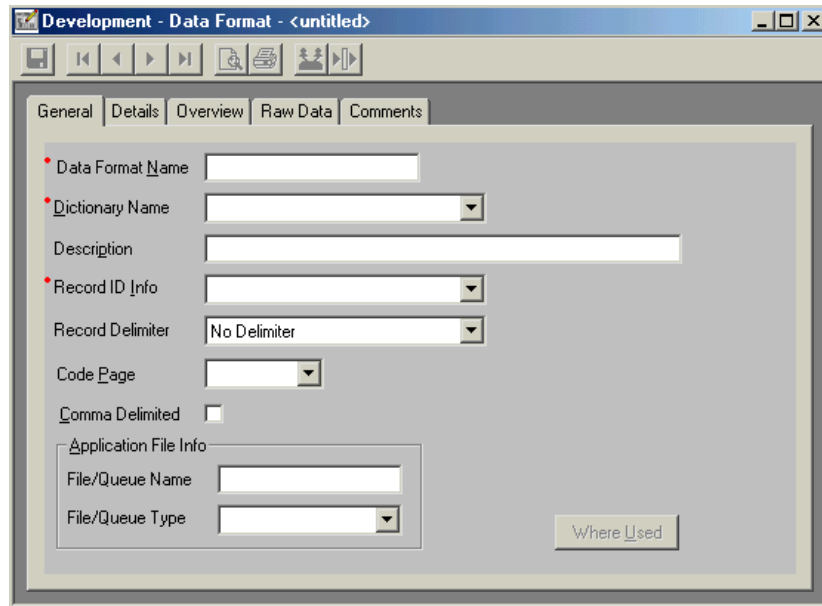
2. Click the tabs of each data format component to display a list of components containing the current Record ID Information profile.

You may open any component by double-clicking it.

An empty list window means that the Record ID Info profile is not used in any data format component of the list window's type.

Using the Data Format Editor

The Data Format Editor allows you to define and structure the components that make up a data format. A data format is a document definition. It defines the layout of your application's data. From the Data Format Editor, you name the data format, identify properties related to the data format, and create and edit the associations to loops and records. It also gives you a visual of the data format from which you can directly navigate to each of its component editors.



The Data Format Editor window contains five tabs: General, Details, Overview, Raw Data, and Comments. Use the:

- General tab to set the name of the data format, select a dictionary and set properties of the data format.
- Details tab to add or change loops and records associated with the selected data format and to edit the information about the association.
- Overview tab to display a visual representation of the entire data format.
- Raw Data tab to enter and change information specific to Raw Data Format translation.
- Comments tab to type any comments you wish about the selected data format.

The editor window also contains a Where Used button, which will list all send and receive maps that use the current data format.

Following are detailed instructions for creating a new data format. For information on viewing, copying, editing, renaming, deleting, and printing data formats, see “Performing common file management tasks” on page 35. For information on exporting data formats, see “Exporting” on page 52.

Creating a data format

Create a new data format when you need to define the new layout of application data that will be used in translation.

◆ To create a new data format:

1. At the Data Formats List window, click New Document on the tool bar.

The Data Format Editor window displays with the General tab in front and the fields blank.


2. Type a name in the Data Format Name field. This is a required field, as indicated by the red dot.

The first and last characters can be A-Z and 0-9. All middle characters can be A-Z, 0-9, _ and -. You cannot type spaces within the name. The name can be 16 characters long.



NOTE: If you use a generic usages or rules, this name is limited to eight characters. For more information on generic usages, see “Defining generic send usages” on page 277.

If you wish, you may enter a more complete description of the data format in the Description field.

3. Use the drop-down list to select the Data Format Dictionary in which you want the data format to occur. This is a required field.
4. Use the drop-down list to select the Record ID Information profile you wish to use for this data format. This is a required field. All loops and records used in this data format must use the same Record ID Information profile.
5. Fill in the optional fields in the Application File Info.
6. Click the Raw Data tab and fill in the fields as needed.
Click  for field names and descriptions.
7. Click the Details tab to enter information on the loops and records contained in this data format.
 - a. From the Type drop-down list, select either Record or Loop, depending on which you are entering.
 - b. Select either the loop or record name from the Loop/Record Name drop-down list. This list displays loops and records that are in the same dictionary and use the same Record ID Information profile as this data format. If you have not created any loops or records using the respective editors, this drop-down list will be empty. You may create a record or loop in this grid by typing in a name.

DataInterchange fills in the default values of the remaining columns, when an existing loop or record is selected.

- c. Select Header, Detail, or Trailer from the Area drop-down list, depending on where the loop or record is used. The default value is Detail. Use these values to organize your data format into sections containing heading information (such as bill-to address), detail information (such as purchase order line items) and trailing information (such as totals).
- d. Enter the maximum number of times the loop or record can be used in the Max Use column. If the loop or record can repeat infinitely either enter the number 32767 or click the Infinite check box to insert a check mark.
- e. For new records, enter the Record ID in the Record ID column. For existing records, DataInterchange Client displays the Record ID as a read-only field. If you would like to edit the Record ID, you can do so from the Data Format Record Editor window.



NOTE: If you are using C and D records, you do not need record ID information. Records are identified by the record name.

- f. For new records and loops, you may type in a more detailed description of the record or loop in the Description column. For existing records or loops, DataInterchange Client displays the description as a read-only field. If you would like to edit the description, you can do so from that particular loop or record's editor window.

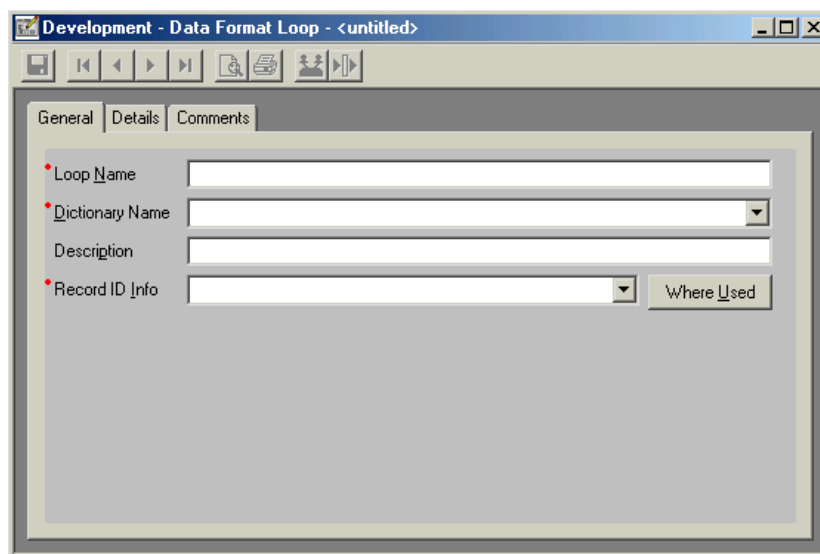
For information on how to use the editor window's grid, see "Using editor window grids" on page 38.

8. When you have completed entering the information, click Save on the tool bar to save the data format.

After you have saved your data format, the Data Format Name, Record ID Info, and Dictionary Name fields become read-only.

Using the Data Format Loop Editor

Use the Data Format Loop Editor window to enter new loops into a Data Format Dictionary or to edit existing loops. Loops are entered into a data format using the Data Format Editor when you create or edit a data format. From the Data Format Loop Editor, you can create and edit the loop's associations with loops and records, as loops can contain loops and records.



The Data Format Loop Editor window contains three tabs: General, Details, and Comments. Use the:

- General tab to name the loop, select its dictionary and set loop properties.
- Details tab to add or change loops and records associated with the selected loop and to edit the information about the association.
- Comments tab to type any comments you wish about the selected loop.

The editor window also contains a Where Used button, which allows you to find other data format components that use the currently selected loop.

Following are detailed instructions for creating a new loop. For information on viewing, copying, editing, renaming, deleting, and printing loops, see “Performing common file management tasks” on page 35.

Creating a loop

Create a new loop when your application data has two or more records that are grouped together and each group can occur more than once.

◆ **To create a new loop:**

1. At the Data Format Loop List window, click New on the tool bar.

The Data Format Loop Editor window displays with the General tab in front and the fields blank.

2. Type a name in the Loop Name field. This is a required field, as indicated by the red dot.

The first and last characters can be A-Z and 0-9. All middle characters can be A-Z, 0-9, , and . You cannot type spaces within the name. The name can be 30 characters long.

If you wish, you may enter a more complete description of the loop in the Description field.

3. Select the Data Format Dictionary in which you want the loop to occur using the Dictionary Name drop-down list. This is a required field.

4. Select the Record ID Information profile you wish to use for this loop through the Record ID Info drop-down list. This is a required field.

5. Click the Details tab to enter information on the loops and records contained in this loop.

- a. From the Type drop-down list, select either Record or Loop, depending on which you are entering.

Loops may contain both loops and records, but must begin with a record. The value of Max Use for the first record must be one (1).

- b. Either select the loop or record name from the Loop/Record Name drop-down list, or type in a loop or record name to create a new loop or record. The drop-down list displays loops and records that are in the same dictionary and use the same Record ID information profile as this loop.

The remaining columns fill in with their default values.

If you have not created any loops or records using their respective editors, this drop-down list will be empty. You may create a record or loop in this grid by typing in a name.

- c. Enter the maximum number of times the loop can be used in the Max Use column. If the loop or record can repeat infinitely either enter the number 32767 or click the Infinite check box to insert a check mark.
 - d. For new records, enter the Record ID in the Record ID column. For existing records, DataInterchange Client displays the Record ID as a read-only field. If you would like to edit the record ID, you can do so from the Data Format Record Editor window.
 - e. For new records and loops, you may type in a more detailed description of the record or loop in the Description column. For existing records and loops, DataInterchange Client displays the description as a read-only field. If you would like to edit the description you can do so from that particular loop or record's editor window.

For information on how to use the editor window's grid, see "Using editor window grids" on page 38.

6. When you have completed entering information, click Save on the tool bar to save the loop.

After you have saved your loop, the Loop Name, Record ID Info, and Dictionary Name fields become read-only.

◆ **To find other data format components that use the current data format loop:**

1. Click Where Used.

A list window containing Data Format and Data Format Loop tabs displays.

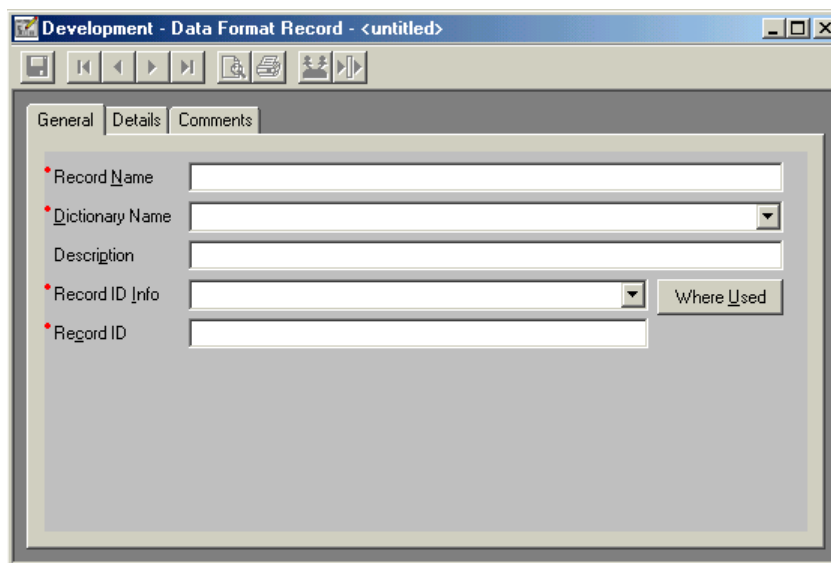
2. Click the tabs of each data format component to display a list of components containing the current loop.

You may open any item in the list by double-clicking it.

An empty list window means that the loop is not used in any other data format component of the list window's type.

Using the Data Format Record Editor

Use the Data Format Record Editor window to enter new records into a Data Format Dictionary or to edit existing records. From the Data Format Record Editor, you create and edit the record's associations with fields and structures, as records can contain fields and structures.



The Data Formats Record Editor window contains three tabs: General, Details, and Comments. Use the:

- General tab to name the record, select its dictionary, and set properties.
- Details tab to add or change fields and structures associated with the selected record and to edit the information about the association.
- Comments tab to type any comments you wish about the selected record.

The editor window also contains a Where Used button, which allows you to find other data format components that use the currently selected record.

Following are detailed instructions for creating a new record. For information on viewing, copying, editing, renaming, deleting, and printing records, see “Performing common file management tasks” on page 35.

Creating a record

Create a new record when your application data layout requires one. Records can contain fields and structures.

◆ To create a new record:

1. At the Data Format Record List window, click New on the tool bar.
The Data Format Record window displays with the General tab in front and the fields blank.
2. Type a name in the Record Name field. This is a required field, as indicated by the red dot.
The first and last characters can be A-Z and 0-9. All middle characters can be A-Z, 0-9, _, and -. You cannot type spaces within the name. The name can be 30 characters long.
If you wish, you may enter a more complete description of the record in the Description field.
3. Select the Data Format Dictionary in which you want the record to occur using the Dictionary Name drop-down list. This is a required field.
4. Select the Record ID Information profile you wish to use for this record through the Record ID Info drop-down list. This is a required field.
5. Type the name of the Record ID in the Record ID field. This is a required field.



NOTE: If you are using C and D records, this field is not available. Records are identified by the record name.

6. Click the Details tab to enter information on the fields and structures contained in this record.
 - a. From the Type drop-down list, select either Field or Structure, depending on which you are entering.
Records may contain both fields and structures.
 - b. Select either the field or structure name from the Structure/Field Name drop-down list, which displays fields and structures that are in the same dictionary.

The remaining columns fill in with their default values.

If you have not created any fields or structures using their respective editors, this drop-down list will be empty. You may create a field or structure in this grid by typing in a name.

- c. Enter the number of times the structure is used in the Occurs column. Fields are fixed at one (1). Alternately, you can indicate that the number of times a structure repeats is dictated by the value found in one of its peer fields. Do this by selecting a field from the Occurs Depending drop-down list.

Occurs Depending is only supported for data transformation maps.

- d. For new fields, select the data type from the Data Type drop-down list. Type in the field length in the Field Len column. For definitions of DataInterchange data types, see Table 62, “Data Types for Data Formats,” on page 177. For existing fields, DataInterchange Client displays the Data Type and Field Len fields as a read-only. If you want to edit these fields, you can do so from the Data Format Field Editor.
- e. For a new field or structure, you may type in a more detailed description of the field or structure in the Description column. For an existing field or structure, DataInterchange Client displays the description as a read-only field. If you would like to edit the description you can do so from that particular field or structures’ editor window.

For information on how to use the editor window’s grid, see “Using editor window grids” on page 38.

- 7. When you have completed entering information, click Save on the tool bar to save the record.

After you have saved your record, the Record Name, Record ID Info, and Dictionary Name fields become read-only.

◆ **To find other data format components that use the current data format record:**

- 1. Click Where Used.

A list window containing Data Format and Data Format Loop tabs displays.

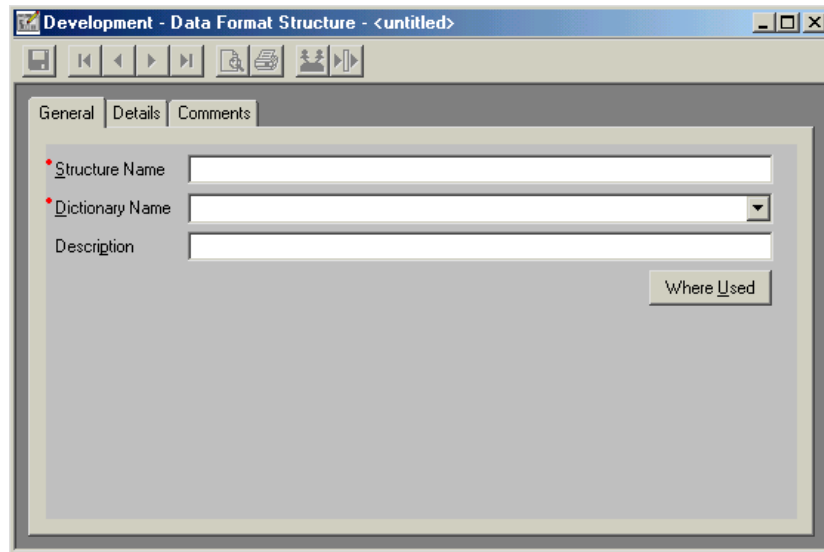
- 2. Click the tabs of each data format component to display a list of components containing the current data format record.

You may open any item in the list by double-clicking it.

An empty list window means that the data format record is not used in any other data format component of the list window’s type.

Using the Data Format Structure Editor

Use the Data Format Structure Editor window to enter new structures into a Data Format Dictionary or to edit existing structures. From the Data Format Structure Editor, you create and edit the structure's associations with fields and structures, as structures can contain fields and structures.



The Data Formats Structures Editor window contains three tabs: General, Details, and Comments. Use the:

- General tab to name the structure, select its dictionary, and set properties.
- Details tab to add or change fields and structures associated with the selected structure and to edit the information about the association.
- Comments tab to type any comments you wish about the selected structure.

The editor window also contains a Where Used button, which allows you to find other data format components that use the currently selected structure.

Following are detailed instructions for creating a new structure. For information on viewing, copying, editing, renaming, deleting, and printing structures, see "Performing common file management tasks" on page 35.

Creating a structure

Create a new structure when your application data has two or more contiguous fields that should logically occur together. For example, a structure describing a date may contain fields for the year, month and day. In another example, Name, Address, City, State, and Zip occur three times in a single record in the same order. You can define those fields as a structure one time and say that it repeats three times in the record.

◆ To create a new structure:

1. At the Data Format Structure List window, click New on the tool bar.

The Data Format Structure Editor window displays with the General tab in front and the fields blank.

2. Type a name in the Structure Name field. This is a required field, as indicated by the red dot.

The first and last characters can be A-Z and 0-9. All middle characters can be A-Z, 0-9, `_`, and `-`. You cannot type spaces within the name. The name can be 30 characters long.

If you wish, you may enter a more complete description of the structure in the Description field.

3. Select the Data Format Dictionary in which you want the structure to occur using the Dictionary Name drop-down list. This is a required field.
4. Click the Details tab to enter information on the fields and structures contained in this structure.
 - a. From the Type drop-down list, select either Structure or Field, depending on which you are entering.
 - b. Select either the field or structure name from the Structure/Field Name drop-down list, which displays fields and structures that are in the same dictionary.

The remaining columns fill in with their default values.

If you have not created any fields or structures using their respective editors, this drop-down list will be empty. You may create a field or structure in this grid by typing in a name.

- c. Enter the number of times the structure is used in the Occurs column. Fields are fixed at one (1). Alternately, you can indicate that the number of times a structure repeats is dictated by the value found in one of its peer fields. Do this by selecting a field from the Occurs Depending drop-down list.

Occurs Depending is only supported for data transformation maps.

- d. For new fields, select the data type from the Data Type drop-down list. Type in the field length in the Field Len column. For definitions of DataInterchange data types, see Table 62, "Data Types for Data Formats," on page 177. For existing fields, DataInterchange Client displays the Data Type and Field Len fields as read only. If you want to edit those fields, you can do so from the Data Format Field Editor window.

- e. For new fields and structures, you may type in a more detailed description of the field or structure in the Description column. For existing fields and structures, DataInterchange Client displays the description as a read-only field. If you would like to edit the description, you can do so from that particular field or structure's editor window.

For information on how to use the editor window's grid, see "Using editor window grids" on page 38.

5. When you have completed entering information, click Save on the tool bar to save the structure.

After you have saved your structure, the Structure Name and Dictionary Name fields become read-only.

◆ To find other data format components that use the current structure:

1. Click Where Used.

A list window containing Data Format Record and Data Format Structure tabs displays.

2. Click the tabs of each data format component to display a list of components containing the current structure.

You may open any item in that component by double-clicking it.

An empty list window means that the structure is not used in any other data format component of the list window's type.

Using the Data Format Field Editor

Use the Data Format Field Editor window to enter new fields into a Data Format Dictionary or to edit existing fields. From the Data Format Field Editor, you can set up and maintain properties of a data format field.

The screenshot shows the 'Development - Data Format Field - <untitled>' window. It has a toolbar with icons for Save, Undo, Redo, and others. The 'General' tab is selected, showing the following fields:

- Field Name:** A text input field.
- Dictionary Name:** A dropdown menu.
- Description:** A text input field.
- Data Type:** A dropdown menu with 'CH' selected.
- Field Length:** A text input field with '1' entered.
- Mapping Command:** A section containing a 'Literal' text input field and two checkboxes: 'Send Map Validation' and 'Receive Map Validation'.
- User Code List:** A dropdown menu.
- Where Used:** A button located at the bottom right of the form area.

The Data Formats Fields Editor window contains two tabs: General and Comments. Use the:

- General tab to enter the field name, select its dictionary, and set properties of the field.
- Comments tab to type any comments you wish about the selected field.

The editor window also contains a Where Used button, which allows you to find other data format components that use the currently selected field.

Following are detailed instructions for creating a new field. For information on viewing, copying, editing, renaming, deleting, and printing fields, see “Performing common file management tasks” on page 35.

Creating a field

Create a new field when your application data requires one.

◆ To create a new field:

1. At the Data Format Field List window, click New Document on the tool bar.

The Data Format Field Editor window displays with the General tab in front and the fields blank.

2. Type a name in the Field Name field. This is a required field, as indicated by the red dot.

The first and last characters can be A-Z and 0-9. All middle characters can be A-Z, 0-9, _, and -. You cannot type spaces within the name. The name can be 30 characters long.

If you wish, you may enter a more complete description of the field in the Description field.

3. Select the Data Format Dictionary in which you want the field to occur using the Dictionary Name drop-down list. This is a required field.
4. Select the data type from the Data Type drop-down list. For definitions of DataInterchange data types, see Table 62, “Data Types for Data Formats,” on page 177. This is a required field.
5. Type the length of the field in the Field Length field. This is a required field.

If this field is required by a trading partner or the EDI standard but may not always contain a value, you can use DataInterchange literals or mapping commands to enter information into the field.



NOTE: This is supported in send and receive maps only. It is ignored in data transformation maps.

- a. Type the name of the literal you wish to use in the Literal field. For a list of DataInterchange literals or mapping commands, search in DataInterchange Client Help on the keyword “literals”.

- b. Literals and mapping commands are validated differently by the translator depending on whether they are used by send maps or receive maps. If you want this value validated, click either or both the Send Map Validation check box or the Receive Map Validation check box, depending which map types the literal or mapping command will be used in.
6. You may associate code lists with fields to validate the data they contain against values in a specific list. Select the list you wish to validate this field against from the Code List drop-down list. This field is used only in send and receive maps.

For information on creating User Code Lists, see “Using the Code List Editor” on page 202.

7. When you have completed entering information, click Save on the tool bar to save the field.
- After you have saved your field, the Field Name and Dictionary Name fields become read-only.

◆ **To find other data format components that use the current field:**

1. Click Where Used.

A list window containing Data Format Record and Data Format Structure tabs displays.

2. Click the tabs of each data format component to display a list of components containing the current field.

You may open any item in the list by double-clicking it.

An empty list window means that the field is not used in any other data format component of the list window's type.

Navigating Data Format Component Editors

DataInterchange Client's Data Format Editor windows are designed to provide maximum flexibility. You can move from editor to editor with ease so that you can tailor your navigation to the requirements of your work.

This section provides information on the various paths from one editor to the next.

Data format dictionary editor paths

The buttons at the bottom of the Data Format Dictionary General tab generate list windows for each data format component that occurs in the current dictionary. From those list windows, you can open the editor for components in the list, where you can edit the current component or create a new one of that type.

This button. . .	Takes you to:
Data Formats	The data formats associated with this dictionary
Loops	The loops associated with this dictionary
Records	The records associated with this dictionary
Structures	The structures associated with this dictionary
Fields	The fields associated with this dictionary

Data Format Editor paths

From the Data Format Editor, you can move to any other component of a data format. The most powerful navigational tool in the Data Format Editor window is its Overview tab, which displays a visual representation of your data format:

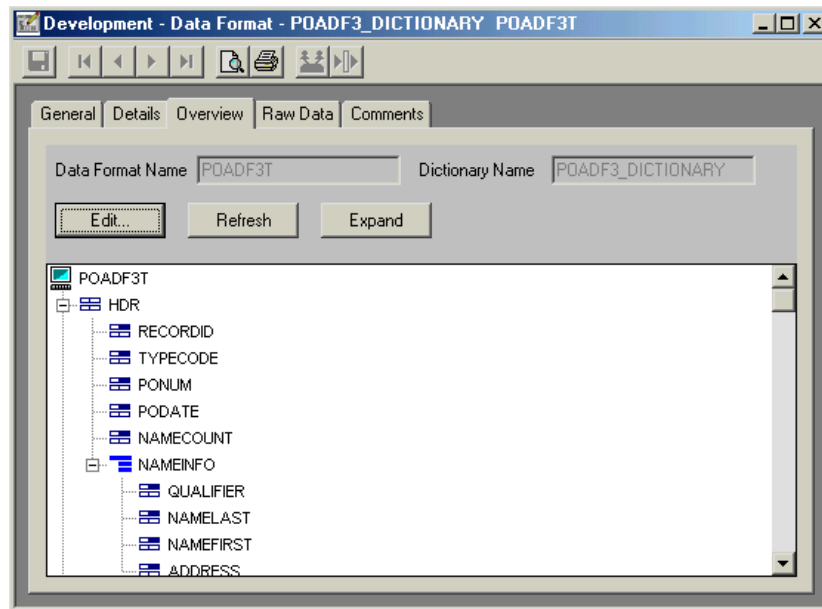






Table 61 defines the symbols on the Data Formats Editor Overview tab.

Table 61. Data Format Symbols

This symbol. . .	Represents:
	A loop
	A record
	A structure
	A field

Click the + sign to expand a section of the data format.

Click the - sign to collapse that section of the data format.

The Expand button expands the node which is currently selected. To expand the entire tree, select the root node and click Expand.

The Overview tab allows you to navigate through the data format as follows:

- Double-clicking a loop starts the Data Format Loop Editor for the loop on which you clicked.
- Double-clicking a record starts the Data Format Record Editor for the record on which you clicked.
- Double-clicking a structure starts the Data Format Structure Editor for the structure on which you clicked.
- Double-clicking a field starts the Data Format Field Editor for the field on which you clicked.

From its Details tab, the Data Format Editor window allows you to start the Data Format Loop Editor and the Data Format Record Editor.

- To start the Data Format Loop Editor, double-click the number of the row containing the loop you wish to edit. You may also select the row and then click Edit.
- To start the Data Format Record Editor, double-click the number of the row containing the record you wish to edit. You may also select the row and then click Edit.

Data Format Loop Editor paths

Within the Data Format Loop Editor, you can navigate to data formats, loops, and records, as follows:

From the Data Format Loop Editor General tab, you can display a list of data formats and loops in which the current loop is used.

Click Where Used to display a list window containing Data Format and Data Format Loop tabs. Double-clicking entries in those lists starts the respective editors.

From the Data Format Loop Editor Details tab, you can start the Data Format Loop and Data Format Record Editors.

To start the Data Format Loop Editor, double-click the number of the row containing the loop you wish to edit. You may also select the row and then click Edit.

To start the Data Format Record Editor, double-click the number of the row containing the record you wish to edit. You may also select the row and then click Edit.

Data Format Record Editor paths

Within the Data Format Record Editor, you can navigate to data formats, loops, structures, and fields, as follows:

From the Data Format Record Editor General tab, you can display a list of data formats and loops in which the current record is used.

Click Where Used to display a list window containing Data Format and Data Format Loop tabs. Double-clicking entries in those lists starts the respective editors.

From the Data Format Record Editor Details tab, you can start the Data Format Structure and Data Format Field Editors.

To start the Data Format Structure Editor, double-click the number of the row containing the structure you wish to edit. You may also select the row and then click Edit.

To start the Data Format Field Editor, double-click the number of the row containing the field you wish to edit. You may also select the row and then click Edit.

Data Format Structure Editor paths

Within the Data Format Structure Editor, you can navigate to records, structures, and fields, as follows:

From the Data Format Structure Editor General tab, you can display a list of records and structures in which the current structure is used.

Click Where Used to display a list window containing Data Format Record and Data Format Structure tabs. Double-clicking entries in those lists starts the respective editors.

From the Data Format Structure Editor Details tab, you can start the Data Format Structure and Data Format Field Editors.

To start the Data Format Structure Editor, double-click the number of the row containing the structure you wish to edit. You may also select the row and then click Edit.

To start the Data Format Field Editor, double-click the number of the row containing the field you wish to edit. You may also select the row and then click Edit.

Data Format Field Editor Paths

Within the Field Editor, you can navigate to records and structures, as follows:

From the Data Format Field Editor General tab, you can display a list of records and structures in which the current field is used.

Click Where Used to display a list window containing Data Format Record and Data Format Structure tabs. Double-clicking entries in those lists starts the respective editors.

Reusing data format components

DataInterchange Client allows you to reuse data format components.

The Data Format Dictionary is the mechanism that allows you to reuse components.

- Once you create a component in a Data Format Dictionary, you can use that component in any **other** data format associated with that dictionary, but you cannot reuse that component in the **same** component.
- You cannot use a loop or structure within itself, directly, or indirectly. That causes a circular reference.
- Names of all components within a dictionary must be unique.
- All data format names must be unique, regardless of dictionary.



ATTENTION: Because you can reuse components, it is important that you check where each component is used when you change it. Your changes may propagate through several data formats. Use the Where Used button on the general tabs to see which data format components will be affected by any change you make.

Data types for data formats

The following table lists the valid data types permitted in data format fields. These data types describe the contents of the fields in your application's data. It also shows the valid mapping data type associated with the data format data type. The EDI Standard Data Types column shows equivalent EDI standard data types.

Most data format data types use the same format for storing, displaying, and printing data. However, sometimes the storage format is different from the format used to display or print data. These differences are noted in Table 62.



NOTE: Data types that contain binary data should not be used for comma-separated data formats. This can result in unpredictable behavior since the binary data may be interpreted as record or field delimiters.

Table 62. Data Types for Data Formats

Data Format	EDI Standard Data Types	Description
A	A AN ID	Alphabetic Any combination of characters from the ALHPANUM table, except the digits 0-9.
AC	A AN ID Nn Rn DT TM	Application control A field that contains a control number by which the application identifies the transaction. A purchase order number is an example. The data itself is alphanumeric. A data format can contain only one field of this type. During transaction mapping, you can specify the application control as a concatenation of up to eight fields. The concatenated application control overrides the AC data type. This data type does not apply to record IDs. An AC data type is assumed to be the same as AN during the value validation at translate time.
AN	A AN ID Nn Rn DT TM	Alphanumeric You can use any combination of characters up to the length of the field.
Bn	AN ID Nn Rn DT TM	Binary (unsigned) Storage format: Data with a binary format with n implied decimal places. A value of 2.3 defined as a 2-byte B2 field would be stored as 1110 0110 (X'E6' or decimal 230). This is the same format as IT or In data, but binary data is not signed and, therefore, all values are considered positive.

Table 62. Data Types for Data Formats

Data Format	EDI Standard Data Types	Description
BN	AN ID Nn Rn DT TM	<p>Binary (unsigned)</p> <p>Any combination of 0-9 without a sign (+ or -).</p> <p>Storage format: The binary equivalent of a numeric value in either two or four bytes, depending on the length of the field.</p> <p>Example: The value 23 is stored as 0000 0000 0001 0111 (X'0017').</p>
CH	A AN ID Nn Rn DT TM	<p>Character</p> <p>Any combination of characters up to the length of the field.</p>
DT	DT	<p>Date; does not apply to record IDs</p> <p>A string of 5 to 8 digits, depending on the date format that is used. The acceptable date formats are:</p> <p><i>ddmmyy, ddmmyyyy, ddyymm, ddyyyyym, ddy, ddyyy</i></p> <p><i>mmddy, mmdyyy, mmyydd, mmyyydd</i></p> <p><i>yyymmdd, yyyyymmdd, yyddmm, yyyyddmm, yydd, yyyydd</i></p>
FN		<p>File name.</p> <p>A field that contains the name of a file whose entire contents are mapped to a binary segment. When not mapped to a binary segment, a field with data type FN is treated as if the data type were AN.</p>
Hn	AN ID Nn Rn DT TM	<p>Hexadecimal</p> <p>Hexadecimal data with n implied decimal places.</p> <p>This format is treated as a Bn field when mapped to a numeric data element and as an HX field when mapped to an alpha data element.</p>
HX	AN ID Nn Rn DT TM	<p>Hexadecimal</p> <p>Any combination of 0-9 and A-F up to twice the length of the field.</p> <p>Storage format: Hexadecimal, where the length of the field determines the number of bytes used to hold the value.</p>
ID		<p>Identifier</p> <p>The ID data type is equivalent to an AN data type.</p>

Table 62. Data Types for Data Formats

Data Format	EDI Standard Data Types	Description
In	AN ID Nn Rn DT TM	Integer (signed) Storage format: Data with a binary format with n implied decimal places. A value of 2.3 defined as a 4-byte I2 field would be stored as 0000 0000 1110 0110 (X'E6' or decimal 230).
IT	AN ID Nn Rn DT TM	Integer (signed) Storage format: The binary equivalent for a positive number or the two's complement binary equivalent for a negative number, in two or four bytes, depending on the length of the field. Example: The value +23 is stored as 0000 0000 0001 0111 (X'0017'). The value -23 is stored as 1111 1111 1110 1001 (X'FFE9').
Ln	AN ID Nn Rn DT TM	Decimal (leading sign) Zoned decimal data with n implied decimal places and a leading sign.
N	AN ID Nn Rn DT TM	Numeric Any combination of 0-9 and an optional sign (+ or -). The length includes the sign. When mapping data elements defined as data type N in UN/EDIFACT standards, use data type R.
Nn	AN ID Nn Rn DT TM	Numeric Any combination of 0-9, an implied decimal point with n places to the right of the decimal, and an optional sign (+ or -). Using N alone is the same as using N0 (N zero). The length includes the sign. Example: N2 for a value of 23949 is interpreted as 239.49.
PD	AN ID Nn Rn DT TM	Packed decimal Any combination of 0-9 with a sign (+ or -). The length defines the number of bytes used to hold the value in external format (minus the sign position). Storage format: The packed decimal equivalent, followed by the sign in the low-order 4 bits of the last byte. The sign is either 1111, 1100, or 1010 for a positive value; or, 1101 or 1011 for a negative value. Example: The value +123 is stored as 0001 0010 0011 1111 (X'123F'). The value -123 is stored as 0001 0010 0011 1101 (X'123D').

Table 62. Data Types for Data Formats

Data Format	EDI Standard Data Types	Description
Pn	AN ID Nn Rn DT TM	Packed decimal Packed decimal data with n implied decimal places.
R	AN ID Nn Rn	Real Numeric data that requires a decimal point for fractional values. The decimal point is optional for integers. A sign (+ or -) is optional for positive numbers. Positive is assumed if a sign is not present. The length includes the decimal point and sign if they are present. Scientific notation with exponent and mantissa formatting is used. You should use this data when mapping data elements defined as data type N in UN/EDIFACT standards. Examples: 23.949, +23.949, -23949, -39846.7, 50E+4.
Rn	AN ID Nn Rn	Real Signed or unsigned numeric data with a minimum for n significant decimal places. The length includes the decimal point and sign. Any combination of 0-9 with a sign (+ or -).
TM		Time A string of four digits in the form <i>hhmm</i> or six digits in the form <i>hhmmss</i> , expressed in the 24-hour clock format, where the hour is specified as 00 to 23 for X12 and 00 to 24 for EDIFACT. This data type does not apply to record IDs.
ZD	AN ID Nn Rn DT TM	Zoned decimal Any combination of 0-9 with a sign (+ or -). The length defines the number of characters used to represent the value in the external format. The external length requires an extra position for the sign. Storage format: The zoned decimal equivalent in the low-order 4 bits of a byte and 1111 in the high-order 4 bits. The sign displays in the high-order 4 bits of the low-order byte and is either 1100 for a positive value or 1101 for a negative value. The length of the field determines the number of bytes used to store the value. Example: The value +123 is stored as 1111 0001 1111 0010 1100 0011 (X'F1F2F3'). The value -123 is stored as 1111 0001 1111 0010 1101 0011 (X'F1F2D3').

Table 62. Data Types for Data Formats

Data Format	EDI Standard Data Types	Description
Zn	AN ID Nn Rn DT TM	Zoned decimal Zoned decimal data with n implied decimal places and a trailing sign. Any combination of 0-9 with a sign (+ or -).

Extensible Markup Language (XML)

Extensible Markup Language (XML) is a language that is becoming a popular way to represent structured documents and data. XML defines the syntax used to represent the data, including information such as how to tell an element name from an element value. However, it does not define the semantics or structure of the data. For example, it does not specify where a purchase order number should appear, or even whether it is part of any particular document.

The structure of an XML document is defined by a Document Type Definition (DTD). The syntax of the DTD is defined as part of the XML language. The DTD provides a list of all components included in the XML document and their relationship to each other. For example, a DTD may state that the Header element of the document contains a PONum element. The meaning of the PONum element may be described either in the comments within the DTD, a separate document, or both.

Unlike EDI standards where there are a small number of dominant EDI standard formats that define the document structure, there are numerous different XML DTDs. Also, since XML is "eXtensible", users are free to create their own DTDs if they choose. Instead of restricting users to a fixed subset of DTDs, DataInterchange allows you to import the DTDs you use. You may obtain your DTDs from various sources, such as industry groups, standards bodies, vendors, trading partners, or you may create them yourself. Once the DTDs are imported into DataInterchange, you can map the documents described by those DTDs.



NOTE: DataInterchange does not currently support XML Schemas. XML Schemas are a newer way of describing the structure of an XML document.

Imported DTDs are used only for data transformation maps; they cannot be used for send and receive maps.

DataInterchange uses XML Dictionaries as way to logically group a set of related DTDs. A DTD name must be unique within its XML dictionary, but does not have to be unique across all XML dictionaries. A DTD may use external references to refer to other DTDs within the same dictionary. Use the XML Dictionary Editor to create and maintain an XML dictionary.

The XML List window provides access to the XML dictionaries and DTDs. The window displayed when you press the XML button on the navigator bar contain two tabs, as follows:

- The XML Dictionary tab provides access to the XML Dictionary List window and XML Dictionary Editor window.
- The DTDs tab provides access to the DTDs List window and the DTDs Editor window.

Accessing XML editors

You access the component editors in essentially the same way. Both the editors display on tabs in the XML List window, as follows.



NOTE: You cannot change the DTD itself, only the DataInterchange properties associated with the DTD. To change a DTD, edit the original DTD outside of DataInterchange using a text editor or a DTD editor, then re-import the DTD into DataInterchange.

◆ To access an XML editor:

1. Click XML on the DataInterchange Client Navigator bar.

The XML List window, which contains tabs for the components, displays.

2. Click the tab of the XML component you wish to work with.

The list window for that component displays.

	Dictionary Name	Description	Lock	Updated User ID	Updated Date/Time
1	DIC001	test dictionary	No	admin	10/22/2001 11:00:42 AM
2	RNMXML_DICTIONARY		No	admin	11/06/2001 09:56:05 AM
3	TEST		No	admin	10/22/2001 11:00:42 AM
4	POXML5SR	poxml_dictionary	No	admin	11/06/2001 09:59:34 AM

This window displays a list of existing XML components. A list of XML dictionaries is shown above. Each row contains information about a component; each column contains data stored in that component. Information in the columns displays in fields in the editor window.

3. To view an item or to add or change information in an item, double-click the row of the item you wish to work with.

The editor window displays. You add information or make changes to the component through its tabs, as described in the following sections.

Following are detailed procedures for creating XML dictionaries and importing DTDs. For information on viewing, copying, editing, renaming, deleting, and printing components, see “Performing common file management tasks” on page 35. For information on exporting components, see “Exporting” on page 52.

The XML component editors are described in the following sections.

Using the XML Dictionary Editor

An XML dictionary essentially is a name within which related DTDs are grouped. The relationship can be DTDs that refer to one another or DTDs created and maintained by an organization.



NOTE: The XML dictionary in DataInterchange 4.1 is not related to the XML dictionary created by the DTD Conversion Utility in DataInterchange 3.1.

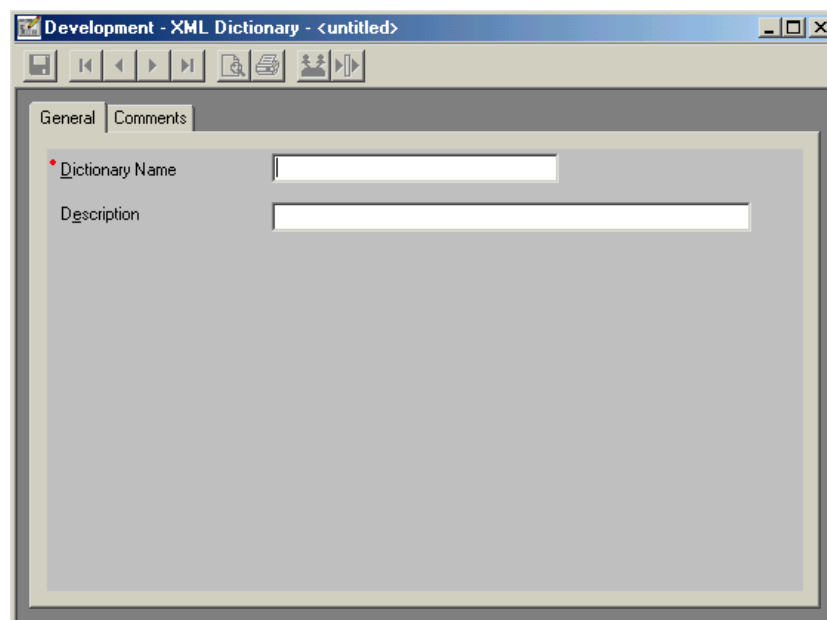
Creating an XML dictionary

Create a new dictionary when you want to create a logical group of DTDs. For example, you might want to put all DTDs from a particular XML format in one dictionary.

◆ To create a new XML dictionary:

1. At the XML Dictionary List window, click New on the tool bar.

The XML Dictionary Editor window displays with the General tab in front and the fields blank.



2. Type a name in the Dictionary Name field.

The name displays in all capital letters. You cannot type spaces within the name.

3. Enter a more complete description of the XML Dictionary in the Description field.
4. Click Save on the tool bar to save the dictionary.

Importing a DTD file

After you have created an XML dictionary, DTDs can be imported into DataInterchange. Once you have obtained a DTD file you wish to load into DataInterchange, refer to “Importing a DTD file” on page 57.

Using the DTD Editor

The DTD Editor is used to maintain properties of the DTD and to view the DTD. It can not be used to change the DTD itself. Change the original DTD using a text editor or a DTD editor.

◆ To open the DTD Editor and view a DTD:

1. Click XML on the DataInterchange Client Navigator bar.

The XML List window, which contains tabs for the components, displays.

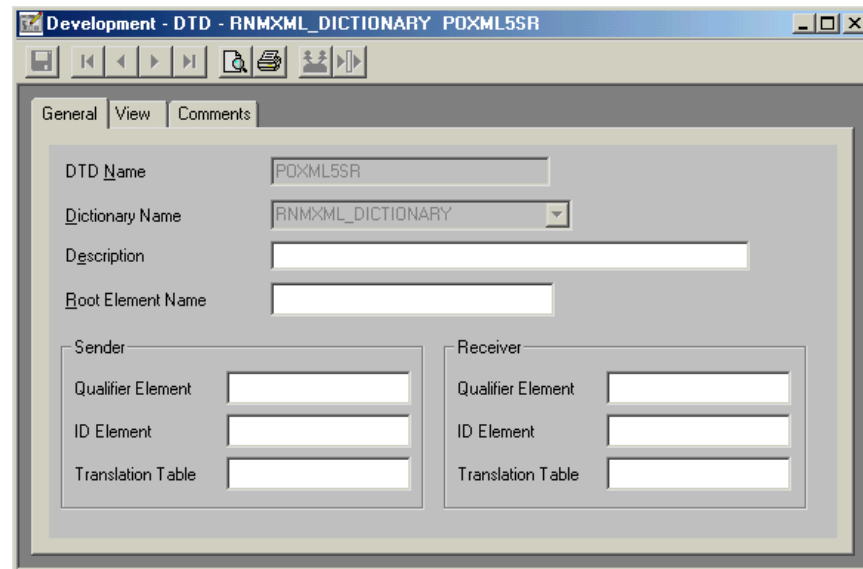
2. Click the DTDs tab.

The list window for DTDs displays.


	Dictionary Name	DTD Name	Sender ID Element	Root Element	Sender
1	DIC001	CXML			
2	DIC001	BPRESOURCES		ttt	
3	DIC001	AUDIENCECOD			
4	RNMXML_DICTIONARY	POXMLSSR			
5	RNMXML_DICTIONARY	RNMXML		WWWRE_ItemCreation	
6	TEST	POXML5		Order	

3. Double-click the row of the DTD you wish to work with.

The General tab in the DTD Editor window displays.

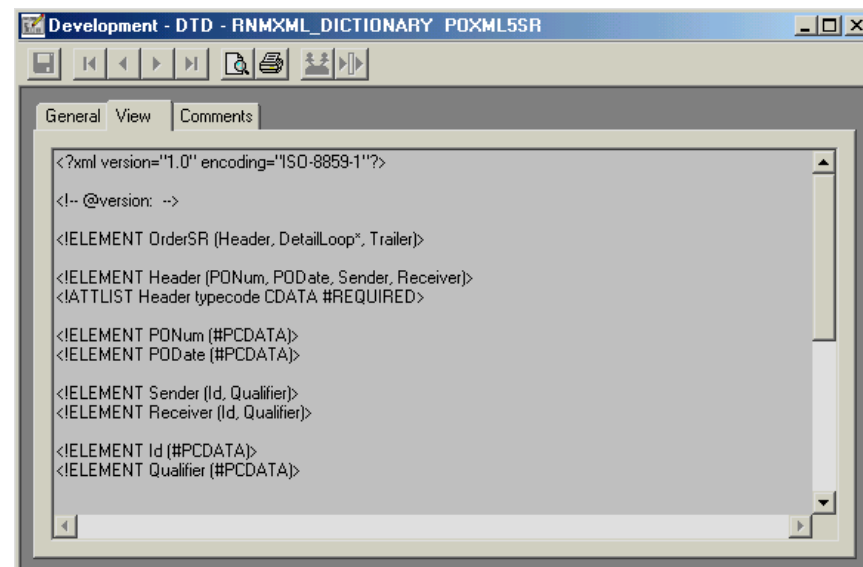


4. Complete any of the optional fields in the editor.

Click  for field descriptions.

5. Click Save on the tool bar to save the DTD.
6. Click the View tab on the DTD View Window.

The View tab page displays, showing the contents of the DTD.



EDI Standards

EDI standards provide a common document layout that trading partners use to exchange data between their computer applications. In essence, EDI standards provide the building blocks for electronic versions of common business documents.

DataInterchange translates data from one document layout to another. Commonly, the source document layout describes a document from a business application, which is translated into an EDI standard transaction for transmission to a trading partner. Conversely, DataInterchange commonly translates data received in an EDI standard transaction into a format used by a business application. When you install DataInterchange, you receive copies of EDI standards currently approved by the primary EDI standards organizations. You can also download EDI standards from the DataInterchange Web site at <http://edi.services.ibm.com/datainterchange>. For instructions on how to install EDI standards, see “Installing EDI standards” on page 16.

In order to begin exchanging documents with a trading partner using an EDI standard transaction, you must select an EDI standard transaction that corresponds to the information you want to send or receive. Ideally, you and your trading partner can agree on an EDI standard transaction that requires no customization to meet your needs, but this is not always possible. Imagine, for example, that you and your trading partner need to exchange specific information that is not included in existing EDI standards. DataInterchange Client allows you to customize currently approved EDI standards so that they will fit your needs.



ATTENTION: When altering EDI standards, you should work in close partnership with your trading partners. If you customize EDI standards without informing your trading partners of the changes, they may not be able to process the transactions you send.

DataInterchange Terminology Note

The terms “transaction set” in ANSI ASC X12 and “message” in UN/EDIFACT are equivalent to “transaction” in DataInterchange.

About standards

An EDI standard structures data into two basic categories: envelopes and transactions.

Envelopes

Envelopes are made up of the control structures that “wrap” data for communications to trading partners. UN/EDIFACT refers to envelope standards as service segments, and ASC X12 refers to them as envelope interchange control segments. Envelope standards also specify the default delimiters used in the EDI standard data, such as the data element delimiter, subelement delimiter, and segment delimiter. See “Accessing EDI standards editors” on page 192.

Transactions

Transactions correspond to business documents such as purchase orders or invoices. An EDI standard contains one definition for each unique segment and data element that occur in a transaction. These segments and data elements are then used in as many transaction sets as necessary.

A transaction is comprised of the following components:

EDI standard dictionary: An EDI standard dictionary contains information about all of the transaction sets, segments, and data elements that comprise the specific version and release of an EDI standard. For detailed information about a particular EDI standard, consult the appropriate EDI standards manuals.

When you install DataInterchange, you receive a copy of EDI standard dictionaries currently approved by the primary EDI standards organizations. For information on how to create dictionaries, see “Using the EDI Standard Dictionary Editor” on page 193.

Transaction set: Transaction sets represent business documents such as invoices or purchase orders. Transaction sets are called messages in UN/EDIFACT and transactions in DataInterchange. A transaction set contains segments. These segments are sometimes grouped into tables representing heading information (such as billing address), detail information (such as line items from a purchase order), and trailing information (such as totals). For more information on how to create or edit transactions, see “Using the EDI Standard Transaction Editor” on page 195.

Segment: A transaction set is composed of segments. In essence, each line of a business document corresponds to a segment in the EDI transaction set. Segments begin with a segment identifier assigned by the EDI standard. They are either mandatory, conditional, optional, or floating (floating segments may display anywhere in the transaction). All segments except for floating segments display in a fixed sequence for a given transaction.

Segments may repeat within a transaction up to the number of times specified by the EDI standard. Groups of segments, such as the group which makes up a name and address, may form a loop. Loops are identified by a loop ID. Entire loops may be repeated in succession up to the number of times specified by the EDI standard. For information on how to create or edit segments, see “Using the EDI Standard Segment Editor” on page 197.

Data Element: A segment is composed of data elements and composite data elements, which represent the individual units of data found in business documents, such as quantity ordered or unit price. Data elements display in a sequence specified by the EDI standard and are separated by a delimiting character, such as an asterisk. They have a minimum and maximum length, and are either mandatory, conditional, or optional.

DataInterchange Client also supports composite data elements. Composite data elements are composed of a group of logically related simple data elements. A Composite Unit of Measure, for example, is a combination of Unit of Basis for Measurement, Component, and Multiplier. Composite data elements are defined in the EDI standards.

All data elements must be of a data type prescribed by the EDI standard, such as date, time, and alpha-numeric, and identifiers, such as data type ID. Data elements may have to contain one of a specific set of codes if prescribed by the EDI standard. The EDI standard specifies the list of acceptable codes, which you can customize. For information on how to create or edit data elements, see “Using the EDI Standard Data Element Editor” on page 199.

Code Lists: A code list is a list of acceptable values for segments or data elements that can only contain certain values. If you include a segment or data element that can only contain certain values in the transaction set you are creating, you should enter all acceptable values into a code list.

When DataInterchange processes the transaction, it references the code list and checks the value of a field against it. If the field contains a value that does not display in the code list, DataInterchange returns a processing error. For information on how to create code lists, see “Using the Code List Editor” on page 202.

DataInterchange Terminology Note

Code Lists are called Validation Tables on the DataInterchange Host.

Using the EDI standards editors

Use the EDI standard editors to create or maintain the various components that make up an EDI standard. Although you can use the DataInterchange Client standard editors to create a completely EDI new standard, most users are not likely to do that.

The EDI standard editors are most often used to modify existing EDI standards to meet a company’s needs. This section provides a generic description of the procedures for using the EDI standards editors.

Use the EDI Standards List window to gain access to the EDI standards component editors. Each component editor corresponds to a tab on the EDI Standards List window, as follows:

- The EDI Standard Dictionary tab provides access to the EDI Standard Dictionary List window and Dictionary Editor window.
- The Transactions tab provides access to the Transactions List window and the EDI Standard Transaction Editor window.
- The Segments tab provides access to the Segments List window and EDI Standard Segment Editor window.
- The Data Elements tab provides access to the Data Elements List window and EDI Standard Data Element Editor window.
- The Code Lists tab provides access to the Code Lists List window and Code List Editor window.

Accessing EDI standards editors

You access all the EDI standard component editors in essentially the same way. All the editors display on tabs in the EDI Standards List window, as follows.

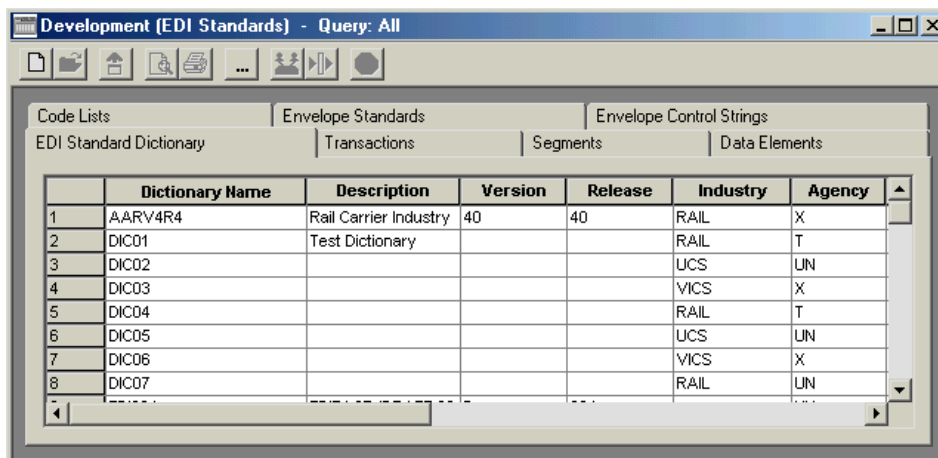
◆ To access an EDI standards editor:

1. Click EDI Standards on the DataInterchange Client Navigator bar.

The EDI Standards List window, which contains tabs for the EDI standards components, displays.

2. Click the tab of the EDI standard component you wish to work with.

The list window for that component displays.



	Dictionary Name	Description	Version	Release	Industry	Agency
1	AARV4R4	Rail Carrier Industry	40	40	RAIL	X
2	DIC01	Test Dictionary			RAIL	T
3	DIC02				UCS	UN
4	DIC03				VICS	X
5	DIC04				RAIL	T
6	DIC05				UCS	UN
7	DIC06				VICS	X
8	DIC07				RAIL	UN

This window displays a list of existing items in an EDI standards component. A list of EDI Standard Dictionaries is shown above. Each row contains information about an item; each column contains data stored in that item. Information in the columns displays in fields in the editor window. The list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to “Creating a query” on page 287.

3. To view an item or to add or change information in an item, double-click the row of the item you wish to work with.

The editor window displays. You add information or make changes to the EDI standard item using its editor, as described in the following sections.

Following are detailed procedures for creating EDI standards components. For information on viewing, copying, editing, renaming, deleting, and printing components, see “Performing common file management tasks” on page 35. For information on exporting components, see “Exporting” on page 52. (EDI Standard dictionaries, EDI standard transactions, and code lists are the only EDI standards components that can be exported.) For information on using the grid editors that display in some EDI standards editors, see “Using editor window grids” on page 38.

The EDI standards component editors are described in the following sections in the order in which you use them when creating an EDI standard from scratch.

Using the EDI Standard Dictionary Editor

An EDI standard dictionary essentially is a name within which other related components are grouped.

Following are detailed procedures for creating a new dictionary. For information on viewing, copying, editing, renaming, deleting, and printing dictionaries, see “Performing common file management tasks” on page 35. For information on exporting dictionaries, see “Exporting” on page 52.

Creating an EDI Standard dictionary

Create a new dictionary when you want to create your own customized EDI standard.

◆ To create a new dictionary:

1. At the EDI Standard Dictionary List window, click New on the tool bar.

The EDI Standard Dictionary Editor window displays with the General tab in front and the fields blank.

The screenshot shows the 'EDI Standard Dictionary Editor' window. The 'General' tab is active, displaying several input fields: 'Dictionary Name' (with a red asterisk indicating it's required), 'Envelope Type' (a dropdown menu), 'Description', 'Purpose', 'Version', 'Release', 'Industry Code' (a dropdown menu), and 'Agency Code' (a dropdown menu). At the bottom, there is a 'List Of' section with four buttons: 'Transactions', 'Segments', 'Composite Elements', and 'Simple Elements'.

2. Type a name in the Dictionary Name field.

The name displays in all capital letters. You cannot type spaces within the name.

3. Select an Envelope Type from the Envelope Type drop-down list. The available choices are:

E	EDIFACT delimiter and envelope definitions
F	Reserved - denotes an EDI standard created as part of the Fixed-to-Fixed mapping process
I	ICS delimiter and envelope definitions
L	Reserved - indicates the EDI standard was created using the DataInterchange DTD Conversion Utility
T	UNTDI delimiter and envelope definitions
U	UCS delimiter and envelope definitions
X	X12 delimiter and envelope definitions



NOTE: Once you save your dictionary, the Transactions, Segments, Composite Elements, and Simple Elements buttons in the List Of box become available. Click those buttons to display list windows that contain the components associated with this dictionary. When you first create a dictionary, the lists are empty.

4. Enter a more complete description of the EDI Standard Dictionary in the Description field.

5. Enter the version and release of the dictionary.

6. Select an Industry Code from the drop-down list provided; if a list is not available, you can type a name in the list. The available codes are:

RAIL	Association of American Railroads
UCS	Uniform Communication Standard
VICS	Voluntary Inter-Industry Communications Standard

7. Select an Agency Code from the drop-down list provided; if a list is not available, you can type a name in the list. The available codes are:

T	TDCC, ODETTE
UN	UN/EDIFACT
X	ASC X12, RAIL, UCS, VICS

8. Click Save on the tool bar to save the dictionary.

◆ **To view lists of EDI standard components in the current dictionary:**

1. Click the button in the List Of group box corresponding to the component you wish to view.

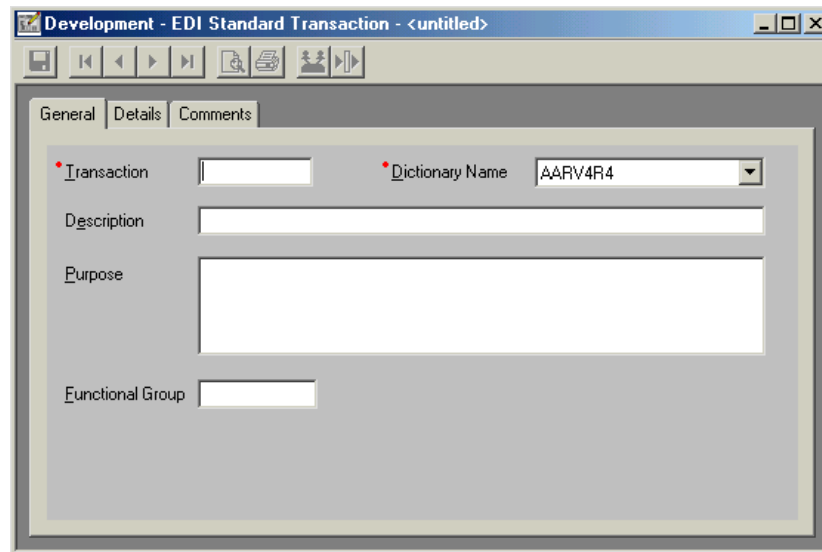
A list window displaying EDI standard components associated with this dictionary displays:

- The Transactions button displays the EDI Standard Transactions List window.
- The Segments button displays the EDI Standard Segments List window.
- The Composite Elements button displays the EDI Standard Data Elements List window showing only composite data elements.
- The Simple Elements button displays the EDI Standard Data Elements List window showing only simple data elements (data elements that are not composite data elements).

2. You may open any item in the list by double-clicking on it.

Using the EDI Standard Transaction Editor

The EDI Standard Transaction Editor allows you to define and structure the components that make up a transaction.



The EDI Standard Transaction Editor window contains three tabs: General, Details, and Comments. Use the:

- General tab to enter and change transaction properties, such as name and description.
- Details tab to add or change the usage of segments associated with the selected transaction.
- Comments tab to type any comments you wish about the selected transaction.

Following are detailed procedures for creating a new transaction. For information on viewing, copying, editing, renaming, deleting, and printing transactions, see “Performing common file management tasks” on page 35.

Creating a transaction

Create a new transaction when the transactions shipped as part of the EDI standards do not meet your business needs. This editor also modifies existing transactions.

◆ To create a new transaction:

1. At the EDI Standards Transactions List window, click New on the tool bar.

The EDI Standard Transaction Editor window displays with the General tab in front and the fields blank.

2. Type a name in the Transaction field.


The name displays in numbers and capital letters. You cannot type spaces within the name.

You can enter a more complete description of the transaction in the Description field, and a brief summary of the transaction's purpose in the Purpose field.

3. Select the dictionary in which you want the transaction to occur through the Dictionary drop-down list. This is a required field, as indicated by the red dot.
4. Enter a Functional Group, such as IN for invoice.
5. Click the Details tab to enter information on the segments and data elements contained in this transaction.

	Table	Pos	Segment	Req Des	Max Use	>1	Loop ID
1	1	20	BAX	M	1	<input type="checkbox"/>	0
2	1	30	AES	M	1	<input type="checkbox"/>	0
3	1	35	YNQ	M	1	<input type="checkbox"/>	0
4	1	40	N9	O	10	<input type="checkbox"/>	0
5	1	50	QTY	O	5	<input type="checkbox"/>	0
6	1	55	MEA	O	10	<input type="checkbox"/>	0
7	1	60	AEI	O	16	<input type="checkbox"/>	0
8	1	70	EI	M	1	<input type="checkbox"/>	EI 50
9	1	80	QTY	O	20	<input type="checkbox"/>	EI 0

6. Complete the optional fields you need.

Click  for field descriptions.

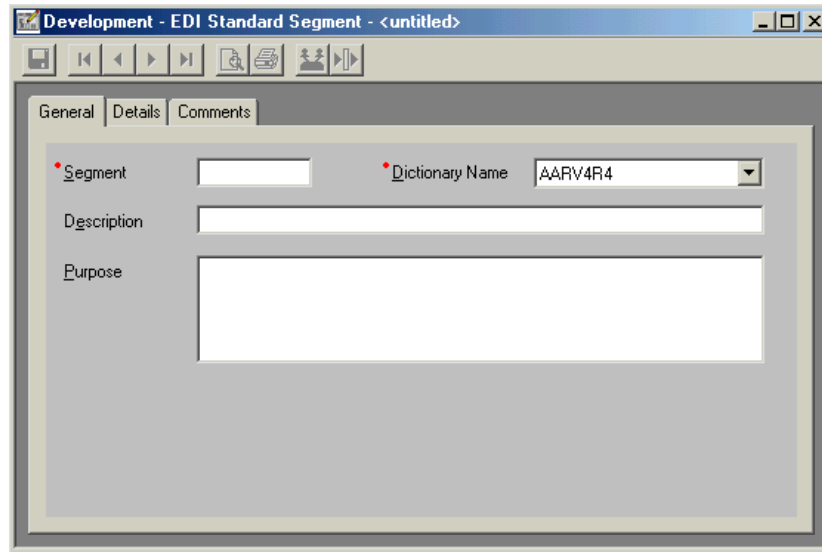
7. Use the Details tab to add and delete the transaction's associations with segments. The Details tab is also used to maintain the properties of each of those associations.

For instructions on working with the grid editor, see "Using editor window grids" on page 38.

8. When you have completed entering information, click Save on the tool bar to save the transaction.

Using the EDI Standard Segment Editor

Use the EDI Standard Segment Editor window to enter new segments into an EDI standard or to edit existing segments. From the Segments Editor, you can add or edit the usage of data elements in segments.



The Segments Editor window contains three tabs: General, Details, and Comments. Use the:

- General tab to name the segment and select its dictionary.
- Details tab to add or change data elements associated with the selected segment.
- Comments tab to type any comments you wish about the selected segment.

Following are instructions for creating a new segment. For information on viewing, copying, editing, renaming, deleting, and printing segments, see “Performing common file management tasks” on page 35.

Creating a segment

Create a new segment when business needs require one.

◆ To create a new segment:

1. At the EDI Standard Segments List window, click New on the tool bar.

The EDI Standard Segment Editor window displays with the General tab in front and the fields blank.

2. Type a name in the Segment field.

The name displays in all capital letters. You cannot type spaces within the name.


If you wish, you may enter a more complete description of the segment in the Description field, and a brief summary of the segment's purpose in the Purpose field.

3. Select the dictionary in which you want the segment to display through the Dictionary drop-down list. This is a required field.

Click the Details tab to enter information on the data elements contained in this segment.

	Pos	Data Element	Req Des	Max Use	Ele
1	1	40	M	0	Equipment Descrip
2	2	380	M	0	Quantity
3	3	1073	M	0	Yes/No Condition c
4					

4. Complete the optional fields you need.

Click  for field descriptions.

For instructions on working with the grid editor, see “Using editor window grids” on page 38.

5. Use the Details tab to add and delete the segment's associations with data elements and composite data elements. The Details tab is also used to maintain the properties of each of those associations.

For instructions on working with the grid editor, see “Using editor window grids” on page 38.

6. When you have completed entering information, click Save on the tool bar to save the segment.

Using the EDI Standard Data Element Editor

Use the EDI Standard Data Element Editor window to enter new data elements into a standard or to edit existing data elements.

The Data Elements Editor window contains three tabs: General, Details, and Comments. Use the:

- General tab to name the data element and select its dictionary.
- Details tab to add or change the component data elements associated with a composite data element.
- Comments tab to type any comments you wish about the selected data element.

Following are detailed procedures for creating a new data element. For information on viewing, copying, editing, renaming, deleting, and printing data elements, see “Performing common file management tasks” on page 35.

Creating a data element

Create a new data element when business needs require one.

◆ To create a new data element:

1. At the EDI Standard Data Elements List window, click New on the tool bar.

The EDI Standard Data Element Editor window displays with the General tab in front and the fields blank.

2. Type a name in the Data Element field.


The name displays in all capital letters. You cannot type spaces within the name.

If you wish, you may enter a more complete description of the data element in the Description field, and a brief summary of the data element’s purpose in the Purpose field.

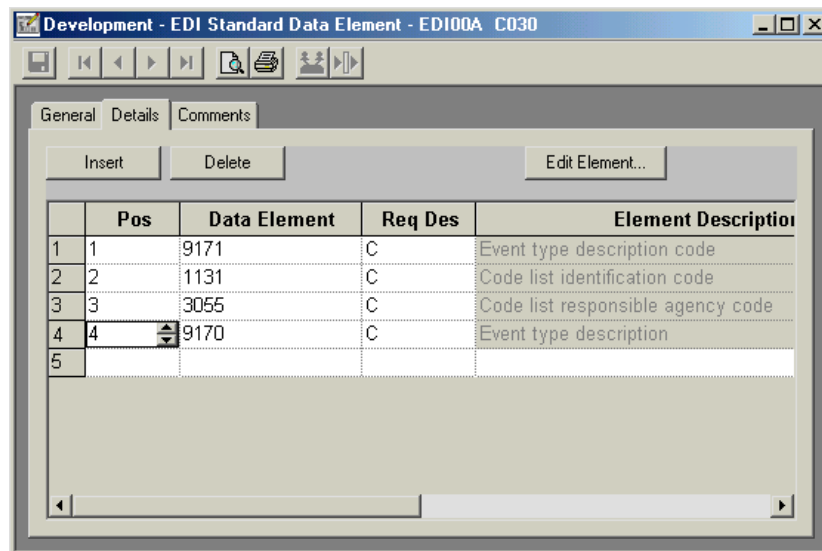
3. Select the dictionary in which you want the data element to display through the Dictionary drop-down list. This is a required field, as indicated by a red dot.
4. Select the data element's data type from the drop-down list in the data type field. This is a required field. The data types are explained in Table 63 on page 201.
5. Enter the minimum and maximum length of the data element in the Min Length and Max Length fields. These are required fields.
6. If the data element you are creating can only contain certain values, select the code list which identifies acceptable values for the data element you are creating from the drop-down list that displays in the Code List field.

If none of the existing code lists specify the acceptable values for your data element, you may create a new code list. See “Using the Code List Editor” on page 202.

7. Complete the optional fields you need.

Click  for field descriptions.

8. The Details tab is only available when the data type is CD (composite data element). If you are creating a composite data element, select CD in the data type drop-down list. Then click on the Details tab to identify the component data elements that will be associated with this composite data element and the properties of those associations.



The screenshot shows the 'Development - EDI Standard Data Element - EDI00A C030' window. The 'Details' tab is active, displaying a table with the following data:

	Pos	Data Element	Req Des	Element Description
1	1	9171	C	Event type description code
2	2	1131	C	Code list identification code
3	3	3055	C	Code list responsible agency code
4	4	9170	C	Event type description
5				

For instructions on working with the grid editor, see “Using editor window grids” on page 38.

9. When you have completed entering information, click Save on the tool bar to save the data element.

Table 63. Data Types

Data Type	Name	Description
A	Alphabetic	Alphabetic characters up to the length of the field.
AN	Alphanumeric	You can use any combination of characters in the ALPHANUM code list, up to the length of the field.
CD	Composite data element	A data element with data type CD and data element ID beginning with a C by convention for EDI standard composite data elements and S for envelope composite data elements. The component data elements are defined using the Details tab in the EDI Standard Data Element Editor.
CH	Character	Any combination of characters up to the length of the field.
DT	Date	Date format <i>yyyymmdd</i> , where <i>yyyy</i> is the year, <i>mm</i> is the month (01-12), and <i>dd</i> is the day.
ID	Identifier	A data element which usually has a code list for the valid values for the data element. For example, data element UM, unit of measure, has data type ID, and the valid values for this data element are listed in the UMCODES code list. The table name is the same as or starts with the data element ID.
IV	Incrementing value	A data element, such as a message reference number, that starts at 1 and increases by 1 for each usage.
N	Numeric	Any combination of 0-9 and an optional sign (+ or -). The length includes the sign. When mapping data elements defined as data type N in UN/EDIFACT EDI standards, replace it with data type R (because data type N in the ASC X12 EDI standards is the same as data type R in UN/EDIFACT EDI standards).
Nn	Numeric	Numeric data with N places to the right of an implied decimal point. The only acceptable characters are the digits 0 through 9. N is the same as N0. For example, if the data type is N2, the value 123 means 1.23. A sign (+ or -) is optional. Positive is assumed if no sign is present. The length does not include the sign. The data type should be used when defining data elements defined as data type N in UN/EDIFACT EDI standards.
PW	Password	A password used in the interchange or functional group header.
R	Real	Numeric data that requires a decimal point for fractional values. The decimal point is optional for integers. A sign (+ or -) is optional. Positive is assumed if no sign is present. The length does not include the decimal point and sign. This data type should be used when defining data elements defined as data type N in UN/EDIFACT EDI standards.
Rn	Real	Signed or unsigned numeric data with a minimum of n significant decimal places. On sending, at least n decimal places are generated. On receiving, the decimal places are the same as data type R. A sign (+ or -) is optional. Positive is assumed if no sign is present. The length does not include the decimal point and sign.

Data Type	Name	Description
TM	Time	Time format is <i>hhmm</i> or <i>hhmmss</i> , depending on the length of the data element, where <i>hh</i> is the hour, <i>mm</i> is the minutes, and <i>ss</i> is the seconds. The time format uses a 24-hour clock, where the hour is specified as 01 to 24 for EDIFACT and 00 to 23 for X12.

Using the Code List Editor

A code list is a list of acceptable values for data elements which can only contain certain values.

Following are detailed procedures for creating a new code list. For information on viewing, copying, editing, renaming, deleting, and printing profiles, see “Performing common file management tasks” on page 35. For information on exporting profiles, see “Exporting” on page 52.



NOTE: Code lists can also be specified in data format fields (when used with send and receive maps) and can be used in all maps.

Creating a code list

Create a new code list when you create a data element which can only contain certain values. A code list can also be created if you wish to restrict the values of any data item during translation. In this case, the code list is specified during mapping.

◆ To create a new code list:

1. At the list window for Code Lists, click New on the tool bar.

The Code List Editor window displays with the General tab in front and the fields blank.

2. Type a name in the Code List field.

The name displays in numbers and capital letters. You cannot type spaces within the name.

If you wish, you may enter a more complete description of the code list in the Description field.

3. Choose a Data Type from the drop-down list that displays for the Data Type field. Your choices are :

CH Characters

R Real numbers

4. Choose the length of entries from the drop-down list that displays for the Length of Entries field. Entries may consist of one to thirty-five characters or numbers.
5. In the grid at the bottom of the Code List General tab, enter the acceptable entries that will be part of this code list in the Entry column, then enter a description of what this entry signifies in the Description column. For instructions on working with the grid editor, see “Using editor window grids” on page 38.
6. Click Save on the tool bar to save the code list.

Using the Envelope Standard Editor

An envelope standard is a name under which components of an EDI envelope standard are grouped.

Following are detailed procedures for editing an envelope standard. For information on viewing, copying, editing, renaming, deleting, and printing profiles, see “Performing common file management tasks” on page 35.



NOTE: An envelope standard cannot be created - it must be imported. Obtain envelope standards from the DataInterchange Web site at <http://edi.services.ibm.com/datainterchange>.

Editing an envelope standard

Edit existing envelope standards when you want to change the default values of envelope fields for envelopes being created during translation.

◆ To edit an envelope standard:

1. Select the envelope standard you want to edit from the list displayed on the Envelope Standards List window.

The Envelope Standard Editor window displays with the General tab in front and the Dictionary field filled in with the name of the selected dictionary.

2. Edit the description of the envelope standard in the Description field.



NOTE: Click the Segments, Composite Elements and Simple Elements buttons to display list windows that contain the components associated with this envelope standard.


3. Enter the version and release of the envelope standard.
4. Select an Industry Code from the drop-down list provided; if a list is not available, you can type a name in the list. The available codes are:

RAIL	Association of American Railroads
UCS	Uniform Communication Standard
VICS	Voluntary Inter-Industry Communications Standard

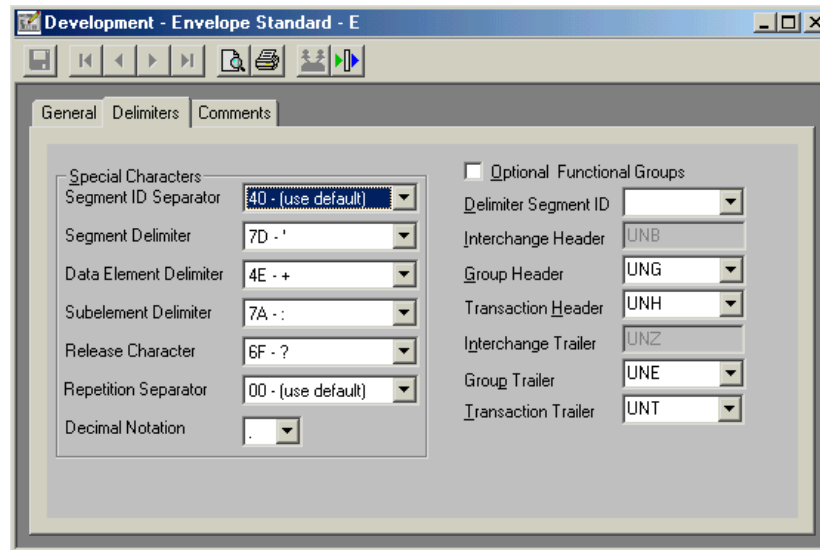
5. Select an Agency Code from the drop-down list provided; if a list is not available, you can type a name in the list. The available codes are:

T	TDCC, ODETTE
UN	UN/EDIFACT
X	ASC X12, RAIL, UCS, VICS

6. Click the Delimiters tab to add the envelope specific information.
7. Complete the fields on the Delimiters tab.

Click  for field descriptions.

8. Click Save on the tool bar to save the changes to the envelope standard.



◆ **To view lists of EDI Standard components in the current envelope standard:**

1. Click the button in the List Of group box corresponding to the component you wish to view.

A list window displaying EDI standard components associated with this envelope standard displays:

- The Segments button displays the EDI Standard Segments List window.
- The Composite Elements button displays the EDI Standard Data Elements List window showing all composite data elements in this envelope standard.
- The Simple Elements button displays the EDI Standard Data Elements List window showing all simple data elements (those that are not composite data elements) in this envelope standard.

2. You may open any of the components by double-clicking them.

Using Envelope Control String list window

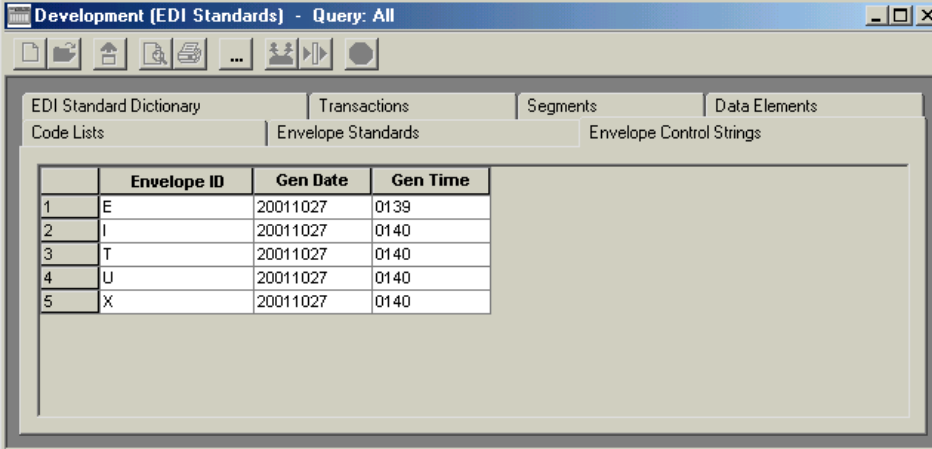
An envelope control string helps improve performance of the translator when envelopes are prepared. Envelope control strings must be compiled after they are installed and after any change is made. If stand-alone mode is being used on the client, the envelope control string must be exported from the Client and imported into the Host. In client/server mode, the envelope control string will be automatically stored on the Host database during the compile.

The Envelope Control String List window displays a list of compiled envelope standards and shows the compilation information about each.

◆ To compile an envelope control string:

1. Select the envelope control string to be compiled from the list displayed on the Envelope Control Strings List window. Alternately, select the envelope standard on the Envelope Standards List window.
2. Click Compile Control String(s) on the tool bar of the list window.

While compiling, DataInterchange Client checks for errors in the changes you made. Error messages are written to the event log.



	Envelope ID	Gen Date	Gen Time
1	E	20011027	0139
2	I	20011027	0140
3	T	20011027	0140
4	U	20011027	0140
5	X	20011027	0140

Creating a map

Mapping is the process by which you relate input and output documents. Through mapping, you specify the relationships between the internal documents used by your applications and the external documents that you exchange with trading partners. Maps are then compiled into control strings which allow DataInterchange Translation Servers to transform documents between the internal and external formats.

DataInterchange Client is designed to make mapping easy. In essence, you select elements in your source document, drag them onto elements in the target document, and drop them. You can then apply DataInterchange's specialized mapping functions as required.

Getting started

The following is an outline of the steps required to create a map. The same basic steps work for maps you create to translate data that will be sent to trading partners and for maps you create to translate data that will be received from trading partners. In both cases, you are mapping data from one document format to another.

1. Obtain the layout of the source document to be used in the translation.

This may be a data format, EDI standard transaction, or an XML DTD. You may need to obtain the layout from your trading partner. Once you have the source document layout (called the source document definition), put that layout into DataInterchange.

If the source document is data in a proprietary format, then create a data format in DataInterchange. For information on creating a data format, see “Creating a data format” on page 146.

If the source document is an EDI standard transaction, study the layout of the transaction. EDI standards are broad. You can use a number of methods to map onto an EDI standard transaction. Any number of organizations are likely to use different methods to map the same data to the same transaction. Once you are comfortable with the transaction, make sure the transaction is defined in DataInterchange. For instructions on how to install EDI standards, see “Installing EDI standards” on page 16.

If the source Document is in XML format, obtain the corresponding DTD and import it into DataInterchange. This is done using import. For information about importing a DTD file, see “Importing a DTD file” on page 57.



ATTENTION: If you change the meta-data for a document, it affects each map which uses that document as either a source or target document.

2. Obtain the layout of the target document to be used in the translation.

This may be a data format, EDI standard transaction, or an XML DTD. You may need to obtain the layout from your trading partner. Once you have the target document layout (called the target document definition), put that layout into DataInterchange. Refer to step 1 for additional information.

3. Decide how source data is handled.

When you are creating a map to send a document to a trading partner, study the target document definition and decide how you want to use it to pass your source data.

When you are creating a map that will receive a document from a trading partner, study your trading partner's document definition to see what data your partner is sending and how you will handle it.

4. Compare your source document definition to the target document definition.

When you map, you are associating elements in one document definition with elements in another document definition.

Study the source document definition for which you are creating a map. Make note of which elements in the source document definition correspond to which elements in the target document definition. Then decide how you are going to map the two document definitions. For example, you may use a loop on an EDI standard transaction to handle repeating records in a data format.

5. Map the data elements in each segment.

Use the Map Editor to drag elements between the source and target document definitions. You may notice that some elements in target document definition may not occur in the source document definition. If they are required, you need to fill them with data on the outbound side. DataInterchange can fill many such elements using special handling options, literals, mapping commands and accumulators.

Special handling options and mapping commands allow you to perform such functions as translating date formats, converting values supplied by a trading partner to values you require, and validating the contents of data elements.

Accumulators and variables allow you to perform such actions as counting segments in a transaction for later placement in a transaction's trailer record.

Literals and mapping commands are a means of supplying data to data elements or fields, such as dates and times.

6. Specify how loops and repeating segments are handled.

Document definitions can include repeating elements. Examples of repeating elements are loops in a data format, records and structures. Repeating elements in data formats include loops, records, and structures. You must specify how DataInterchange should handle the repeating elements.

7. Associate the map with a trading partner using a usage or map rule.

After you have created a map, you must associate it with trading partners. Because you can use the same map for multiple trading partners, each usage or rule identifies one or more trading partners that use the map. The usage or rule can supply values specific to that trading partner or group of trading partners, such as:

- Validation and error levels.
- Unique values for enveloping that override the default values.
- The ddname of the file to which output data should be written.

For more information, see “Understanding processes and rules” on page 132 and “Understanding minimal trading partners” on page 133.

8. Compile a map.

The DataInterchange translator uses a control string in its processing, not the map itself. When you complete your mapping work, you must compile the map to create a control string. The compiler identifies any errors as it creates a control string.

Data transformation maps vs send and receive maps

Release 4.1 is a transitional release for the DataInterchange product from an EDI translator to a B2B Gateway. A key component of a B2B Gateway is any-to-any data transformation with particular emphasis on the XML and EDI syntaxes, since these are the most common syntaxes used for exchanging documents between businesses. DataInterchange 4.1 provides this any-to-any transformation capability and has a new any-to-any mapper which is used to define the transformations. The new any-to-any mapper differs fundamentally from the EDI mapper provided in previous releases, both because any-to-any mapping is more complex than data format-to-EDI mapping, and because the concepts of send and receive mapping into a single transformation mapping have been rationalized and simplified.

Because the new transformation map is sufficiently different from the send and receive maps of previous releases, some retraining will be required for those familiar with the previous releases of DataInterchange. In order to ease this period of transition, the send and receive maps from release 3.1 remain in the 4.1 version of DataInterchange. Current users can install DataInterchange 4.1, migrate all of their current maps, and begin using DataInterchange 4.1 in a familiar way. As time permits, they can familiarize themselves with the new data transformation map editor and start using it to create new maps. It is strongly recommended that current customers use the new data transformation map editor for all new maps, and use the send and receive map editors for maintenance of existing send and receive maps only.

New DataInterchange users should not create send and receive maps. Instead, they should use the data transformation map editor for all their maps, since it is the strategic mapping editor. Consequently, they can skip the chapters on creating send and receive maps and usages. Send usages and receive usages are the send and receive mapping equivalents of data transformation map rules.

Creating a data transformation map

See Chapter 22, “Data transformation mapping,” on page 211.

Creating a send or receive map

See Chapter 23, “Send and Receive mapping,” on page 239.

Data transformation mapping

Data transformation maps are one of the three supported map types in DataInterchange. A data transformation map is a set of mapping instructions that describes how to translate data from a source document into a target document. Both the source and target documents can be one of several supported document types.

Using the Map Editor

DataInterchange Client's map editor features a visual means for associating elements from a source document format with elements in a target document format. The map editor uses a split screen that permits you to create map associations.

The source document definition appears in the upper left corner of the map editor and displays the layout of the document that will be used as the source of the translation. The target document definition displays the layout of the document that will be created by the translation. It is displayed in the upper right hand corner of the Map Editor. The map editor displays a mapping commands window pane in its lower left hand corner. The mapping commands window pane displays the source document layout with mapping commands and comments inserted into it. There is also a variables window pane in the map editor. It is divided into three sub-panes that list all special variables, global variables, and local variables.

Starting the Map Editor

The Map Editor displays when you select a map from the Mapping List window, as follows.

◆ To start the Map Editor:

1. Click Mapping on the DataInterchange Client Navigator bar.

The Mapping List window displays.

	Map Name	Map Type	Compile Required	Description	Lock	Updated ▲
1	ADF-TO-EDI	Data Transform	No		No	10/22/2001 08
2	ALPH01	Send Map	Yes	send map	Yes - ad	10/22/2001 10
3	ALPH02	Data Transform	Yes	Test map	No	10/22/2001 10
4	ALPH03	Data Transform	Yes	dt map test	No	10/22/2001 10
5	ALPH04	Data Transform	Yes	dt test map	No	10/22/2001 10
6	ALPH05	Send Map	Yes	map	No	10/22/2001 10
7	MP1025	Send Map	Yes	send map	No	11/16/2001 09
8	EDI-TO-ADF	Receive Map	Yes	850	No	11/09/2001 09
9	EDI-TO-EDI	Data Transform	Yes	X12V4R2 850 to an	No	10/22/2001 10
10	MP004	Send Map	Yes	Send Map	Yes - ad	10/22/2001 10

This window displays a list of existing maps. All the different types of map are listed. Each row contains information about a component; each column contains data stored in that component. Information in the columns displays in fields in the editor window. The list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to “Creating a query” on page 277.

2. To view a map or to add or change its information, double-click the row of the map you want to work with.

The appropriate Map Editor window displays, with the Details tab in front. You add information or make changes to maps through its tabs and related dialog boxes, as described in the following sections.

Utilizing the Map Editor

The Map Editor allows you to show how data is to be translated from the source document to the target document. The editor works by displaying the source document definition on one side of the screen and target document definition on the other. The Map Editor allows you to associate components of your source document with components of the target document by dragging source components and dropping them into the correct locations in the target document definition. Components include simple elements, such as fields in a data format or data elements in an EDI standard transaction, and compound elements, such as loops and record in a data format and segments in an EDI standard transaction.

You can map the simple elements in the source document definition to the simple elements in the target document definition in any of these patterns:

- One simple element to one simple element
- Several simple elements to one simple element
- One simple element to several simple elements

The Map Editor window contains three tabs: General, Details, and Comments. Use the:

- General tab to enter and change map properties.
- Details tab to create and maintain the mapping commands.
- Comments tab to type any comments you wish about the selected map.

Following are detailed procedures for creating and editing maps. For information on viewing, copying, editing, renaming, deleting, and printing maps, see “Performing common file management tasks” on page 41. For information on exporting maps, see “Exporting” on page 58.

Creating a data transformation map

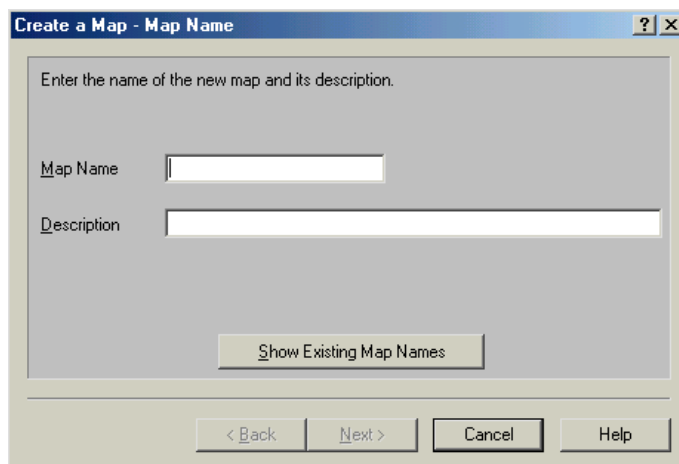
Create a new map when you want to translate a document from one format to another.

In the most basic sense, creating a new map consists of associating simple elements in a source document definition with simple elements in a target document definition. The following procedure shows how DataInterchange Client makes basic associations between simple elements.

◆ To create a new map:

1. In the Mapping List window, click New Document on the tool bar.

The Map Name wizard page displays in the Create a Map wizard.



The screenshot shows a dialog box titled "Create a Map - Map Name". It contains a text area with the instruction "Enter the name of the new map and its description." Below this are two input fields: "Map Name" and "Description". A button labeled "Show Existing Map Names" is positioned below the "Description" field. At the bottom of the dialog are four buttons: "< Back", "Next >", "Cancel", and "Help".

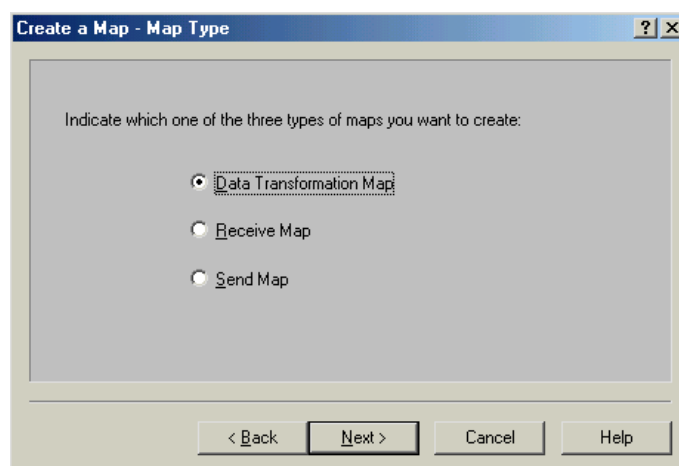
2. Type a map name in the Map Name field.

You may use both letters and numbers to identify your map. Letters display in capitals. You cannot type spaces within the name.

If you wish, you may enter a more complete description of the map in the Description field.

3. Click Next to continue.

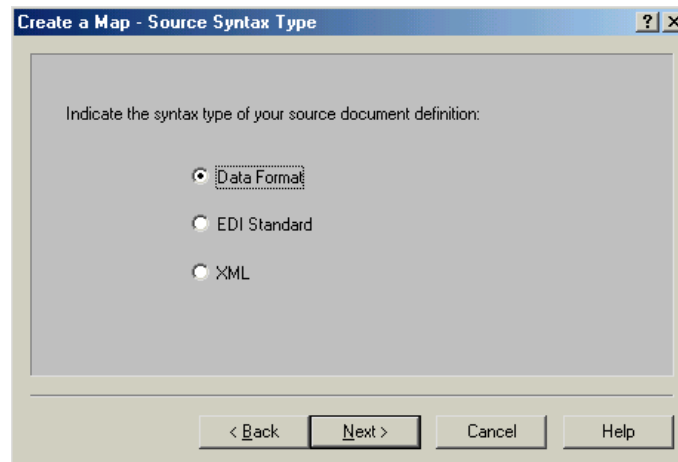
The Map Type wizard page displays.



The screenshot shows a dialog box titled "Create a Map - Map Type". It contains a text area with the instruction "Indicate which one of the three types of maps you want to create:". Below this are three radio buttons: "Data Transformation Map" (which is selected), "Receive Map", and "Send Map". At the bottom of the dialog are four buttons: "< Back", "Next >", "Cancel", and "Help".

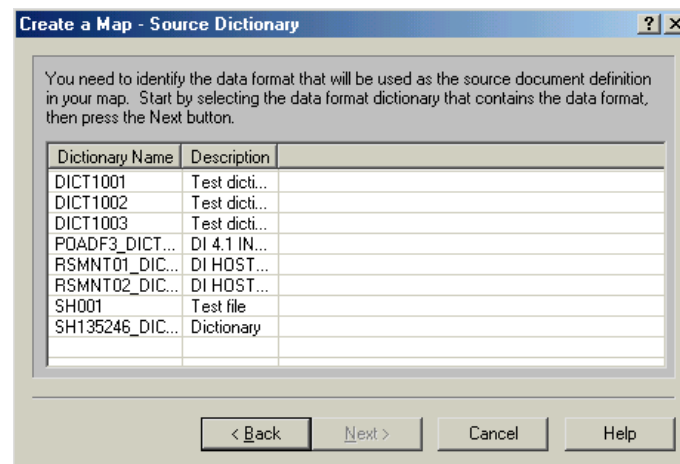
4. Click the appropriate radio button for a data transformation map, and then click Next.

The Source Syntax Type wizard page displays.



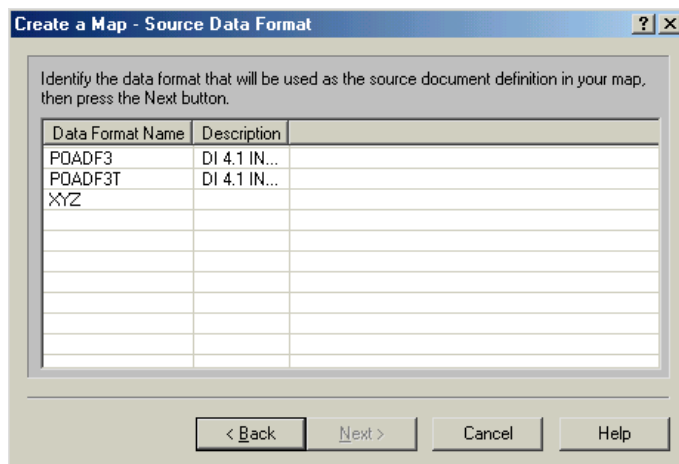
5. Select the radio button that indicates the syntax type of your source document definition.

The Source Dictionary wizard page displays with a list of the dictionaries for your specified syntax type.



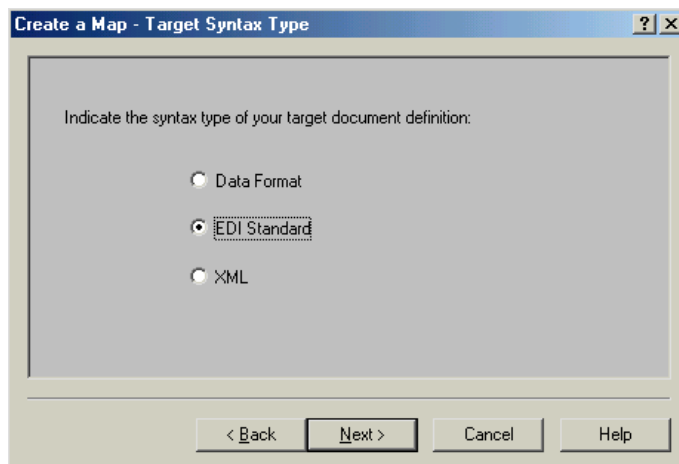
6. Select your source dictionary and click Next.

A wizard page displays with a list of the document definitions in your source dictionary. The wizard page is titled Source Data Format, Source EDI Standard Transaction, or Source XML DTD, depending on the Source Syntax Type previously selected.



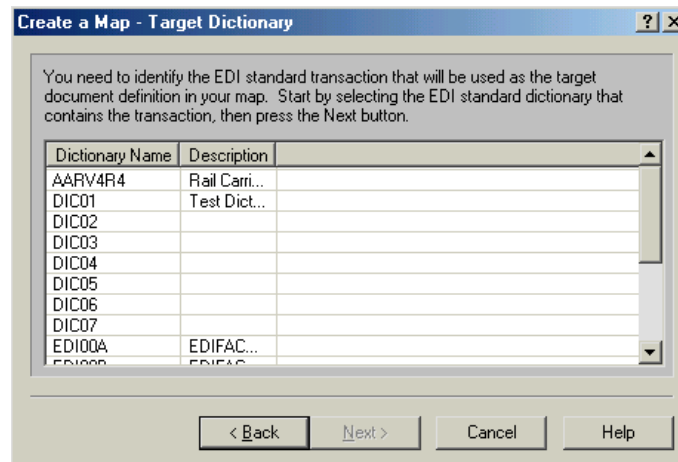
7. Select the data format, the EDI Standard transaction, or the XML DTD to be used as the source document definition in your map, and then click Next.

The Target Syntax Type dialog box displays.



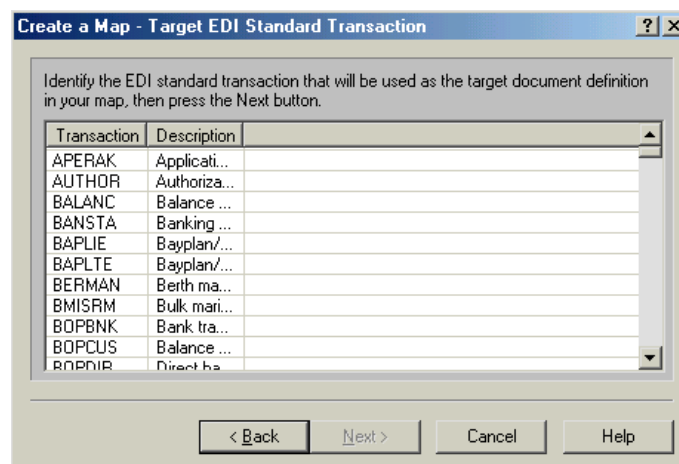
8. Select the syntax type of your target document, and then click Next.

The Target Dictionary wizard page displays with a list displays of the dictionaries for your specified syntax type.



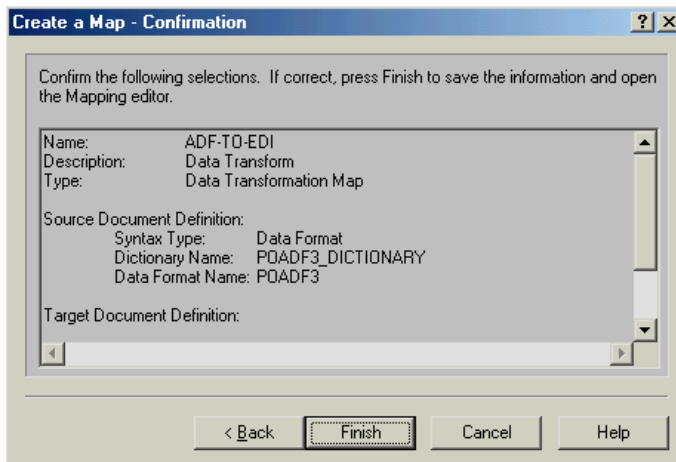
9. Select your target dictionary, and then click Next.

A wizard page displays with a list of the document definitions in your target dictionary. The wizard page is titled Target Data Format, Target EDI Standard Transaction, or Target XML DTD, depending on the Target Syntax Type previously selected.



10. Select the data format, the EDI standard transaction, or the XML DTD to be used as the target document definition in your map, and then click Next.

The Confirmation window displays.

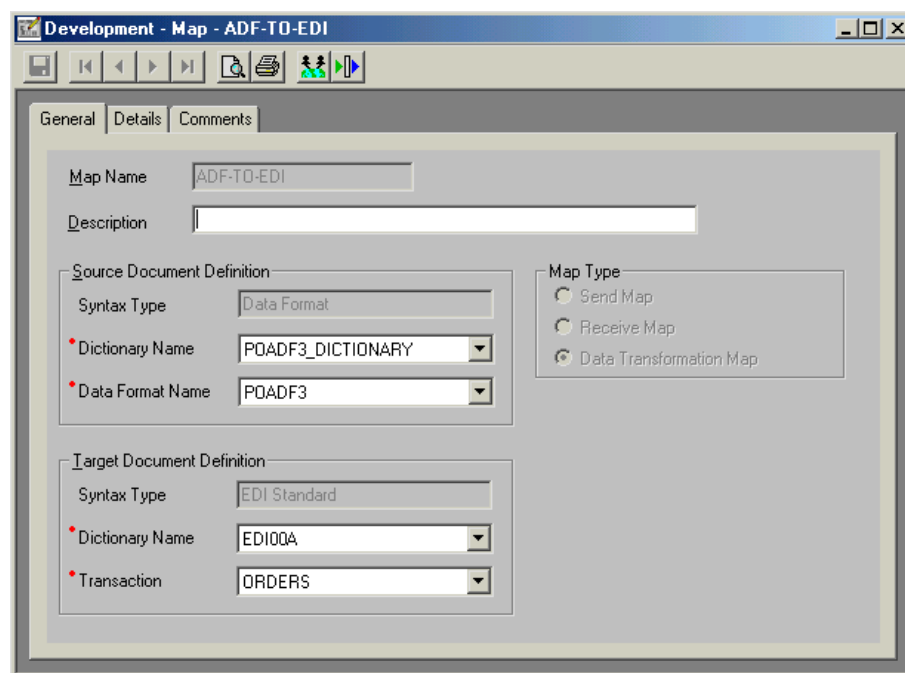


11. Confirm the selections displayed. If correct, click Finish to save the information.

After finishing the map information, the Map Editor displays with the Details tab opened.

- a. Click the General tab.

The General tab displays.



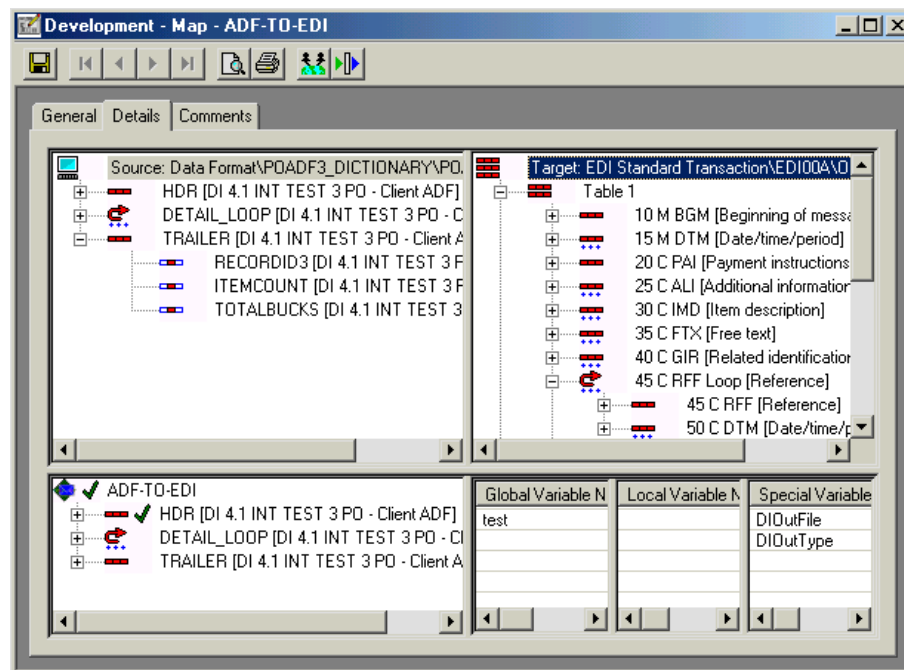
The Map Editor's General tab page contains general information about the map. It includes the map name, a brief description of the map and it identifies the source and target document definitions.

The name of the map is set when the map is created. It displays on the General tab page for informational purposes only. It can not be changed without using the rename action.

Use the Description field to provide a brief description of the map. The description can be up to 50 characters long.

Using the Map Editor

The Details tab in the Map Editor allows you to perform drag and drop mapping on your documents.



The top left pane in the window displays the source document definition, and the top right pane displays the target document definition. The lower left pane is the Mapping Command window pane, and the lower right pane is the variables window pane, which includes the lists for Global, Local, and Special Variables.

For more information about the Map Command window pane, see “Utilizing the Map Command window pane” on page 223.

For a list of mapping components and their associated graphics, see Table 64, “Symbols used in Data Transformation maps,” on page 222.

1. Click the plus (+) sign next to a compound element, such as a loop, segment, or record in the target document definition. Simple elements, such as fields and data elements, will not have a plus (+) sign next to them.

The compound element expands to show the compound and simple elements that comprise the compound element.

If you wish to close any expanded compound element, click the minus (-) sign.

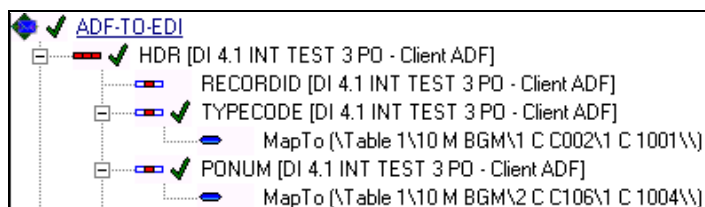
2. Click the element you wish to map on the top left side of the screen. While holding down the mouse button, drag it to the corresponding element in the target document definition on the top right pane.

When you have dragged the element to the right side of the screen over the element with which you want to associate it, that component becomes highlighted. Release the mouse button.

You can drag all simple elements and repeating compound elements. When you are dragging an element, holding it over a compound element will cause the compound element to expand after a few moments, if it is not already expanded. Dragging an element to any edge of the window pane will cause the window pane to automatically scroll up, down, left or right, depending on which edge you are near and whether the window pane can be scrolled in that direction.



NOTE: If the element does not become highlighted when you move the cursor on top of it, that means it is not a valid place to perform a drop. For example, you cannot drop a data format record onto a data element in an EDI transaction.



After you have mapped an element, a command is created in the mapping commands window pane and inserted into an appropriate position. A green check mark displays next to the mapped element in the mapping command window. A green check mark also displays next to its parent elements so that you know the parent element contains mapped components.



NOTE: This procedure presents a simple mapping situation in which you associate elements in the source document definition with elements in the target document definition. For anything but the most basic mapping associations, you must use DataInterchange's advanced mapping capabilities. For more information, see "Utilizing the Map Command window pane" on page 223.

3. Continue dragging and dropping elements in the source document definition onto elements in the target document definition until you have mapped all of the information required for this map.
4. Enter any general comments on the map in the Comments tab.
5. When you have completed mapping, click Save on the tool bar to save the map.

Editing a map

Edit a map when you have modified the associated target or source document definition, or when you need to add, delete, or change mapping commands within the map. You may also need to edit a map to meet specific requirements of a new trading partner.

◆ To edit a map:


1. In the Mapping List window, double-click the map you wish to edit.

The map displays in the Map Editor window with the Details tab in front.

Changes you made earlier to the map or its associated items display on the screen, as well as changes you made to the source or target document definition.

2. Create new associations between the source and target elements, or change existing ones.
 - To make an association between a new element in the source document definition and an element in the target document definition, drag the source element and drop it on the proper target element.
 - To edit an existing association, double-click the mapping command in the mapping command window pane. The Mapping Command Editor displays.
 - To delete an association, select the mapping command you want to delete and press the Delete key.
3. Change information as required in the General tab.
4. Click Save on the tool bar to save the map.

Table 64. Symbols used in Data Transformation maps

Symbol	Mapping	Data Format	EDI Standard	XML DTD
				
			EDI Standard Transaction	
			Table	
		Record	Segment	
		Repeating Record	Repeating Segment	
		Loop	Loop	
		Repeating Loop	Repeating Loop	
		Structure	Composite Data Element	Compound Element
		Repeating Structure	Repeating Composite Data Element	Repeating Compound Element
		Field**	Simple Data Element**	Simple Element**
			Repeating Simple Data Element**	
	Mapping Indicator			
	Mapping Command			
	Command Group			
	Comment			
	Comment Group			
	If, ElseIf, Else, EndIf			
	Qualify			
	Qualify (multi-occurrence)			

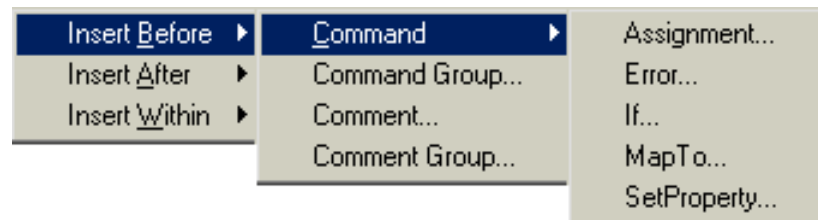
** Simple Elements - all others are Compound Elements

Utilizing the Map Command window pane

The Map Command window pane allows you to apply DataInterchange Client's advanced mapping capabilities.

◆ To apply advanced mapping capabilities:

1. Right-click an element in the Mapping Command window pane and follow the cascading menus.



2. If you want to insert a command, select where the command should be inserted, select Command, and then select the command to be inserted.

The Mapping Command Editor displays with a prototype of the command.



3. Edit the prototype to create the command you need.
4. Click Repeat if you want to add the command, and then create another similar to it.
5. Click OK when you have your last command ready.

Specifying qualification

Within your source document, certain elements and groups of elements can repeat. When you map an element that repeats within the source document (whether compound or simple element), you must tell DataInterchange which occurrence of the segment or loop you are using. This is called "qualifying" the segment or loop.

Terminology

The following terminology is important to know when using qualification.

- Simple element - a single value
- Compound element - a grouping of elements
- Element - refers to simple or compound element

In data formats, simple elements are fields within the data format. Compound elements include structures, records, and loops.

In EDI standards, simple elements are data elements or sub-elements. Compound elements include loops, segments, and composite data elements.

An XML element is a compound element. An XML attribute and an XML value are considered simple elements.

DataInterchange supports four types of qualification for data transformation maps. You can qualify by:

- Occurrence
- Multi-occurrence
- Value
- Expression

When you qualify by occurrence, your mapping is related to the position of data in a repeating sequence. When you qualify by multi-occurrence, your mapping is related to a specific structure or record in the source document definition and how it handles repeating segments in the target document definition. When you qualify by value, your mapping is related to the value that you receive in a simple element. When you qualify by expression, your mapping is based upon a condition.

Qualifying repeating simple and compound elements

DataInterchange allows you to qualify compound elements by single occurrence, multi-occurrence, value, and expression.

Qualify by occurrence when the order in which repeating data occurs in either the source or target document is important. For more information, see "Qualifying an element by occurrence" on page 225.

Qualify by multi-occurrence when you want to create multiple elements in the target to correspond to repeating elements in the source. For more information, see "Qualifying an element by multi-occurrence" on page 226.

Qualify by value when you want to specify data received in the source to trigger the creation or population of fields in your target. For more information, see "Qualifying an element by value" on page 226.

Qualify by expression when you need a more complex qualification. For more information, see “Qualifying an element by expression” on page 227.

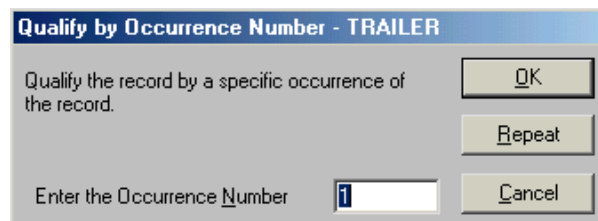
Qualifying an element by occurrence

Qualify an element by occurrence when a specific instance of the compound element must be mapped to a specific area of the target. For example, say that you are working with a source document that has an address compound element. The first occurrence of the address compound element has the send-to address, the second occurrence has the ship-to address. Map each occurrence to the appropriate target element using occurrence qualification.

◆ To qualify an element by occurrence:

1. Right-click the element in the Mapping Command window pane and select Qualify.
2. Select By Occurrence.

The Qualify by Occurrence Number dialog box displays.



3. Type in a number in the Enter the Occurrence Number field that corresponds to the occurrence number of the element you are mapping.

If you are mapping the second occurrence of an element, type 2 in the Enter the Occurrence Number field.

4. If you need to map additional occurrences, click Repeat.

A mapping command is created that qualifies the element by occurrence and then redispays the Qualify by Occurrence Number dialog box.

5. When you have specified the desired number of occurrences, click OK.

A mapping command is created that qualifies the element by occurrence, and the Qualify by Occurrence Number dialog box is closed.

Qualifying an element by multi-occurrence

Qualify an element by multi-occurrence when you need DataInterchange to create multiple instances of an element in the target based on multiple instances of an element in the source. For example, say you are working on an EDI standard transaction to data format map and you find that the PO1 segment in a purchase order repeats to handle multiple purchase-order line items. You need DataInterchange to create a separate record for each instance of the PO1 segment.

Consequently, you would qualify the PO1 segment by multi-occurrence. That way, DataInterchange will create as many line-item records in your application data as there are occurrences of PO1 in your trading partner's transaction.



NOTE: Multi-occurrence qualification finds repeating elements in the source document and creates corresponding repeating elements in the target document. Use multi-occurrence qualification when the number of repeating elements DataInterchange may need to create is unknown.

DataInterchange Client's mapping function automatically qualifies repeating elements. When you drop a repeating element from a source document definition onto a repeating element in the target document definition, the repeating element is automatically qualified by multi-occurrence; a MapTo command is added to the source compound element in the mapping command window.

The multi-occurrence command Default only displays when there are other existing qualifications for the same repeating element. When multiple qualifications exist and the multi-occurrence qualification is present, it will always display last. This indicates that mappings under the multi-occurrence qualification will be performed only when the specific instance of the repeating element is not resolved to one of the other existing qualifications.

◆ To qualify an element by multi-occurrence:

1. Drag a compound element onto a repeating element in the source document definition.

If there are no existing qualifications for the element, a MapTo command will be inserted under the corresponding repeating element in the Mapping Command window pane.

If there is an existing qualification for the repeating element, the Default command is inserted as the last qualification for the repeating element in the Mapping Command window pane.

Qualifying an element by value

Qualify an element by value when you want the value of data received in another simple element to drive DataInterchange's translation of a whole element.

For example, say you want to qualify the N1 loop with the value of BY in Element 98, which is the "Entity Identifier Code," received in a purchase order to create a buyer record. Further, say that you want the buyer's name to be mapped into the buyer record depending on the value in Element 98 of the N1 loop.

To handle that case, you would qualify the element by value. That way, DataInterchange will put specific information into the buyer records it creates from the purchase order depending on the name in each order's Entity Identifier Code.

◆ To qualify an element by value:

1. Right-click the repeating element in the Mapping Command window pane.
2. Right-click to select Qualify.
3. Click By Value.

The Mapping Command Editor displays.

4. Drag the element you wish to qualify from the source document window and drop it onto the word **path** on the Mapping Command Editor.
5. Highlight the word **value** in the Qualify field in the Mapping Command Editor.
6. Type the value.
7. If you are creating multiple qualifications by value, click Repeat. Then repeat the previous steps.

Clicking Repeat creates the qualification in the Mapping Command window pane and redisplay the Mapping Command Editor.

8. When you have created the desired number of qualifications of this element, click OK.

Clicking OK creates the qualification in the Mapping Command window pane and closes the Mapping Command Editor.

Qualifying an element by expression**◆ To qualify an element by expression:**

1. Right-click the repeating element in the Mapping Command window pane.
2. Right-click to select Qualify.
3. Click By Expression.

The Mapping Command Editor displays.

4. Enter a valid expression into the Mapping Command Editor.
5. If you are creating multiple qualifications by expression, click Repeat. Then repeat the previous steps.

Clicking Repeat creates the qualification in the Mapping Command window pane and redisplay the Mapping Command Editor.

6. When you have mapped the desired number of qualifications of this element, click OK.

Clicking OK creates the qualification in the Mapping Command window pane and closes the Mapping Command Editor.

Editing an element qualification

Edit a qualified element when you want to change the qualification.

◆ To edit a qualified element:

If the element is qualified by occurrence, value, or expression, and you wish to change occurrence, value, or expression qualifications:

1. Double-click the qualify command in the Mapping Command window pane that is to be changed.

The Mapping Command Editor displays.

2. Edit the qualification.
3. Click OK.

◆ If the element is qualified and you wish to change or add a different multi-occurrence qualification:

1. Create a multi-occurrence qualification if one does not already exist.
2. Check that the multi-occurrence qualification command (Default) is selected.
3. Drag repeating element from the source document definition onto a repeating element in the target document definition.

A MapTo command is created in the Mapping Command window pane under the multi-occurrence qualification command (Default).

4. If the new MapTo command is to replace an existing MapTo command, then delete the existing MapTo command by right-clicking on it and selecting Delete.

Advanced mapping techniques

DataInterchange techniques for using literal keywords and other advanced mapping techniques are documented in Appendix B, “Data Transformation mapping commands and functions,” on page 311.

To make mapping easier, DataInterchange Client Help contains syntax for literals, mapping commands, and operators. Search for the following keywords:

- Named Variables
- Literals
- Expressions
- Boolean Operators
- Comparison Operators
- Arithmetic Operators
- Unary Operators
- Special Operators
- Date Conversion Special Operators
- Order of Precedence
- Variables

Translation Tables

DataInterchange Terminology Note

When translation tables uses the terms EDI standard value or trading partner value, it refers to the value in the target document. The term local value refers to the value in the source document.

When DataInterchange translates data from your source document to a target document, it can also substitute one value for another. DataInterchange substitutes values through translation tables. Use translation tables to handle:

- Differences between your data and your trading partners' data. For example, say your trading partner uses its own numbers for parts you sell. You can set up a translation table to convert the part numbers your trading partner uses to those you use. An example of such a translation table is illustrated in Table 65.

Table 65. Translation table, differences in data

Local Value	Trading Partner Value
GLF8088	FR0100
GLF8588	FR0600
GLF8788	FR0800

- Conflicts between data in your source document and data in your target document are different. For example, say the source document uses an abbreviated unit of measurement that is not the same as in the target document. You can create a translation table to substitute a different code in your target document than is in your source document. An example of such a translation table is illustrated in Table 66.

Table 66. Translation table, conflicts with standards

Local Value	Standards and Trading Partner Value
Boxes	BX
Cases	CS
Doz	DZ
Each	EA

You associate translation tables with maps through the use of the Translate function. Refer to Appendix B, “Data Transformation mapping commands and functions,” on page 311 for more information.

Following are detailed procedures for creating new Translation Tables. For information on viewing, copying, editing, renaming, deleting, and printing translation tables, see “Performing common file management tasks” on page 35. For information on exporting translation tables, see “Exporting” on page 52.

Creating the tables

DataInterchange provides forward translation tables in data transformation mapping.

Set up a forward translation table when you want to translate values from the source document into values required by the target document. The translation table contains values with a one-to-one relationship or many local values to one standard or trading partner value. The local value side of the table definition must be unique, as illustrated in Table 67.

Table 67. Sample forward translation table values

Local Value	Standard Value
01	AA
02	BB
03	CC
04	CC

Creating a new translation table

Create a new translation table when you need to substitute values in the target document with values supplied in the source document.

◆ To create a translation table:

1. In the Mapping List Window, click either the Forward Translation tab.
2. Click New on the tool bar.
The General tab displays.
3. Type in a name for the translation table. This is a required field.
You may add a more complete description in the Description field if you wish.
4. Select—from the Data Type drop-down list in the Local Variable group box—the type of data in the source document.
 - Select CH if the data is character data.
 - Select R if the data is numeric data.
5. Select—from the Max Length drop-down list in the Local Variable group box—the maximum length of the data in your source document's fields. The maximum supported length is 35 characters, numbered 001 through 035.

6. Select—from the Data Type drop-down list in the Standard or Trading Partner Variable group box—the type of data in the standard or in your trading partner's format.
 - Select CH if the data is character data.
 - Select R if the data is numeric data.
7. Select—from the Max Length drop-down list in the Standard or Trading Partner Variable group box—the maximum length of the data in the standard or in your trading partner's fields. The maximum supported length is 63 characters, numbered 001 through 063.



NOTE: The combined lengths for the local variable and the standard or trading partner variable cannot exceed 68 characters.

8. Type the translation table in the grid at the bottom of the tab.
 - a. Type the value in your source document in the Local Value column, then press the Tab key.
 - b. Type the value in the target document in the Standards or Trading Partner column, then press the Tab key.
9. When you have finished entering all values required in the translation table, click Save on the tool bar to save the translation table.

Specifying map rules

Once you have completed a map, you must associate it with a trading partner or trading partners. DataInterchange calls those associations rules, map rules, or data transformation map rules.

Applying the minimal trading partners concept

The concept of Minimal Trading Partners attempts to reduce the amount of time spent on administrative functions. The traditional DataInterchange was based on the idea that each Trading Partner would be identified to the product through a Trading Partner Profile and a map rule or usage. Thus, a DataInterchange installation with tens of thousands of trading partners would require an equal number of profiles and map rules or usages, even though the options might be identical. The DataInterchange concept of generic rules and usages reduces the administrative impact of this model, but does not completely meet all its needs. Some installations do not need a setup for a trading partner because they keep that information in their application and pass it to DataInterchange at transformation time. DataInterchange uses a combination of techniques and terminology to accommodate this minimal administrative model.

The following procedures can be carried out from the Trading Partner List windows as well as from the Mapping List window.

Viewing map rules

◆ To view a map rule:

1. In the Mapping List window, select the map for which you want to view map rules, or go to that map's editor window.
2. Click View Usages on the tool bar.

DataInterchange Client runs a query that displays the Map Rules List window, which contains one tab, Data Transformation Map Rule. A list of map rules associated with the map displays.

Creating a map rule

Create a new map rule after you create a new data transformation map and need to associate an existing trading partner with the map. You may also create map rules to associate trading partners with existing maps.

◆ To create a new map rule:

1. In the Mapping List window, select the map for which you want to create map rules, or go to that map's editor window.
2. Click View Usages on the tool bar.

DataInterchange Client runs a query that displays a list of map rules associated with that map. If there are no map rules the list window displays empty.


3. Click New on the tool bar.

The Data Transformation Map Rule Editor displays with the General tab in front.

4. Indicate whether the map rule will be trading partner based or process based.



NOTE: DataInterchange allows you to categorize trading partners by a “process”. You can create a rule between a process (or category) and a map, as well as between a trading partner and a map, to reduce the number of rules needed. For more information, see “Understanding processes and rules” on page 132.

5. Enter the Process ID or the sending and receiving trading partners, as required.
6. Complete any desired optional fields.
Click  for optional field names and descriptions.
7. When you have finished entering all the values you require in the map rule, click Save on the tool bar to save the map rule.

Editing map rules

Edit a map rule when you need to change translation specifications.

◆ To edit a map rule:

1. In the Mapping List window, click the map for which you want to edit a map rule.
2. Click View Usages on the tool bar.

The Data Transformation Map Rule tab displays. If you have created map rules for this map, the existing map rules display in the list window.

3. Double-click the map rule you need to edit.

The general tab displays.

4. Add, change, or delete entries as required.
5. When you have finished entering all values required in the map rule, click Save on the tool bar to save the map rule.

Copying map rules

The copy function allows you to duplicate a map rule within the DataInterchange system in which you are working. If you want to base a new map rule on an existing one, for example, copy the existing map rule under a new name and edit it to the new specifications. You can also copy a rule to a different map or to a new trading partner.

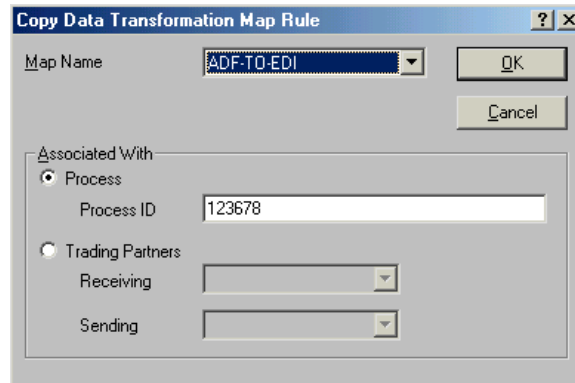
◆ To copy a map rule:

1. In the Mapping Editor, click the map for which you want to create a map rule.
2. Click View Usages on the tool bar.

The Data Transformation Map Rule tab displays. If you have created map rules for this map, the existing map rules display in the list window.

3. Select the map rule you wish to copy.
4. Select Copy from the Actions menu.

The Copy Data Transformation Map Rule dialog box displays.



5. Select a map from the Map Name drop-down list provided.
6. Complete any fields you need in the Associated With section.
7. Click OK.

DataInterchange Client copies the map rule under the new name.

For information on editing and deleting map rules, see “Performing common file management tasks” on page 35.

Compiling control strings

After you complete a map and associate it with trading partners, you must compile a control string before DataInterchange can use the map. DataInterchange Client uses the data you created during the mapping process as input to a program that compiles the map to create a control string. The DataInterchange Host uses the control string in its translation processing.

While compiling, DataInterchange Client checks for errors in the map you created. Error messages are displayed in the Execution Status window. Serious errors are also logged to the Message log.

Compiling a control string is the last thing you do after adding or updating a map. Any time you change a map, you must compile a new control string.

◆ To compile a control string:

1. In the Mapping List window, select the map for which you want to compile a control string.
2. Click Compile Control String.

An Execution Status window displays. Any errors are noted in the window.



NOTE: If you are not using DataInterchange Client in client-server mode, export the control string into the DataInterchange Host. For more information on exporting and importing, see Chapter 4, “Export/Import,” on page 51.

Viewing compiled control strings

Compiled control strings display in the Control Strings List window. Click Maps on the Navigator bar to view the Control Strings List window. Table 68 describes the fields that display in the Control Strings List window.

Table 68. Control String List Window field descriptions

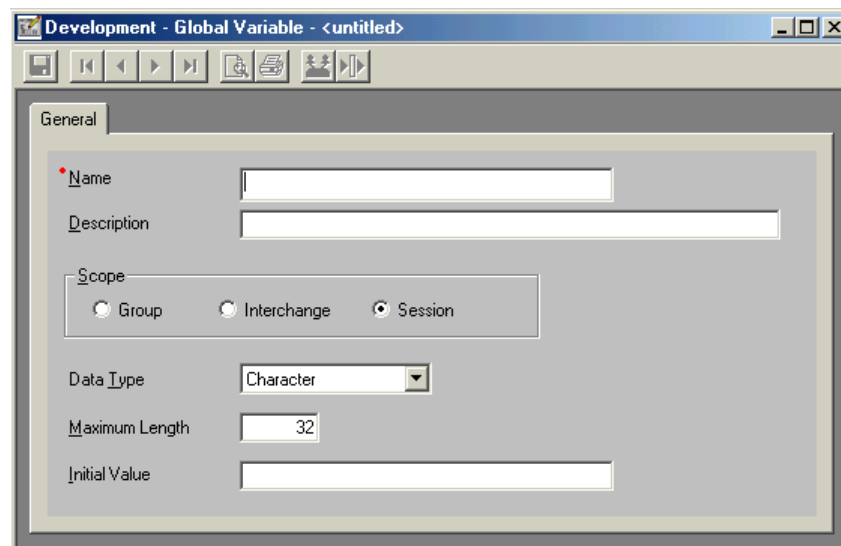
This field. . .	Displays:
Map Name	The name of the map that this control string compiled.
Gen Date	The date the control string was compiled.
Gen Time	The time the control string was compiled.


Defining Global Variables

Global variables are similar to local variables, but they are not unique to any map; they are shared across maps. You can define and maintain a global variable from a map, or from the Mapping List window, using the Global Variables tab. During a translation session, a global variable will retain its value across document translations. It is not reset unless a mapping command specifically changes the value in the global variable.

◆ To define a global variable

1. Click the Global Variables tab on the Mapping List window. The Global Variable editor displays.



2. Complete the required field and any optional fields on the editor. Required fields are preceded by a red dot.
Click  for field descriptions.
3. Select Save from the File menu to save the variable.

Migrating a map to a different source or target document

The source or target document definition in a map can be changed at any time. This is most commonly done to change from one version of a document definition to another. This occurs commonly with EDI standards. To change the source or target document definition:

1. On the General tab of the Mapping Editor, select a different source or target document from the drop-down menus. When changing the version or release of an EDI standard transaction, usually only the dictionary name must be changed. For more information on using the Mapping Editor, see “Utilizing the Map Editor” on page 213.
2. Click Save.

The Map Editor will make an attempt to validate existing mapping commands. If the source document definition was changed, the Map Editor attempts to locate the appropriate place in the Mapping Commands window pane to migrate the existing mapping commands.

Any mapping commands that could not be migrated display with a red X in a circle. You can edit those mappings or drag them to the correct place in the map. Any elements in the source document definition that the map could not identify appear with a red X. Any commands under these can be moved or deleted as needed. Once you are sure you no longer need the unknown elements that have a red X by them, you can delete them by right-clicking and selecting Delete.

Send and Receive mapping

Send maps and receive maps are two of the three supported map types in DataInterchange. Send maps are set of mapping instructions that describe how to translate a proprietary application data document into an EDI standard transaction. Receive maps are a set of mapping instructions that describe how to translate an EDI standard transaction into a proprietary application data document.

Using the Map Editor

DataInterchange Client's Map Editor features a visual means for associating your data fields and records with the EDI standard's elements and segments. The Map Editor uses a split screen that permits you to create map associations.

The data format you created for your application displays on the left side of the screen, and the EDI standard transaction displays on the right. The transaction displays exactly as published, unless you have modified the EDI standard loaded when you set up DataInterchange Client.

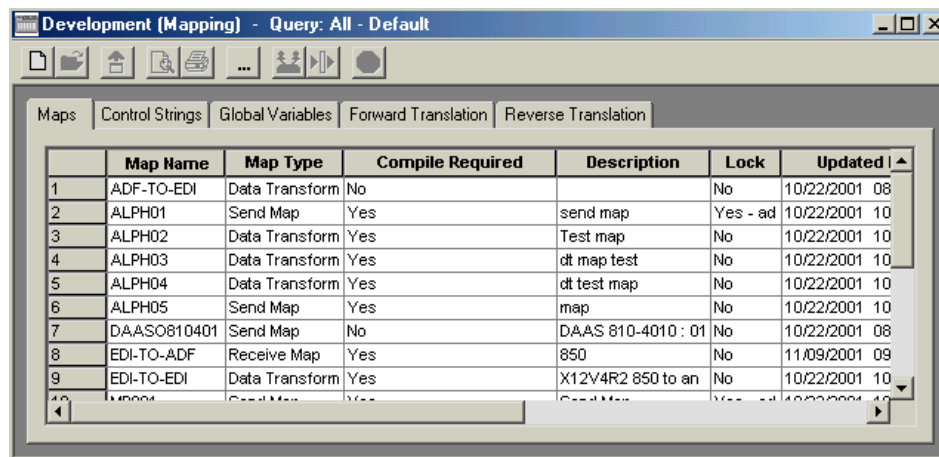
Starting the Map Editor

The Map Editor displays when you select a map from the Map List window, as follows.

◆ To start the Map Editor:

1. Click Mapping on the DataInterchange Client Navigator bar.

The Map List window displays.



	Map Name	Map Type	Compile Required	Description	Lock	Updated ▲
1	ADF-TO-EDI	Data Transform	No		No	10/22/2001 08
2	ALPH01	Send Map	Yes	send map	Yes - ad	10/22/2001 10
3	ALPH02	Data Transform	Yes	Test map	No	10/22/2001 10
4	ALPH03	Data Transform	Yes	dt map test	No	10/22/2001 10
5	ALPH04	Data Transform	Yes	dt test map	No	10/22/2001 10
6	ALPH05	Send Map	Yes	map	No	10/22/2001 10
7	DAASO810401	Send Map	No	DAAS 810-4010 : 01	No	10/22/2001 08
8	EDI-TO-ADF	Receive Map	Yes	850	No	11/09/2001 09
9	EDI-TO-EDI	Data Transform	Yes	X12V4R2 850 to an	No	10/22/2001 10
10	Unknown	Send Map	Yes	Send Map	Yes - ad	10/22/2001 10

This window displays a list of existing maps. All the different types of map are listed. Each row contains information about a component; each column contains data stored in that component. Information in the columns displays in fields in the editor window. The list window, however, also contains the date, time, and user ID of the last update.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to “Creating a query” on page 277.

2. To view a map or to add or change its information, double-click the row of the map you want to work with.

The appropriate Map Editor window displays, with the Details tab in front. You add information or make changes to maps through its tabs and related dialog boxes, as described in the following sections.

Utilizing the Map Editor

The Map Editor allows you to associate your application data with an EDI standard transaction. The editor works by displaying the data format you created on one side of the screen and the EDI standard transaction on the other. The Map Editor allows you to associate components of your data format with the transaction by dragging data format components and dropping them into the correct locations in the EDI standard transaction.

You can map the data format fields to EDI standard data in any of these patterns:

- One field to one data element
- Several fields to one data element
- One field to several data elements

The Map Editor window contains three tabs: General, Details, and Comments. Use the:

- General tab to enter and change global map information.
- Details tab to associate components of a data format with components of the EDI standard transaction.
- Comments tab to type any comments you wish about the selected map.

Following are detailed procedures for creating and editing maps. For information on viewing, copying, editing, renaming, deleting, and printing maps, see “Performing common file management tasks” on page 41. For information on exporting maps, see “Exporting” on page 58.

Creating a send or receive map

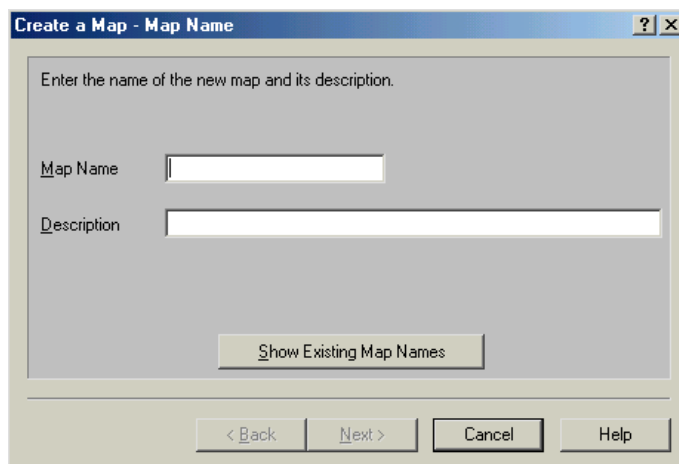
Create a new map after you have created a data format. In some cases, you may need to create a new map to meet the data requirements of a particular trading partner. You can also create a new map from an existing data format to meet new requirements. The EDI standard transaction you are going to work with must also be loaded into DataInterchange Client.

In the most basic sense, creating a new map consists of associating fields in a data format with data elements in a transaction. The following procedure shows how DataInterchange Client makes basic associations between fields and data elements.

◆ To create a new map:

1. In the Map List window, click New Document on the tool bar.

The Create Map wizard displays with the Map Name wizard page.



The dialog box is titled "Create a Map - Map Name". It contains a text area with the instruction "Enter the name of the new map and its description." Below this are two input fields: "Map Name" and "Description". A button labeled "Show Existing Map Names" is positioned below the "Description" field. At the bottom of the dialog are four buttons: "< Back", "Next >", "Cancel", and "Help".

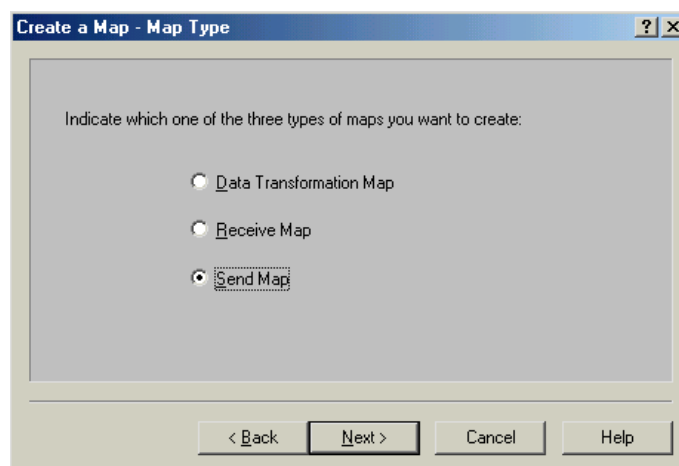
2. Type a map name in the Map Name field.

You may use both letters and numbers to identify your map. Letters display in capitals. You cannot type spaces within the name.

If you wish, you may enter a more complete description of the map in the Description field.

3. Click Next to continue.

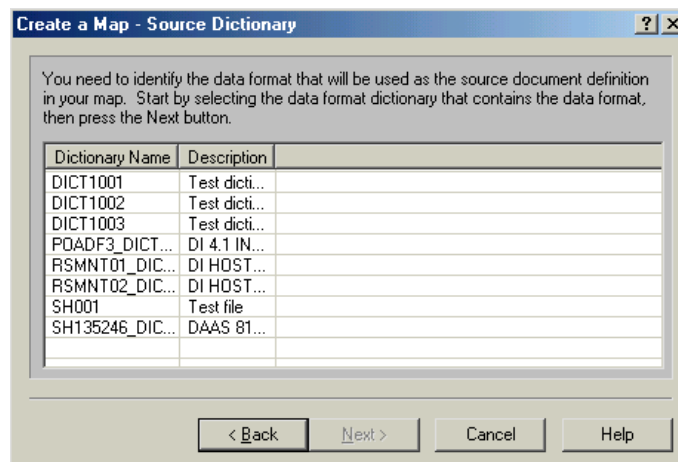
The Map Type wizard page displays.



The dialog box is titled "Create a Map - Map Type". It contains a text area with the instruction "Indicate which one of the three types of maps you want to create:". Below this are three radio button options: "Data Transformation Map", "Receive Map", and "Send Map". The "Send Map" option is selected. At the bottom of the dialog are four buttons: "< Back", "Next >", "Cancel", and "Help".

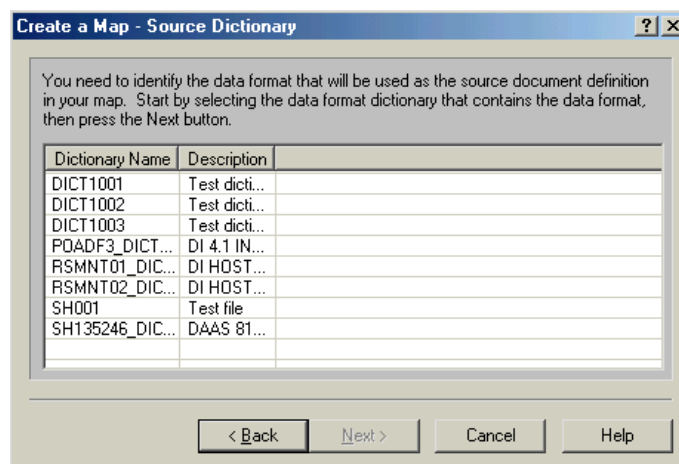
4. Click the appropriate radio button for a send or receive map, and then click Next.

The appropriate Source Dictionary wizard page displays. This page contains a list of data format dictionaries if you are creating a send map. The page contains a list of EDI standard dictionaries if you are creating a receive map.



5. Select the dictionary for the source document, and then click Next.

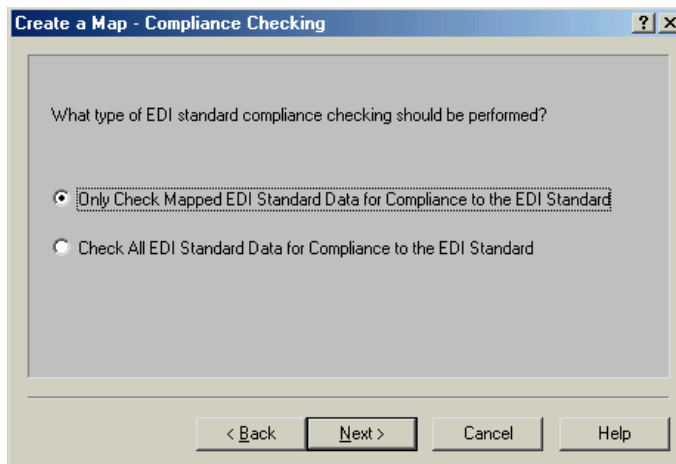
The wizard page listing the available document definitions within the selected dictionary displays. For send maps, this is a list of data formats within the selected data format dictionary. For receive maps, the list contains all transactions within the selected EDI standard dictionary.



6. Select the document definition to be used as the source document definition in your map, and click Next.

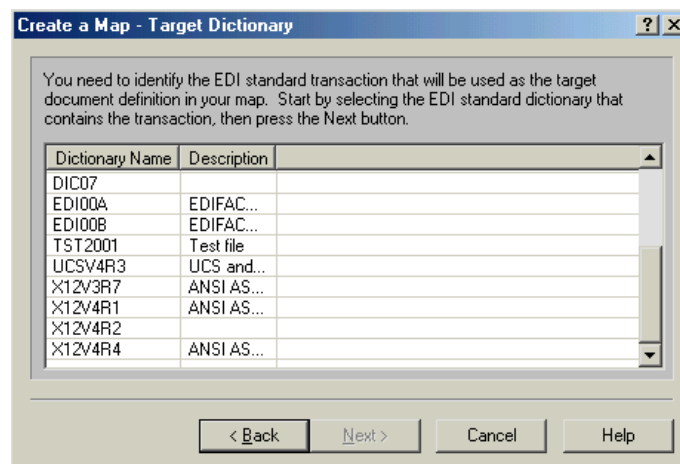
If this is a send map, the Target Dictionary wizard page displays. This page contains a list of EDI standard dictionaries. Proceed to step 8.

If this is a receive map, the Compliance Checking wizard page displays. Proceed to step 7.



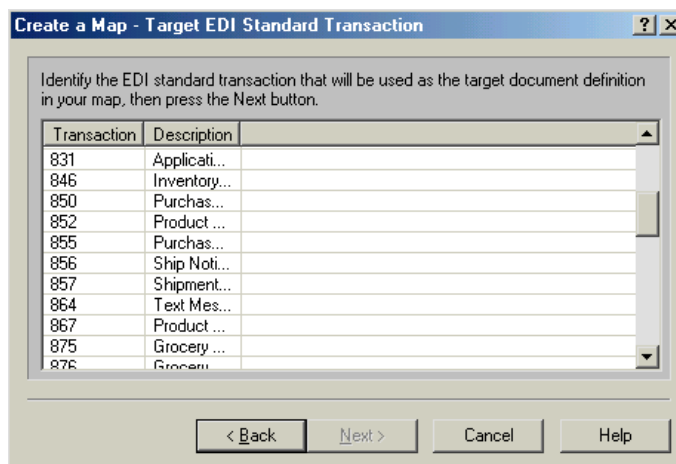
7. Indicate which type of compliance checking you want on this map, then click Next.

The Target Dictionary wizard page displays. This page contains a list of data format dictionaries.



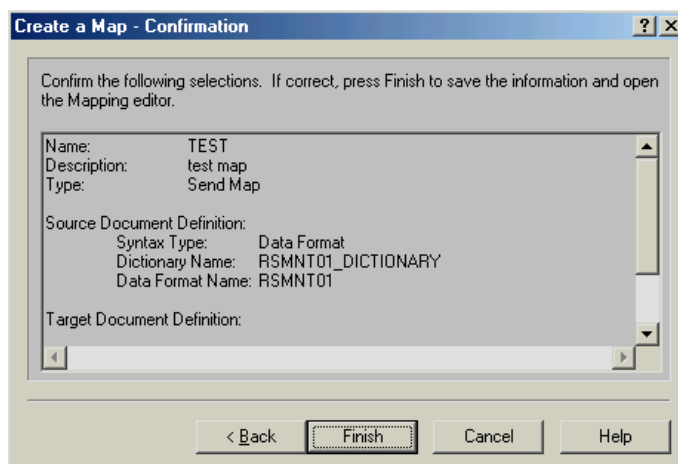
8. Select the dictionary that contains the target document definition you intend to use in your map, and then click Next.

The wizard page listing the available document definitions within the selected dictionary displays. For receive maps, this page contains a list of data formats within the selected data format dictionary. For send maps, the page contains a list of all transactions within the selected EDI standard dictionary.



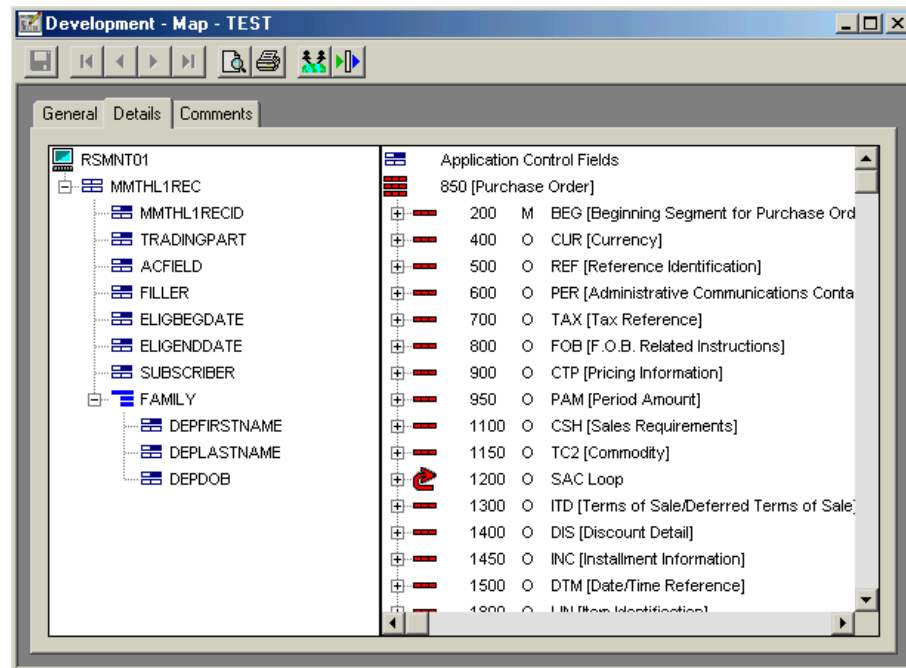
9. Select the document definition to be used as the target document definition in your map, and then click Next.

The Confirmation window displays.



10. Confirm the selections displayed. If correct, click Finish to save the information.

After the information has been saved, the Create Map wizard closes and the Map Editor displays with the Details tab in front. The Details tab allows you to drag components of your data format and drop them on data elements within the transaction.



NOTE: The Direction group box displays grayed. It is for information purposes only. You cannot change the direction of a map.

- a. Click the plus (+) sign next to the component (a loop or segment) in the EDI standard transaction you wish to associate with the data format component. The transaction displays on the right side of your screen.

If you clicked a loop, you see the segments and nested loops that comprise the loop. If you clicked a segment, you see the data elements and composite data elements that comprise the segment, as shown above.

By default, all components of a data format are expanded to show their fields. If you wish to close any portion of the data format in order to see more of it on the screen, click the minus (-) sign.

- b. Click the field you wish to map on the left side of the screen. While holding down the mouse button, drag the field to the corresponding data element of the EDI standard transaction on the right.

The cursor changes to an arrow connected to an outline of the component you are dragging, as illustrated for a field.



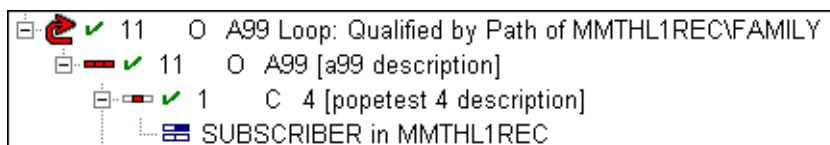
For a list of mapping components and their associated graphics, see Table 69, ‘Mapping components and their graphics,’ on page 249.

When you have dragged the field to the right side of the screen over the data element with which you want to associate it, that component becomes highlighted. Release the mouse button.



NOTE: If the data element does not become highlighted when you move the cursor on top of it, that means it is not a valid place to drop a field. For example, you cannot drop a data format record onto a data element in an EDI transaction.

The Mapping Data Element Editor may display (see Note below). An element mapping displays below the data element, as illustrated.



After you have mapped a data element, a green check mark displays next to the data element to which you have mapped the field. A green check mark also displays next to the segment containing that data element so that you know the segment contains mapped components.



NOTE: This procedure presents a simple mapping situation, in which you associate data format fields with EDI standard data elements. For anything but the most basic mapping associations, you must use DataInterchange’s advanced mapping capabilities. For more information on DataInterchange’s advanced mapping capabilities, see “Utilizing the Mapping Data Element Editor” on page 249.

11. Continue dragging and dropping fields onto data elements until you have mapped all of the information required for this map.
12. Enter any comments on the map in the Comments tab.
13. When you have completed mapping, click Save on the tool bar to save the map.

Editing a map

Edit a map when you have modified its associated data format. You may also need to edit a map to meet specific requirements of a new trading partner.

◆ To edit a map:

1. In the Map List window, double-click the map you wish to edit.

The map displays in the Map Editor window with the Details tab in front.

Any changes you made earlier to the map or its associated items display on the screen. Any changes you made to the data format display on the left side of the screen. If you made any changes to the data format or EDI standard since the last time you opened the map, you see those changes on the Details tab.

2. Create new associations between the data format and the transaction or change existing ones.










- To make an association between a new data format component and a EDI standard component, drag the data format component and drop it on the proper EDI standards component.
- To edit an existing association, double-click the element mapping below the data element. The Mapping Data Element Editor displays.
- To delete an association, click the element mapping you want to delete and press the Delete key.
- You can also change the order of multiple element mappings by clicking on the element mapping you wish to move and dragging it to its new location.

3. Change information as required in the General tab.

You may not change the direction of a map. Create a new map if you need to turn a send map into a receive map or vice versa.

4. Click Save on the tool bar to save the map.

Table 69. Mapping components and their graphics

This icon. . .	Represents a:
	Data Format
	Record
	Field
	Structure
	Transaction
	Loop
	Segment
	Composite Data Element
	Data Element

Utilizing the Mapping Data Element Editor

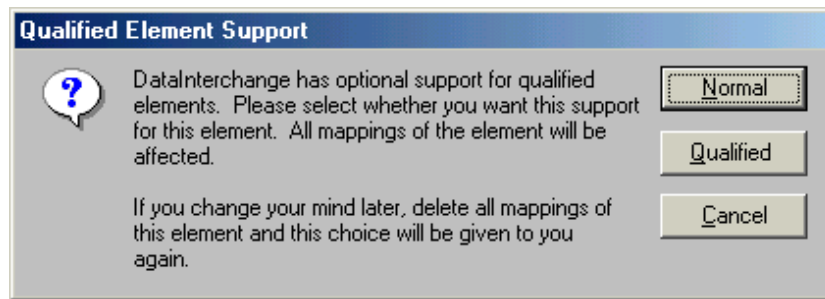
The Mapping Data Element Editor allows you to apply DataInterchange Client's advanced mapping capabilities to data elements and fields. You can:

- Use an accumulator to store, count, and add values to a field or data element. Accumulators allow you to perform such actions as counting segments in a transaction for later placement in a transaction's trailer record. For more information on accumulators, see "Using accumulators" on page 262.
- Associate a DataInterchange literal or other mapping commands with a field or data element. Literals allow you to perform such actions as providing data to trading partners that your application does not contain. For more information on literals, see "Using literals and mapping commands" on page 264.
- Use any of DataInterchange's special handling options on a field or data element. Special handling options allow you to perform such actions as editing dates, verifying data in a field against predefined lists, and converting data from one value to another.

◆ **To apply advanced mapping capabilities to a field or data element:**

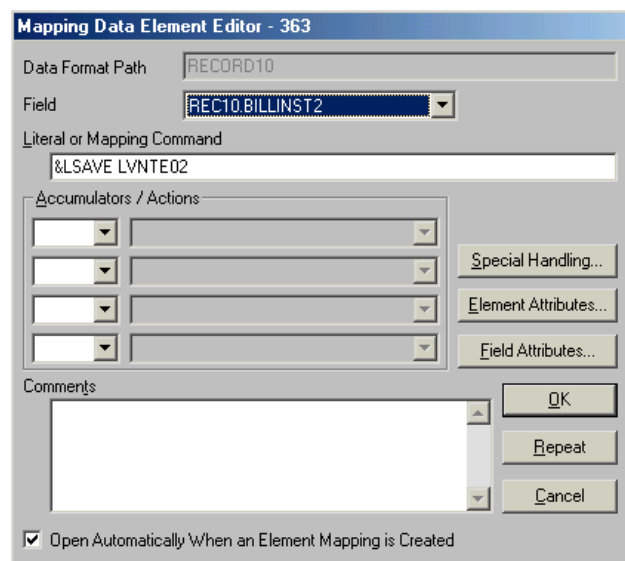
1. Double-click a data element in an EDI transaction that has not been mapped, or if the element has been mapped, then double-click one of the mappings below it.

If this is a receive map, the Qualified Element Support dialog box displays. If this is a send map, skip to step 3.



2. Click Normal.

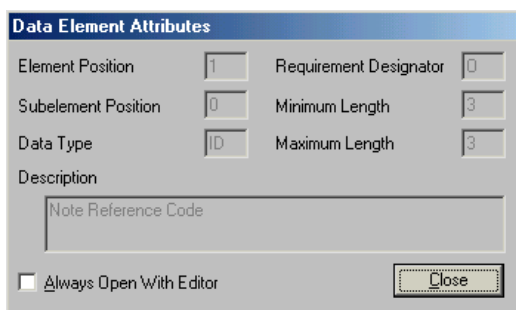
The Mapping Data Element Editor displays.



3. Select the mapping capability you want to apply to this field or data element. You may use more than one.
 - To use a literal, type any literal or mapping commands you desire to use on the field or data element in the Literal or Mapping Commands field. See "Using literals and mapping commands" on page 264.

- To use an accumulator, select an accumulator from the Accumulators drop-down list. For each accumulator, you must select an action from the Actions drop-down list. You may use up to four accumulators on any data element or field. See “Using accumulators” on page 262.
 - To use a special handling option on a field or data element, click Special Handling. See “Using the Special Handling Button” on page 252.
 - To create another element mapping for this data element or field, click Repeat. See “Using the Repeat Button” on page 252.
4. If you wish, you may view attributes of the data element and field you are mapping.
- To view the attributes of the data element on which you are working, click Element Attributes.

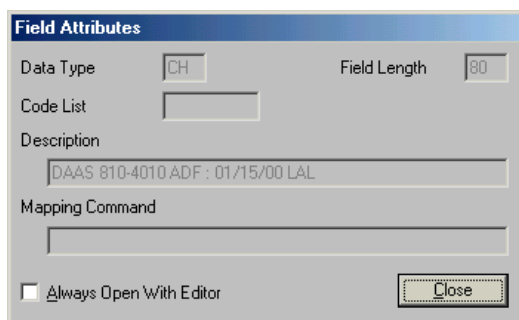
The Data Element Attributes dialog box displays. You see the same information about the data element that displays on the General tab of the Data Elements Editor.



The Data Element Attributes dialog box is shown. It has a title bar 'Data Element Attributes'. Inside, there are several input fields: 'Element Position' with value '1', 'Requirement Designator' with value '0', 'Subelement Position' with value '0', 'Minimum Length' with value '3', 'Data Type' with value 'ID', and 'Maximum Length' with value '3'. Below these is a 'Description' field containing 'Note Reference Code'. At the bottom left is a checkbox 'Always Open With Editor' which is unchecked. At the bottom right is a 'Close' button.

- To view the attributes of the field on which you are working, click Field Attributes.

The Field Attributes dialog box displays. You see the same information about the field that displays on the General tab of the Data Format Field Editor.



The Field Attributes dialog box is shown. It has a title bar 'Field Attributes'. Inside, there are several input fields: 'Data Type' with value 'CH', 'Field Length' with value '80', 'Code List' (empty), and 'Description' containing 'DAAS 810-4010 ADF : 01/15/00 LAL'. Below the description is a 'Mapping Command' field (empty). At the bottom left is a checkbox 'Always Open With Editor' which is unchecked. At the bottom right is a 'Close' button.

5. Type any comments on the mapping in the Comments field.
6. Click OK to save the mapping.

Using the Special Handling Button

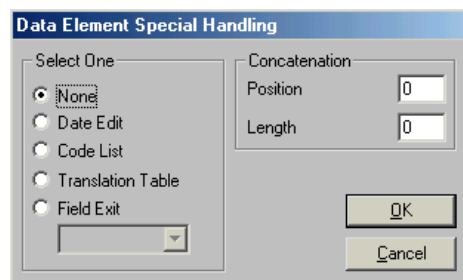
Use the Special Handling button when you want to request special handling on data elements and fields for both send and receive maps. You can:

- Change date formats from one format to another.
- Verify field values against predefined lists.
- Translate field values from one predefined value to another.
- Call an external program to perform custom processing on the value contained in a field or data element.
- Specify the length and position of fields when mapping a concatenation for send maps or specify the length and position of fields that are included in a substring for receive maps.

◆ To request special handling options:

1. While working on a data element or field in the Mapping Data Element Editor, click Special Handling.

The Data Element Special Handling dialog box displays.



2. Click the option corresponding to the Special Handling option you desire.

Click  for more information about the options.

3. Select the item you need from the drop-down list.
4. Click OK to save your options.



NOTE: You may use more than one Special Handling option on any field by using repeat mapping. Click Repeat to create another instance of the element mapping you are working with. Edit that instance with the Mapping Data Element Editor using the Special Handling button again to select another option.

Using the Repeat Button

Use the Repeat button when you want to duplicate the element mapping you completed in the Mapping Data Element Editor. One reason to duplicate an element mapping is to apply more than one special handling option for the data element or field. For example, if you want to use more than one literal command on a field, you need to repeat the element mapping as you can only use one literal command per element mapping.

You can also use repeat mapping to add data to your application when it does not occur in the EDI standard and vice versa. The following example shows how you can add data to your application that does not occur in the EDI standard through a receive map using repeat mapping.

Say that an EDI transaction contains a purchase order total in a particular data element in a particular segment. A purchase order total less than \$100 is an error that you want to report. You can use the Repeat button to repeat the element mapping and enter the mapping commands necessary to accomplish this task, as follows.

◆ **To repeat an element mapping:**

1. Double-click the data element that contains the purchase order total.

The Mapping Data Element Editor displays.

2. Enter in the Literal or Mapping Command field the mapping command &SAVE TOTPO to save the data element value in variable TOTPO.

3. Click Repeat.

The element mapping is saved and another element mapping is created. The Mapping Data Element Editor redisplay with the new element mapping. The saved element mapping appears under the associated data element in the Map Editor.

You may select any options you desire for the new element mapping.

4. Enter the mapping command in the Literal or Mapping Command field to check if the value saved is less than 100 and to issue a user error.

&IF (TOTPO < 100) &ERR(1,1,0,"PO is less than \$100.00").

5. Click OK and finish editing the map.

The new element mapping is designated Literal of (the entered mapping command).

Setting an Application Control Key

To make it easier to find a series of documents in DataInterchange's Transaction Store database, you can set up keys in your maps using application control fields. The application control field contains an ID used to identify the document in the transaction store, such as the purchase order number in a purchase order document. The Application Control Fields dialog box allows you to select up to eight fields, literals and variables to used to construct the application control field.

For example, say your company has a string of characters that occur in front of a purchase order number. Your trading partners likely send only the number, but you can use the Application Control Fields dialog box to have DataInterchange add the characters you desire to their values. You can also use DataInterchange variables to modify data in the application control field.

To make documents received from or sent to a particular trading partner easier to find in the Transaction Store, you could add a string of characters representing that company to the application control field. That character string becomes something you can search on to build a list of documents when viewing the Transaction Store.

For more information on viewing the Transaction Store, see Chapter 26, "Transaction Store," on page 301. For more information on how the Transaction Store works, see the *DataInterchange Administrator's Guide*.



NOTE: Application control fields are optional; the translator does not require them.

◆ **To set up application control fields:**

1. Click the Details tab of the map for which you want to set up application control fields.
2. On the data format (left) side of the screen, click the field you want to set as an application control field and hold down the left mouse button. Drag the field over the Application Control Fields section of the transaction (right) side of the screen, and release the mouse button.

The Application Control Fields dialog box displays.

The Path column contains the name of the data format loop(s), record and (possibly) structures in which the field you dragged occurs. The Field column contains the name of the field. The Length column contains the length of the field.

You may add as many as eight fields to the dialog box.

3. If you want to add fixed data to the application control fields, click Add &LIT.

The Application Control Fields - &LIT Support dialog box displays.

- a. Type in the Enter Literal field the data you wish to add to the application control field.
- b. Type in the Enter Length field the length of the data you wish to add to the application control field.
- c. Click OK.

The data you typed displays in the Field column in a row below the field to which you added it. The data is preceded by &LIT.

4. If you want to include the data from a DataInterchange variable, click Add &VAR.

The Application Control Fields - &VAR Support dialog box displays.

- a. Type in the Enter Variable field the name of the DataInterchange variable you wish to use.
- b. Type in the Enter Length field the length of the data you wish to add to the application control field.
- c. Click OK.

5. When you have completed your setup, click OK.

The Details tab of the Map Editor redisplay.

Specifying qualification

Within the EDI standards, certain segments and groups of segments, called loops, can repeat. Some EDI standards also support data elements or composite data elements that can repeat.

When you map a segment, loop, or data element that repeats within an EDI standard, you must tell DataInterchange which occurrence of the segment, loop, or data element you are using. This is called “qualifying” the segment, loop, or data element.

DataInterchange supports three types of qualification for send and receive map types. You can qualify by:

- Occurrence
- Path
- Value (loops and repeating segments on receive maps only)

Non-repeating data elements can also be qualified by value on receive maps.

When you qualify by occurrence, your mapping is related to the position of data in a repeating sequence. When you qualify by path, your mapping is related to a specific structure or record in the data format and how it handles repeating segments or data elements in the EDI standard. When you qualify by value, your mapping is related to the value that you receive in a data element.



NOTE: You can mix types of qualification for the same loop, segment, or data element. Path qualification works with Value qualification on receive. Occurrence works with Path on both send and receive. You cannot mix Occurrence and Value qualification.

Qualifying loops and segments

DataInterchange allows you to qualify loops and segments by occurrence, path, and value.

Qualify by occurrence when the order in which repeating data occurs in either the data format or EDI standard transaction is important. For more information, see “Qualifying a loop or segment by occurrence” on page 255.

Qualify by path when you want to create multiple segments in an EDI standard transaction to correspond to multiple occurrences of a record or structure in a data format and vice versa. For more information, see “Qualifying a loop or segment by path” on page 257.

Qualify by value when you want to specify data received in a standard transaction to trigger the creation or population of fields in your data format. For more information, see “Qualifying a loop or segment by value” on page 257.

Qualifying a loop or segment by occurrence

Qualify a loop or segment by occurrence when a specific instance of the segment or loop must be mapped to a specific component of the data format. For example, say that you are working on a send map and need to send two addresses, the send-to address and the bill-to address. Map the ship-to address to the first occurrence of the N1 segment and the bill-to address to the second occurrence of N1.



ATTENTION: Qualification by occurrence creates records or segments first and then looks for the data to fill them. If you use occurrence qualification on a receive map, then you need to move data to that record because DataInterchange will create the record, and it may not contain any data.

DataInterchange Client's mapping function automatically qualifies loops and segments that repeat. When you drop a field onto a data element that occurs in a loop or a repeating segment, the loop or segment is automatically qualified by occurrence; the title of the loop or segment changes after the drop to include the words, "Qualified by Occurrence #1". Use the Qualify a Loop or Segment dialog box, as follows.

◆ **To qualify a loop or segment by occurrence:**

1. Double-click a loop or repeating segment (a segment that has a maximum use value greater than one).

For receive maps, either the Qualify a Loop or Qualify a Segment dialog box displays. For send maps, proceed to step 3.



NOTE: You may also drop a field onto a data element in a loop or repeating segment. The loop or segment is automatically qualified by occurrence. See "Editing a loop or segment qualification" on page 258 if you want to change the occurrence number.

2. Click Occurrence.

Either the Qualify a Segment by Occurrence dialog box or the similar dialog box for loops displays. The name of the loop or segment displays in the title bar.

3. Type in a number in the Enter the Occurrence Number field that corresponds to the occurrence number of the component you are mapping.

If you are mapping the second occurrence of a field in your data format to an EDI standard transaction, type 2 in the Enter the Occurrence Number field.

4. If you need to map additional occurrences, click Repeat.

The qualification is added to the map and displays on the transaction side of the Map Editor. The Qualify a Segment by Occurrence Number dialog box or its corresponding loop dialog box redisplay with the next available number in the Enter the Occurrence Number field.

5. When you have specified the desired number of occurrences, click OK.

The qualification is added to the map and displays on the transaction side of the Map Editor.

Qualifying a loop or segment by path

Qualify a loop or segment by path when you need DataInterchange to create multiple instances of either:

- a record or structure for receive maps
- a segment or loop for send maps

For example, say you are working on a receive map and find that the PO1 segment in a purchase order repeats to handle multiple purchase-order line items. You need DataInterchange to create a separate record for each instance of the PO1 segment.

Consequently, you would qualify the PO1 segment by path. That way, DataInterchange will create as many line-item records in your application data as there are occurrences of PO1 in your trading partner's transaction.



NOTE: Qualification by path finds loops or repeating segments and creates records for them on receive and vice versa on send. Use path qualification when the number of records DataInterchange may need to create is unknown.

DataInterchange Client's mapping function automatically qualifies loops and segments that repeat. When you drop a record or structure onto a loop or repeating segment, the loop or segment is automatically qualified by path; the title of the loop or segment changes after the drop to include the words, "Qualified by Path of (Path Name)," as follows.

◆ To qualify a loop or segment by path:

1. Drag a record or structure onto a loop or repeating segment (a segment that has a maximum use value greater than one).

The title of the loop or segment changes to include the words, "Qualified by Path of (Path Name)."

Qualifying a loop or segment by value

Qualify a loop or segment by value when you want the value of data received in a data element to drive DataInterchange's translation of a whole loop or segment. Qualification by value is not supported on send maps.

For example, say you want to qualify the N1 loop with the value of BY in Element 98 (Entity Identifier Code) received in a purchase order to create a buyer record. Further, say that you want the buyer's name to be mapped into the buyer record depending on the value in Element 98 of the N1 loop.

To handle that case, you would map the segment by value. That way, DataInterchange will put specific information into the buyer records it creates from the purchase order depending on the name in each order's Entity Identifier Code.

◆ To qualify a loop or segment by value in receive maps:

1. Double-click a loop or a repeating segment (a segment that has a maximum use value greater than one).

The Qualify a Loop or Qualify a Segment dialog box displays.

2. Click Value.

Either the Qualify a Segment by Element dialog box or a similar dialog box for loops displays. The name of the segment or loop displays in the title bar.

3. Select from the Select an Element drop-down list the name of the data element by which you want to qualify the loop or segment. The list includes all data elements in the segment or in the first segment within a loop.

4. In the Enter a Qualifying Value drop-down list, select a value or type in the value you want to qualify the loop or segment on. The list will contain values when a code list is associated with the selected data element.

5. If you are mapping multiple occurrences, click Repeat.

The qualification is added to the map and displays on the transaction side of the Map Editor. Then either the Qualify a Segment by Element dialog box or the similar dialog box for loops redisplay. Repeat Step 4 and Step 5.

6. When you have mapped the desired number of occurrences of this segment, click OK.

The qualification is added to the map and displays on the transaction side of the Map Editor.

Editing a loop or segment qualification

Edit a qualified loop or segment when you want to change the default qualification that DataInterchange Client places on a segment when you use drag-and-drop mapping of fields, structures, and records.

◆ To edit a qualified loop or segment:

If the Loop or Repeating Segment is Path Qualified and you wish to change to or add occurrence or value qualifications:

1. Double-click the qualified loop or repeating segment.

The Qualify a Segment dialog box displays.

2. Click New or the Replace button, depending on whether you want to add another qualification or replace the existing qualification.

If this is a receive map and you have no other qualifications for this loop or repeating segment, the Qualify a Loop or Qualify a Segment dialog box displays with the choices Value, Occurrence, and Cancel. Continue to step 3.

If other qualifications exist for the loop or segment, then additional qualifications must be of the same type. The Qualify a Loop by an Element, Qualify a Segment by Element, Qualify a Loop by Occurrence Number, or Qualify a Segment by Occurrence Number dialog box displays directly in this case.

For send maps, the Qualify a Loop by Occurrence Number or Qualify a Segment by Occurrence number displays. Skip to step 4.

3. Click Occurrence or Value (if given the choice) depending on how you want to qualify the loop or segment.
4. When you have qualified the desired number of occurrences of the loop or segment, click OK.

◆ If the loop or repeating segment is qualified and you wish to change to or add a different path qualification:

1. Drag a record or structure onto the loop or repeating segment.

The Qualify a Segment or Qualify a Loop dialog box displays.

2. Click New or Replace, depending on whether you want to add another qualification or replace the existing qualification.



NOTE: For receive maps, you may only have one path qualified mapping for a loop or segment. You will be issued an error message if you attempt to create a second path qualified mapping.

Qualifying data elements

Nonrepeating data elements can be used to qualify by value other nonrepeating data elements. Repeating data elements can be qualified by occurrence number or path. A repeating data element can be qualified by both an occurrence number and a path.

Qualify data elements by value when you receive data elements that have qualifiers in a segment. In such cases, you may need to qualify how DataInterchange handles each occurrence of the data element. Data element qualification by value is not supported on send maps.

For example, if data you receive from a trading partner that contains many units of measure (each, case, etc.), you may need to qualify each data element. By doing so, DataInterchange can translate data in the data elements into a single unit of measure for your application.

When you qualify a repeating data element by occurrence, your mapping is related to the position of data in a repeating sequence. When you qualify a repeating data element by path, your mapping is related to a specific structure in the data format and how it handles repeating data elements in the EDI standard.

Qualifying a data element by value in receive maps

Use the Add an Element Qualification dialog box to qualify a data element by the value of another data element, as follows.

◆ To qualify a data element by value:

1. Double-click the data element in the segment that will be used to qualify one or more data elements in the segment. Make sure you qualify data elements before you map them.

The Qualified Element Support dialog box displays.

2. Click Qualified.

The Add an Element Qualification wizard displays.

If you click Normal, the Mapping Data Element Editor displays. For more information, see “Utilizing the Mapping Data Element Editor” on page 249.

3. Select the data element or data elements you want to qualify.

- a. Select the data element or data elements in the Select Elements group box.

- b. Click > to move the selected data element or data elements to the Qualified Elements group box.

Clicking >> moves all data elements in the list.

Clicking < or << removes the selected element or all data element(s) from the Qualified Elements group box.

4. Click Next.

The second page of the Add an Element Qualification wizard displays.

5. Select or enter a qualifying value. Values will appear in the drop-down list if a code list is associated with the qualifying element.

- a. Click the value or values from the Enter a Value drop-down list. You can also type a value in the list.

- b. Click > to move the value to the Qualifying Values group box.

Clicking >> moves all values in the list.

Clicking < or << removes the selected value or all values from the Qualifying Values group box.

6. Click Finish.

An element mapping icon (or icons, if you selected several values) displays below the data element or data elements you qualified and below the qualifying data element. The mapping element is designated Not Mapped - Qualified by Element in Position x with a Value of y where x is the position of the qualifying data element in the segment and y is the value you selected.

Editing a by value data element qualification

◆ To edit a qualified by value data element:

1. Double-click the qualifying data element if you want to add additional values.
The Update an Element Qualification dialog box displays.
2. Select or enter a new qualifying value. Values will appear in the drop-down list if a code list is associated with the qualifying element.
 - a. Click the value or values from the Enter a Value drop-down list. You can also type a value in the list.
 - b. Click > to move the value to the Qualifying Values group box.
Clicking >> moves all values in the list.
Clicking < or << removes the selected value or all values from the Qualifying Values group box.
3. If you want to see all information on the qualified element, click Previous Selected Info.
The Previous Element Qualification Data dialog box displays, displaying all data elements qualified by this element and the values used to qualify those elements.
4. Click OK.
An element mapping icon (or icons, if you specified several values) displays below the data element or data elements you previously qualified. The mapping element is designated Not Mapped - Qualified by element in Position x with a Value of y where x is the position of the qualifying element in the segment and y is the value you specified.

Qualifying a repeating data element or composite data element by occurrence

Qualify a repeating data element or composite data element by occurrence when a specific instance of the data element must be mapped to a specific component area of the data format.

◆ To qualify a repeating data element or composite data element by occurrence:

1. Double-click the repeating data element or composite data element (maximum use value greater than one).
The Qualify an Element by Occurrence dialog box displays.
2. Type in a number in the Enter the Occurrence Number field that corresponds to the occurrence number of the data element you are mapping.
If you are mapping the second occurrence of the data element, type 2 in the Enter the Occurrence Number field.

3. If you need to map additional occurrences, click Repeat.

The qualification is added to the map and displays on the transaction side of the Map Editor. The Qualify an Element by Occurrence Number dialog box dialog box redisplay with the next available number in the Enter the Occurrence Number field increased by one.

4. When you have specified the desired number of occurrences, click OK.

The qualification is added to the map and displays on the transaction side of the Map Editor.

◆ **To qualify a repeating data element or composite data element by path:**

1. Drag a structure onto a repeating data element (a data element that has a maximum use value greater than one).

The title of the data element or composite data element changes to include the words Qualified by Path of (Path Name).

◆ **To add or change to a different path qualification, if the repeating data element is qualified:**

1. Drag a structure onto the loop or repeating segment.

For send maps, the Qualify an Element dialog box displays. Continue to step 2.

For receive maps, you may only have one path qualified mapping for a repeating data element. Dragging a structure onto an existing path qualified repeating data element results in the qualification being replaced by the new path qualification.

2. When the Qualify an Element dialog box displays, click New or Replace, depending on whether you want to add another qualification or replace the existing qualification.

Using accumulators

Accumulators are ad hoc fields that keep running totals or accumulate numeric data. Accordingly, DataInterchange's accumulators can add data to an EDI transaction that does not occur in an application database and vice versa. DataInterchange Client supports global and transaction accumulators. The scope of a global accumulator is an entire translation session. The scope of a transaction accumulator is a single transaction.

You can use accumulators to count the occurrences of a repeated event, and you can use them to total field values for control purposes. For example, an accumulator can count the detail line items in a payment order to provide a hash total required by the EDI transaction. Or you could use an accumulator to total a field named QUANTITY to cross check the amount of invoices paid.

Accumulators can apply to individual transactions or to all transactions in a translation session. To map both an accumulator and a received value for the same element, use the Repeat action to create another occurrence of the element mapping. Then map one occurrence from the data element to a field and the other occurrence from the accumulator to a field. Each element mapping can support up to four accumulators.

Adding an accumulator to a map

You set up an accumulator through the Mapping Data Element Editor, as follows.

◆ To add an accumulator to a map:

1. Double-click an element mapping in an EDI transaction that has been mapped to a field from a data format.
2. Double-click a data element in an EDI transaction that has been mapped to a field from a data format or on a field that has been mapped onto a data element.

The Mapping Data Element Editor displays.

3. Select an accumulator from the Accumulators drop-down list.

Accumulator values are described in Table 70, 'DataInterchange accumulator types,' on page 263.

4. Select an action for the accumulator from the Actions drop-down list.

Only actions that are valid for this data element or field display in the list. Actions are described in Table 71, 'Accumulator actions,' on page 264.

5. Complete any other mapping you need from the Mapping Data Element Editor and click OK.

For outgoing data, accumulator actions are not processed unless:

- Data is generated for the data element or segment, or
- The accumulator is mapped.

For incoming data, the accumulator actions are not processed unless:

- The data element associated with the accumulator is received, or
- The accumulator is mapped and at least the segment containing the data element is received.

Accumulator types

DataInterchange supports transaction accumulators and global accumulators, as follows.

Table 70. *DataInterchange accumulator types*

Name	Type	Description
T0-T9	Transaction Accumulator	Applies only to one transaction and are reset at the beginning of each transaction. You can use a maximum of 10 per transaction. Each accumulator holds a maximum of 31 binary digits.
G0-G9	Global Accumulator	Applies to entire translation session and are reset at the beginning of each translation session. You can use a maximum of 10 per session. Each accumulator holds a maximum of 31 binary digits.

Accumulator actions

DataInterchange supports the following accumulator actions.

Table 71. Accumulator actions

This action. . .	Does this:
Increment the accumulator	Adds 1 to the value stored in the accumulator.
Map the accumulator	Maps the value stored in the accumulator to the data element for send maps and to the application field for receive maps.
Zero the accumulator	Sets the accumulator value to 0.
Map the accumulator and then increment it	Maps the value stored in the accumulator to the data element for send maps and to the application field for receive maps, and then adds 1 to that value. Only one accumulator may be mapped on any given mapping.
Increment the accumulator and then map it	Adds 1 to the value stored in the accumulator, and then maps the value stored in the accumulator to the data element for send maps and to the application field for receive maps.
Add to the accumulator and then map it	Adds the value of an element or field, and then maps the value stored in the accumulator to the data element for send maps and to the application field for receive maps.
Map the accumulator and then add to it	Maps the value stored in the accumulator to the standard data element for send maps and to the application field for receive maps, and then adds the value of a data element or field to it.

Using literals and mapping commands

DataInterchange literals and mapping commands let you add data to a transaction that is not contained in your business application for send maps, or put data into your application data that is not contained in the transaction. When your application does not have specific information required by the transaction, or when you need to pass specific information, you can map a literal to the data element. Conversely, when your application requires data that is not specified in the transaction, you can supply the information by mapping a literal.

A literal is a value that you specify for the field or data element. The value can be a constant or it can be calculated by any of several DataInterchange expressions.

A mapping command is a DataInterchange command that begins with an ampersand (&). For a list of mapping commands, search for the key “mapping commands” in DataInterchange Client Help.

Adding a literal or mapping command to a map

You can add literals or mapping commands to maps through the Mapping Data Element Editor, as follows.

◆ To add a literal or mapping command to a map:

1. Double-click an element mapping that has been mapped to a field from a data format, or double-click on a data element that has not been mapped. If this is a receive mapping, the application data field is required for a literal.

The Mapping Data Element Editor displays.

2. Type the name of the literal and any required expressions.

Note that literals are case-sensitive. Mapping commands and variable names are not.

3. Complete any other mapping you require from the Mapping Data Element Editor, and click OK.

Literals and data types

The following apply to literals used in both send and receive maps:

- For data types BN, Bn, HX, IT, In, Ln, PD, Pn, Zd, and Zn, the translator converts the literal before placing it in the outgoing EDI standard data or the incoming application data. In send maps, it converts the literal to character data. In receive maps, it converts the literal to the application data type. For example, if the data type is BN, the translator converts the literal to binary and then moves it to the application field.
- Do not type a decimal point when one is implied. To use a default value of 9.99 for a field defined as P2 (packed number with two implied decimal positions), enter the literal as 999.
- Literal values for hexadecimal fields are hexadecimal strings. For example, if the application field is defined as a one-byte hexadecimal field and you want to use a default value of X'FF', enter FF as the value of the literal. For incoming data, the translator converts each two bytes of literal value to a single byte of application data.



ATTENTION: Type hexadecimal numbers using their EBCDIC values, not their ASCII values.

- Specify a literal value of zero to move this value into a data element. DataInterchange generally removes leading zeros from application data so that a field containing all blanks or all zeros will result in no value for the data element. A value of zero is treated the same as all other literal values when determining if a segment should be created. The &ZEROSIG special literal can also be used to indicate that zeros within the application field are significant.
- In translation and validation tables, enter numeric values left-justified and formatted according to the data format. For example, if the data format defines a field as R2, enter the value 7 as 7.00 or the value 7.1 as 7.10.



NOTE: Before creating a new map for a specific trading partner, check to see if you can use DataInterchange's advanced mapping functions to meet a trading partner's special processing requirements. You can set up a map so that DataInterchange meets a trading partner's custom data and processing requirements. For more information, see "Utilizing the Mapping Data Element Editor" on page 249.

Advanced mapping techniques

DataInterchange techniques for using literal keywords and other advanced mapping techniques are documented in Appendix C, "Advanced send and receive mapping," on page 355.

To make mapping easier, DataInterchange Client Help contains syntax for literals, mapping commands, and operators.


Click  for more information about mapping.

Table 72. DataInterchange mapping variables for Set Command

In this field. . .	Type:
DIAPPFIL	<p>This variable may be used to change the name of the file to which the application data will be written during a Receive Translate. It will override any value that was used in the receive usage or in the data format definition. It provides the capability for data that is being received to influence the final destination for the data. For example, the statement:</p> <pre>&SET DIAPPFIL 'SPECIAL'</pre> <p>would force the current application data to be written to the application file identified by the DDNAME SPECIAL.</p>
DIAPPTYPE	<p>This variable sets the application file type that corresponds with the file name provided by DIAPPFIL.</p>
DIAUTOCC	<p>This variable may be used to allow automatic century manipulation for both send and receive translation. Century can be automatically added to or removed from the date using the length of the data element or application field.</p> <p>For example, the statement:</p> <pre>&SET DIAUTOCC 1</pre> <p>would allow the translator to add or remove the century to the date depending on the length of the data element or application field.</p>
DICCCTRL	<p>This variable may be used to set the century control year. The default DataInterchange uses is 10. This means, if year is greater than 10, then century is assumed to be 19, otherwise century is assumed to be 20.</p> <p>For example, the statement:</p> <pre>&SET DICCCTRL 95</pre> <p>would force DataInterchange to interpret year as follows, if year is greater than 95, then century is assumed to be 19, otherwise century is assumed to be 20.</p> <p>This variable overrides the Application Defaults profile's Century Control Year field.</p>

Table 72. DataInterchange mapping variables for Set Command

In this field. . .	Type:
DICUSERDATA	This variable is a reserved variable. When this variable occurs in a map, the mapped value will be moved to the associated field in the TRCB. This variable is valid only on receive maps.
DIERRFILTER	This variable can be used to control which errors are actually meaningful to you at a point in time during a translation. A description of the error filter can be found in the DataInterchange Programmers Reference under the section "Error Filtering" and under the description for "ERRFILTER".
DIEXPTRACE	This variable, when given a nonzero value (&SET DIEXPTRACE 1), causes DataInterchange to create a TRACE of the results of all expression evaluations. When tracing is active, DataInterchange will write out message TR0411 to the PRFILE for each expression. The message will show the expression being evaluated and the result of the evaluation. Tracing will remain active until the DIEXPTRACE is given a zero value (&SET DIEXPTRACE 0).
DIMAPCHAIN	<p>This variable can be used when an inbound transaction must be translated into more than one output document. It provides a way for more than one map to be executed for the specified transaction. The last value given to DIMAPCHAIN in a map will establish the application sender ID value that will be used to locate the next map to execute. For example, if MAPABC had this coded:</p> <pre>&SET DIMAPCHAIN APPLICATIONB</pre> <p>the inbound transaction would be translated using map MAPABC, and then it would be translated using the map that is associated with application sender ID APPLICATIONB. The DIMAPCHAIN command will complete translation under the current map before retranslating using the next map indicated by DIMAPCHAIN, whereas the DIMAPSWITCH command will stop translating the map that has the DIMAPSWITCH variable in it, and literally switch to the new map indicated in the command.</p>
DIMAPSWITCH	<p>This variable can be used when data being received needs to be inspected before it can be determined exactly what map should be used against the transaction. It allows you to switch the map that is being executed dynamically based on the data that is being received. A map could be created to initially look at the data being received. Only those data elements necessary to make a map decision would be mapped. DataInterchange would determine the real map to be used by interpreting values resulting from conditional logic expression. For example, a map would contain conditional logic expression:</p> <pre>&IF(X > Y) &SET DIMAPSWITCH APPLICATIONA</pre> <p>Here, if X is greater than Y, the map identified with an application sender ID value of APPLICATIONA would be used to translate the transaction.</p>
DIPROLOG	This variable allows you to override the default XML prolog if you are translating from data format to XML using a send map. Note that data transformation maps are the preferred way to translate to XML. However, send maps are still supported so that XML maps created with DataInterchange 3.1 can continue to be used until they are migrated to data transformation maps.

Table 72. DataInterchange mapping variables for Set Command

In this field. . .	Type:
DISAPSEQ	This variable can be used to allow saving of the SAP IDOC record sequence number on the first error encountered during outbound processing. The sequence number may be provided through the application or using the DataInterchange accumulators. Variable DISAPSEQ is captured in the SAP status record to indicate first record in error. For more information, see the <i>DataInterchange Programmer's Reference</i> .
DIVALLEVEL	This variable can be used to control the level of validation done. It can have the same values as the validation level specified in a usage record, which are: 0 (no validation), 1 (validation tables activated), and 2 (validation tables plus type checking). Any other value other than 0, 1, or 2 will be treated as a 0.
DIVALTYPE	<p>This variable can be used to control the data types for which data type checking is done (validation level of 2). The types that may be specified are DT, TM, N, R, CH, AN, A, and HX. They must be specified in uppercase and separated by a comma. Any value specified that is not in the list above will be ignored. For example, to activate DT, TM and HX validation, the following could be done:</p> <p>&SET DIVALTYPE DT,TM,HX</p>
DIVARTRACE	This variable, when given a nonzero value (&SET DIVARTRACE 1), causes DataInterchange to create a TRACE of all accesses to variables. When tracing is active, DataInterchange will write out message TR0410 to the PRTFILE for each variable access. The message will indicate the variable being accessed and its current value. Tracing will remain active until the DIVARTRACE is given a zero value (&SET DIVARTRACE 0).

Translation Tables

When DataInterchange translates data from your data format to an EDI transaction or from an EDI transaction to your data format, it can also substitute one value for another. DataInterchange substitutes values through translation tables. Use translation tables to handle:

- Differences between your data and your trading partners' data. For example, say your trading partner uses its own numbers for parts you sell. You can set up a translation table to convert the part numbers your trading partner uses to those you use. An example of such a translation table is illustrated in Table 73.

Table 73. Translation table, differences in data

Local Value	Trading Partner Value
GLF8088	FR0100
GLF8588	FR0600
GLF8788	FR0800

- Conflicts between application data and EDI standards. For example, your application uses a code for a unit of measure that does not occur in the EDI standard. You can create a translation table to substitute an EDI standard code for your code on the send side and your code for the EDI standard code on the receive side. An example of such a translation table is illustrated in Table 74.

Table 74. Translation table, conflicts with standards

Local Value	Trading Partner Value
Boxes	BX
Cases	CS
Doz	DZ
Each	EA

You associate translation tables with maps through the Special Handling button on the Mapping Data Element Editor, as described on page 249. This section describes how to set up translation tables.

Following are detailed procedures for creating new Translation Tables. For information on viewing, copying, editing, renaming, deleting, and printing translation tables, see “Performing common file management tasks” on page 35. For information on exporting translation tables, see “Exporting” on page 52.

Creating the tables

DataInterchange provides two types of translation tables:

- Forward translation tables
- Reverse translation tables

Forward translation tables

The most commonly used is the forward translation table. Set up a forward translation table when you want to translate values from your application into values required by your trading partner when sending data, or translate values from your trading partner data into values required by your application when receiving data. This type of translation table contains values with a one-to-one relationship or many application values to one EDI standard value. The local value side of the table definition must be unique, as illustrated in Table 75.

Table 75. Sample forward translation, table values

Application Value	Standard Value
01	AA
02	BB
03	CC
04	CC

This type of translation table may be used on either send or receive maps, when both application and EDI standard values are unique. It can be used in send maps when only the application values are unique.

Reverse translation tables

Set up a reverse translation table when you want to translate one or more values from your trading partner to a single value in your application. This type of translation contains values with a one-to-one relationship or many EDI standard values to one application value. The EDI standard value side of the table definition must be unique, as illustrated in Table 76.

Table 76. Sample reverse translation, table values

Application Value	Standard Value
01	AA
01	BB
01	CC
22	DD
22	EE

The procedures for creating Forward Translation tables and Reverse Translation tables are exactly the same.

Creating a new translation table

Create a new translation table when you need to substitute values supported by your application for values supported by the trading partner's. You may also need to create a translation table to substitute values supported by your application and EDI standard transactions.

◆ To create a translation table:

1. In the Mapping List Window, click either the Forward Translation tab or the Reverse Translation tab.

2. Click New on the tool bar.

The General tab displays.

3. Type in a name for the translation table. This is a required field.

You may add a more complete description in the Description field if you wish.

4. If you are creating a Forward Translation Table, select—from the Data Type drop-down list in the Local Variable group box—the type of data in your data format.

If you are creating a Reverse Translation Table, select—from the Data Type drop-down list in the Standard or Trading Partner Variable group box—the type of data in the EDI standard or in your trading partner's format. This is a required field.

- Select CH if the data is character data.
- Select R if the data is numeric data.

5. If you are creating a Forward Translation Table, select—from the Max Length drop-down list in the Local Variable group box—the maximum length of the data in your data format's fields. The maximum supported length is 35 characters, numbered 001 through 035.

If you are creating a Reverse Translation Table, select—from the Max Length drop-down list in the Standard or Trading Partner Variable group box—the maximum length of the data in the EDI standard's or your trading partner's fields. The maximum supported length is 35 characters, numbered 001 through 035. This is a required field.

6. If you are creating a Forward Translation Table, select—from the Data Type drop-down list in the Standard or Trading Partner Variable group box—the type of data in the EDI standard or in your trading partner's format.

If you are creating a Reverse Translation Table, select—from the Data Type drop-down list in the Local Variable group box—the type of data in your data format. This is a required field.

- Select CH if the data is character data.
- Select R if the data is numeric data.

7. If you are creating a Forward Translation Table, select—from the Max Length drop-down list in the Standard or Trading Partner Variable group box—the maximum length of the data in the EDI standard or in your trading partner's fields. The maximum supported length is 63 characters, numbered 001 through 063.

If you are creating a Reverse Translation Table, select—from the Max Length drop-down list in the Local Variable group box—the maximum length of the data in your data format's fields. The maximum supported length is 35 characters, numbered 001 through 035. This is a required field.



NOTE: The combined lengths for the local variable and the EDI standard or trading partner variable cannot exceed 68 characters.

8. Type the translation table in the grid at the bottom of the tab.
 - a. If you are creating a Forward Translation Table, type the value in your data format in the Local Value column, then press the Tab key.

If you are creating a Reverse Translation Table, type the value in the EDI standard or the value your trading partner wants to receive in the Standards or Trading Partner column, then press the Tab key.
 - b. If you are creating a Forward Translation Table, type the value in the EDI standard or the value your trading partner wants to receive in the Standards or Trading Partner column, then press the Tab key.

If you are creating a Reverse Translation Table, type the value in your data format in the Local Value column, then press the Tab key.

DataInterchange Client inserts another row and the display shifts back to the Local Value column in Forward Translation Tables and the Standards or Trading Partner Value column in Reverse Translation Tables. For more information on how the grid editor works, see "Using editor window grids" on page 38.
9. When you have finished entering all values required in the translation table, click Save on the tool bar to save the translation table.

Specifying send and receive usages

Once you have completed a map, you must associate it with a trading partner or trading partners. DataInterchange calls those associations send usages or receive usages, or jointly usages. Send usages are also referred to as send map usages. Receive usages are also referred to as receive map usages.

Applying the minimal trading partners concept

The concept of Minimal Trading Partners attempts to reduce the amount of time spent on administrative functions of EDI. The traditional DataInterchange was based on the idea that each Trading Partner would be identified to the product through a Trading Partner Profile and a send usage or receive usage. Thus, a DataInterchange installation with tens of thousands of trading partners would require an equal number of profiles and usages, even though the options were

identical. The DataInterchange concept of generic usages reduces the administrative impact of this model, but does not completely meet all its needs. Some installations do not need a setup for a Trading Partner, relying on post-translator processes to validate EDI transactions. DataInterchange uses a combination of techniques and terminology to accommodate this minimal administrative model.

Applying the Minimal Trading Partners concept, usages allow you to specify the same transaction mapping for several trading partners and provide specific overrides for each trading partner. This gives you the capability to use a single map for several different trading partners. Each send or receive usage instructs DataInterchange Client which components of the map should be used and which should be overridden for that particular trading partner. Refer to “Specifying usages and rules” on page 136 for more information.

The following procedures can be carried out from the Trading Partner windows as well as the Mapping windows.

Viewing usages

◆ To view a usage:

1. In the Mapping List window, click the map for which you want to view usages, or go to that map's editor window.
2. Click View Usages on the tool bar.

DataInterchange Client runs a query that displays the Usages List window, which contains one tab. The Send Map Usages tab displays if the map is a send map. It displays a list of Send Usages associated with the map. The Receive Map Usages tab displays if the map is a receive map. It displays a list of Receive Usages associated with the map.

Creating a send or receive usage

Create a new send or receive usage after you create a new send or receive map and need to associate an existing trading partner with the map. You may also create usages to associate trading partners with existing maps.

◆ To create a new send or receive usage:

1. In the Mapping List window, click the map for which you want to create usages, or go to that map's editor window.
2. Click View Usages on the tool bar.

DataInterchange Client runs a query that displays a list of send or receive usages associated with that map. Either the Send Map Usages tab or the Receive Map Usages tab displays listing the results of the query. If you created usages for this map previously, the existing usages display in the list window.

3. Click New on the tool bar.

The Send Map Usage Editor or the Receive Map Usage Editor displays with the General tab in front.

4. For both send and receive usages, you must identify the sending and receiving trading partners.

For send maps, the sending trading partner identifies the internal trading partner who is the sender of the document. The receiving trading partner identifies the external trading partner who is the receiver of the document. Only trading partner profiles defined with a trading partner type of application trading partner or both will be included in the sending trading partner TP nickname drop-down list. Trading partner profiles are defined with a trading partner type of EDI Trading partner or both.

For receive maps, the sending trading partner identifies the external trading partner who is the sender of the document. The receiving trading partner identifies the internal trading partner who is the receiver of the document. Only trading partner profiles defined with a trading partner type of EDI Trading partner or both will be included in the sending trading partner TP nickname drop-down list. Trading partner profiles are defined with a trading partner type of application trading partner or both.

For information on defining generic send and receive usages, see “Defining generic send usages” on page 277 and “Defining generic receive usages” on page 278.



NOTE: In send usages, the primary key is Map ID + Sending Trading Partner + Receiving Trading Partner + Internal Trading Partner ID. It is not possible to have two usages on the same map where all the primary key fields are the same. For example, it is not possible to have two usages on the same map where the only difference is that one usage is a test usage while the other one is production. If you need to send both test and production data to the same trading partner using the same map, then you can create a production usage, and in the Application Defaults profile set the check box Production Usage - Test Message. If that will not work in your environment, and if both the test and production usage must be active, then you must use a different internal trading partner ID for the test and production usages. If it is not necessary that they both be active at the same time, then you could copy the map and associate the test usage with one map, and the production usage with the other.



NOTE: In receive usages, the primary key is Map ID + Sending Trading Partner + Receiving Trading Partner + Application Sender ID + Application Receiver ID. It is not possible to have two usages on the same map where all the primary key fields are the same. For example, it is not possible to have two usages on the same map where the only difference is that one usage is a test usage while the other one is production. If you need to send both test and production data to the same trading partner using the same map, then you can create a production usage, and in the Application Defaults profile set the check box Production Usage - Test Message. If that will not work in your environment, and it is not necessary that they both be active at the same time, then you could copy the map and associate the test usage with one map, and the production usage with the other.

You must create a Trading Partner profile before a trading partner's nickname displays in the list. For information on creating Trading Partner profiles, see Chapter 17, "Trading Partners," on page 127.

5. Type values for any desired optional fields in the proper fields.

Click  for optional field names and descriptions.

6. When you have finished entering all the values you require in the send or receive usage, click Save on the tool bar to save the send or receive usage.

Editing send and receive usages

Edit a send or receive usage when you need to change translation specifications.

◆ To edit a send or receive usage:

1. In the Mapping List window, click the map for which you want to edit a send or receive usage.
2. Click View Usages on the tool bar.

Either the Send Map Usages tab or the Receive Map Usages tab displays. If you have created usages for this map, the existing usages display in the list window.
3. Double-click the send or receive usage you need to edit.

The general tab displays.
4. Add, change, or delete entries as required.
5. When you have finished entering all values required in the send or receive usage, click Save on the tool bar to save the send or receive usage.

Copying send or receive usages

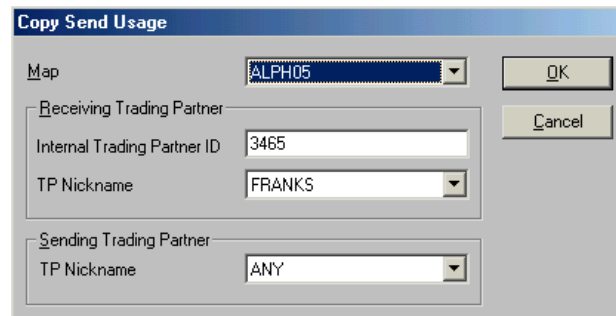
The copy function allows you to duplicate a send or receive usage within the DataInterchange system in which you are working. If you want to base a new send or receive usage on an existing one, for example, copy the existing trading send or receive under a new name and edit it to the new specifications. You can also copy a usage to a different map or to a new trading partner.

◆ To copy a send or receive usage:

1. In the Mapping List window, select the map that is associated with the send or receive usage you wish to copy.
2. Click View Usages on the tool bar.

Either the Send Map Usages tab or the Receive Map Usages tab displays. If you have created usages for this map, the existing usages display in the list window.
3. Select the send or receive usage you wish to copy.
4. Select Copy from the Actions menu.

The Copy Send Usage or Copy Receive Usage dialog box displays.



The dialog box is titled "Copy Send Usage". It contains the following fields and controls:

- Map:** A dropdown menu showing "ALPH05".
- Receiving Trading Partner:** A group box containing:
 - Internal Trading Partner ID:** A text field with "3465".
 - TP Nickname:** A dropdown menu showing "FRANKS".
- Sending Trading Partner:** A group box containing:
 - TP Nickname:** A dropdown menu showing "ANY".
- Buttons:** "OK" and "Cancel" buttons are located on the right side of the dialog.

5. Select or enter new values into the field provided on the dialog box.
6. Click OK.

DataInterchange Client copies the send or receive usage using the information specified on the copy dialog box.

DataInterchange Client copies the send or receive usage using the information specified on the copy dialog box. For information on deleting send or receive usages, see “Performing common file management tasks” on page 35.

Defining generic send usages

Where multiple trading partners can use the same send usage definition and map, a generic send usage can be defined to DataInterchange. When combined with a generic routing code supplied by the application, it provides the capability to define one or more generic usages, each of which can handle multiple trading partners.

The generic routing code is an optional, three-character code provided by the application to select the correct generic usage when no specific usage has been defined for the trading partner. The Generic Routing Code can be provided in one of three ways:

1. The Translator Control Block (TRCB) includes a three-character field for the generic routing code. The API calls can provide the routing code in the TRCB.
2. For C and D processing, the C record includes a field for the three-character generic routing code.
3. For raw data processing, an application field that contains the generic routing code can be specified when defining the data format. The application field should be from one to three characters. If less than three characters, the value is filled with blanks. If greater than three characters, it is truncated to three.

Defining send usages

When defining send usages, an Internal Trading Partner ID that begins with an ampersand (&) indicates it is a generic usage. For a specific usage by routing code, specify an ampersand (&) followed by the three-character generic routing code that the application provides to select this usage.

If the application does not want to provide a routing code or does provide a routing code but wants to select a default generic usage, the Internal Trading Partner ID can be defined as a single (&) followed by blanks. This type of generic definition is the default definition and is selected when no routing code is provided, or when the routing code is provided but no specific usage is found, or when the routing code contains blanks.

The purpose of the generic routing code is to allow a file containing one data format to generate more than one type of transaction (such as purchase orders and purchase-order changes). The generic routing code is required when the user wants to process transactions using different maps that reference the same data format.

For example, if the user wants to process purchase orders and purchase-order change documents that are defined using the same data format, the application could provide the following: a routing code of POR for a purchase order and DataInterchange would select the usage with the Internal Trading Partner ID of &POR, which would reference a purchase order map; and a routing code of POC for a purchase-order change and DataInterchange would select usage &POC, which would reference a purchase-order change map. An additional benefit when using this type of definition is that multiple documents can be included in the same raw data file if they all use the same data format.

Defining generic receive usages

When transactions received from multiple trading partners can use the same mapping, a generic receive usage can be defined to DataInterchange to handle multiple trading partners with a single usage and map.

Defining Receive Usages

When defining the generic receive usages, a trading partner nickname with only an ampersand (&) is a special form indicating that it is a generic usage. The generic usage is selected when the normal selection process using the trading partner nickname does not find a receive usage.

Many generic receive usages can be defined as long as one of the listed match criteria is different (such as Application Sender, Application Receiver, Agency, Version, or Release, production/test).

Compiling control strings

After you complete a map, you must compile a control string from the map before DataInterchange can use the map. DataInterchange Client uses the data you created during the mapping process as input to a program that compiles the map to create a control string. The DataInterchange translator uses the control string in its translation processing.

While compiling, DataInterchange Client checks for errors in the map you created. Error messages are displayed in the Execution Status window. Serious errors are also logged to the Message log.

Compiling a control string is the last thing you do after adding or updating a map. Any time you change a map, you must compile a new control string.

◆ To compile a control string:

1. In the Mapping List window, click the map for which you want to compile a control string.
2. Click Compile Control String.

An Execution Status window displays. Any errors are noted in the window.



NOTE: If you are using DataInterchange Client in client-server mode, skip Step three and four, as the client communicates directly with the DataInterchange Host database. For more information on exporting and importing, see Chapter 4, “Export/Import,” on page 51.

3. Export the control string from DataInterchange Client.
4. Import the control string into DataInterchange Host.

Viewing compiled control strings

Compiled control strings display in the Control Strings List window. Click Mapping on the Navigator bar to view the Control Strings List window. Table 77 describes the fields that display in the Control Strings List window.

Table 77. *Control String List window field descriptions*

This field . . .	Displays:
Map Name	The name of the map that this control string compiled.
Gen Date	The date the control string was compiled.
Gen Time	The time the control string was compiled.

Mapping hierarchical loops

A hierarchical loop is similar to an organization chart. Just as an organization chart shows you the various groups of people and their relationships to the whole, a hierarchical loop shows you each group of data and its relationship to the whole.

Hierarchical loops define different levels of data, which can be used in any sequence and skipped when appropriate. This allows you to place the loop anywhere in your data.





NOTE: DataInterchange includes support for Hierarchical Loops. For detailed explanations of HL loops and how DataInterchange handles them, refer to “Hierarchical loops” on page 398.

Mapping the HL Segment

DataInterchange provides special handling for Hierarchical Loops. This section shows how to map the HL segment using the DataInterchange Client interface.

◆ To map an HL segment:

1. Double-click an HL Loop.
The Hierarchical Loop Support dialog box displays.
2. Click Special HL Support.
The Qualify a Hierarchical Loop dialog box displays.
3. Type the ID of the node you are mapping in the Node Number field.
Click  for field descriptions.
4. Select a hierarchical level code from the Hierarchical Level Code (HL03) drop-down list.
Click  for field descriptions.
5. If you need to repeat the segment mapping, click Repeat.
The Hierarchical Loop Support dialog box redisplay.
6. When you have completed all repeat mappings, click OK.
The words Qualified by HL Logic. . . display next to the segment name in the Mapping Editor.

Creating fixed-to-fixed maps

Fixed-to-fixed translation is a method of translating application data from one format to application data of another format. This requires a send map to direct the movement of data between data formats.



NOTE: It is recommended that a data transformation map be used to create a map from a data format to a data format instead of using a send map. Using a data transformation map allows you to specify the original data format definitions in the source and target documents, avoiding the need to convert a data format to an EDI standard.

◆ To create a fixed-to-fixed map

1. The target data format must be converted to an EDI standard. Begin by selecting the target data format from the Data Format List window.
2. Select Create Standard from Data Format from the Action menu to convert the target data format into an EDI standard.



NOTE: The name of the generated EDI standard is taken from the Application File/Queue name field on the General tab of the Data Format Editor.

This conversion only needs to be done a single time once the target data format is defined. If the target data format changes, the Create Standard from data format can be repeated to delete the old EDI standard and create a new one.

An EDI standard is created. You can view this using the EDI standards related list windows and editors.

3. Create a send map using the source data format and the target EDI standard just created. Refer to “Creating a send or receive map” on page 241.
4. Create send usages as needed. Refer to “Specifying usages and rules” on page 136.
5. Compile the control string for the map. Refer to “Compiling control strings” on page 278.

Migrating a map to a new standard

Although migration is usually from one version of an EDI standard transaction to a later version of the same EDI standard transaction, sometimes it is necessary to migrate a map from one EDI standard to another.

Client migration option

1. Ensure that the new version of the EDI standard is loaded.
2. Migrate the map to the new EDI standard. On the General Tab of the Map Editor, select the new EDI standard and transaction using either the Source Document Definition or Target Document Definition fields, depending on whether this is a send or receive map. For more information on using the Map Editor, see “Utilizing the Map Editor” on page 241.

PART 5. Administration

Queries

DataInterchange Client's Query function gives you complete control over the information that displays in any DataInterchange Client list window. Queries control both the fields that display in the list window and which items display in the list window.

Queries are particularly useful after you have been using DataInterchange Client for a while and the number of items in your list windows is large. You can write queries that filter items to display only the set you wish to view. This ability will be particularly useful in working with the Transaction Store in order to monitor the status of your activity.

DataInterchange Client comes with preset queries for each of the list windows you can access in the application. You can use these queries as a template to create and modify your own.

For information on printing reports of query results, see Chapter 25, "Reports," on page 295.

About queries

A query is, in simplest terms, a request for specific information from the database. A query allows you to determine what information you want to see and the order in which you want to see it.

Purpose

Queries have three main uses. First, they provide you with general information on what is currently stored in the database. Second, they allow you to monitor various aspects of your transaction. Third, they increase system performance by limiting the items that display in DataInterchange Client windows.

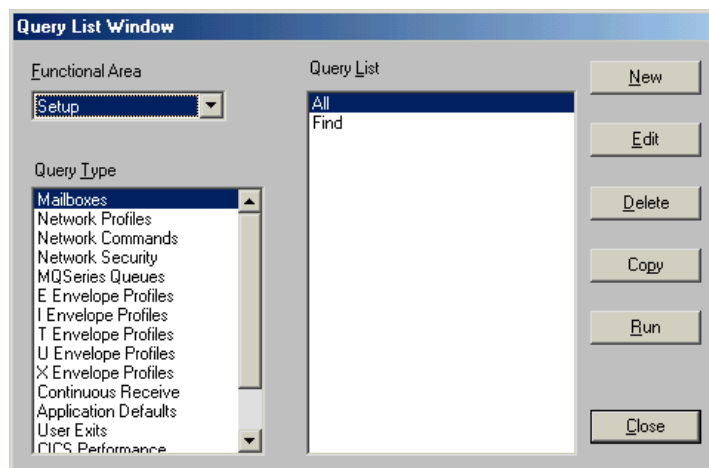
You run a query each time you open any of the list windows in DataInterchange Client. You see the results of that query displayed in the list portion of the window. For example, when you open the Trading Partners List window, you generally run the Trading Partners "All" query, which finds all Trading Partner profiles and displays the resulting list with many trading partner fields shown in the window. This list contains all Trading Partner profiles and sorts them by their Trading Partner Nickname.

When you open a list window, the last query that was run is executed. To modify the query or to select an alternate query, use the Properties button, as described in “Modifying list window information” on page 24.

The second purpose of queries is to allow you to monitor the status of you translation activity. This will likely be most useful in the Transaction Store. For example, you can track translation activity with a specific trading partner, or any other type of information you would like to track.

Setup overview

You work with queries starting in the Query List window, which you access by selecting the Open Query List command from the File menu.



The Functional Area drop-down list at the top left of the window allows you to select a functional area within DataInterchange Client. The Query Type list box allows you to select the type of item you wish to work with within the functional area.

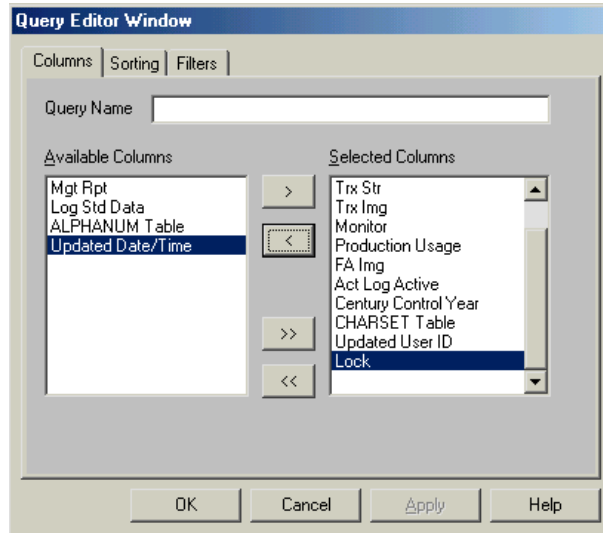
For example, selecting the Trading Partner functional area displays queries available for the Trading Partner List window. You can then choose to work with queries for either Trading Partner profiles or Contact profiles, which are the two profiles that are available in the Trading Partner functional area.

The Query List box in the middle of the window displays all queries that have been defined for a specific item within a functional area. The buttons on the right side of the window allow you to create a new query; edit, delete, copy or run an existing query; and, finally, close the window when you are finished with it.



NOTE: Queries provided with DataInterchange Client are read-only. Therefore, you cannot edit or delete them. You can, however, copy them and then make changes to or subsequently delete the copy. Understand that in copying a preset query, you are actually creating an identical query with a new name.

To create an identical query, edit an existing query, or copy an existing query, you work in the Query Editor window. By clicking the appropriate tab, you can select the columns, or fields, that will display in your query, decide how you want DataInterchange Client to sort information in the columns, and apply criteria that filter, or limit, the information that displays in query results.



Working with queries

Once you have decided what information you want to receive from a query, you can create a new query, or copy and/or edit an existing query. DataInterchange Client also allows you to run queries at any time so you can view specific information without having to go to a particular functional area. When you are finished with a query, you can also delete it, except for the preset queries, as previously noted.

Creating a query

To assemble and view a combination of information, you may either create a new query or copy an existing query, edit it, and save the changes under a new name.

It is best to create entirely new queries when the query you wish to create differs from previously created queries. It is convenient to use the Copy and Edit features to create new queries when the query you wish to create is similar to an existing query.

◆ To create a query:

1. Choose the Open Query List command from the File menu.
The Query List window displays.
2. Use the Functional Area drop-down list to select a specific functional area within DataInterchange Client.
3. In the Query Type list box, click the item you want to work with.
4. If you want to create a new query by modifying an existing query, make a selection in the Query list box and then click Copy. Otherwise, click New. The Query Editor window displays.

5. In the Columns tab, use the Query Name field to type in the query name. Then use the Available Columns and Selected Columns list boxes, and the arrow selection buttons to determine which columns will be included in the query results.

The > button moves the highlighted item from the Available Columns to the Selected Columns list box.

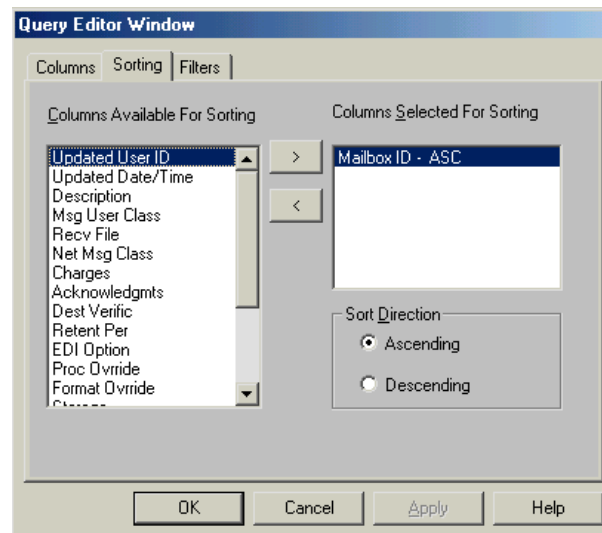
The < button moves the highlighted item from the Selected Columns to the Available Columns list box.

The >> button moves all items from the Available Columns to the Selected Columns list box.

The << button moves all items from the Selected Columns to the Available Columns list box.

6. If you want to sort information that displays in the list, click the Sorting tab.

The Query Editor window displays changes. Sorting lets you specify the way in which that data will be organized in the results of your query.



- a. Use the Columns Available For Sorting list box to choose which columns the query results will be sorted on. The < and > buttons allow you to move column selections back and forth as desired.
 - b. For each column, choose whether you want to sort Ascending (0 to 9, A to Z) or Descending (Z to A, 9 to 0).
7. If you want to narrow the list based on the value of a particular field, set up a filter, as described below.

8. When you are finished working with the various tabs in the Query Editor window, click OK.
DataInterchange Client returns to the Query List window.
9. At this point, you can begin again in another Functional Area or with a different query type.
If you are finished, click Close to close the Query List window.

Using filters in queries

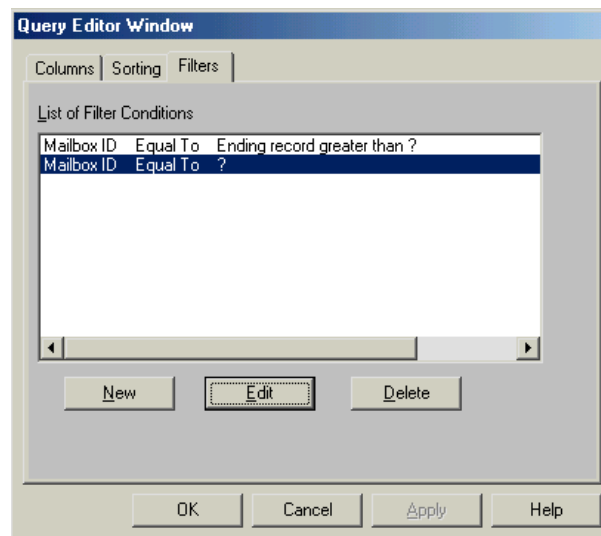
Filters allow you to narrow the list of items that display in a list window based on values in specific fields or columns. Only information that meets the filter specifications is displayed in the query results. If no filters are defined, then all information available for each selected column displays.

You can set up two types of filter queries: static and dynamic. Static filters allow you to set up the query to filter for the same value in a column every time you run the query. Dynamic filters prompt you to type in a value each time you run the query so that you can specify the particular value that meets your needs at that time.

◆ To set a filter in a query:

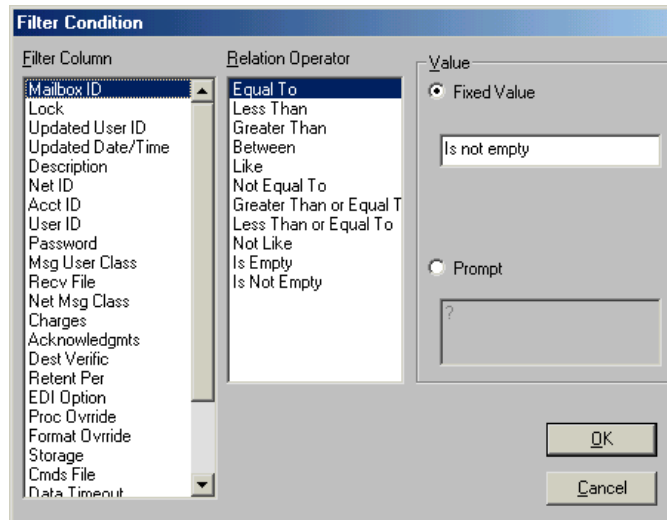
1. Create a new query following the procedure beginning on page 287 or edit an existing query following the procedure beginning on page 291.
2. Click the Filters tab.

The Query Editor window displays changes.



3. If you are creating a new filter condition, click New to set up filters for this query. If you are copying and modifying existing filter conditions, you can highlight an existing filter and then click Edit.

The Filter Condition dialog box displays.



4. In the Filter Column list box, click the column you want to filter.
5. In the Relation Operator list box, click the desired operator for the selected column. The function of most of the operators are self-evident. Explanations of “Like,” “Between,” “Is Empty,” and “Is Not Empty,” however, display in Table 78 on page 291.
6. The Value area allows you to specify whether you want this specific filter to always be the same, and what the value of the filter is, or whether you want to be prompted each time you run the query so that you can specify the particular value that meets your needs at that time. Note that if you click the Prompt option, the prompt field below is automatically completed with the selected column and relation operator. You can change the prompt text as desired. Changing the prompt text will not change the filter.



NOTE: Many database management systems are case sensitive. Use the correct case when entering a value.

For example, Mailbox ID and Equal To are selected in the illustration above. If you always want the query to return the same Mailbox ID, click Fixed Value option and then type in the specific Mailbox ID you want the query to return.

If, however, you may want to see different Mailbox IDs at different times, click the Prompt option. When you run the query, you will be prompted for a specific Mailbox ID.



NOTE: Not all operators work in all columns. A Date column, for example, will not accept a “Like” operator. You will receive database errors if you run such a query.

7. When you are finished setting filter conditions, click OK.
You return to the Filters tab of the Query Editor window.
8. When you have finished working with the query, click OK.
DataInterchange Client returns to the Query List window.
9. Click Close to close the Query List window.

Table 78. Selected Relation Operators

This operator. . .	Displays:
Like	Items similar to the string of characters typed in the field. You can also use the % wildcard with the Like operator. The % stands for any string of characters. If you typed in RPT%, for example, you would receive a list of all items that begin with RPT. NOTE: Results of queries using Like may vary depending on the client-server middleware your company uses. It is recommended that you always use the % operator at the end of strings when using Like.
Between	Items between the two values you enter in the Value list box. Whether the endpoints are included when you use the Between operator depends on the middleware you use.
Is Empty	All fields that contain Null as a value. Use this operator when you want to find an error, as null fields only exist when there are errors.
Is Not Empty	All fields besides those containing Null as a value. Use this operator when you want to show all fields, including those that include blanks.

Editing a query

DataInterchange Client allows you to make changes to existing queries when the information you wish to view changes.

◆ To edit a query:

1. Choose the Open Query List command from the File menu.
The Query List window displays.
2. Use the Function Type drop-down list to select a specific functional area within DataInterchange Client.
3. In the Query Type list box, click the type of query you want to work with.
4. Make a selection in the Query List box, and then click Edit.

The Query Editor window displays.

At this point, you can follow Step 5 through 10 of the procedure for creating a new query. See “Creating a Query” on page 287.

5. When you are finished working with the various tabs in the Query Editor window, click OK.

DataInterchange Client returns to the Query List window.

At this point, you can choose another query to edit. If you are finished, click Close to close the Query Editor window.

Copying a query

You may want to create a new query that is similar to an existing one. For example, one of your queries displays all purchase orders received on a given day and you need one to display all purchase order acknowledgments received. Rather than creating an entirely new query, copy the first query and make the appropriate changes.

◆ To copy a query:

1. Highlight the query you want to copy.
2. Click Copy on the Query List window.

The Query Editor window displays.

3. Type a new name for the query in the Query Name field.
4. Modify the query in any way you want, then click OK.

The Query List window redisplay, and the query displays in the Query List.

Deleting a query

If you have not used a particular query in some time and do not anticipate using it again, you may want to delete that query.

◆ To delete a query:

1. Highlight the query you want to delete.
2. Click Delete on the Query List window.
3. If you are sure you want to delete the query, click Yes.

DataInterchange Client deletes the query.

Running a query

After you have created a query, DataInterchange Client allows you to run it so you can view the information you have selected in a tabular format. You can run any existing query at any time.

◆ **To run a query:**

1. Choose the Open Query List command from the File menu.

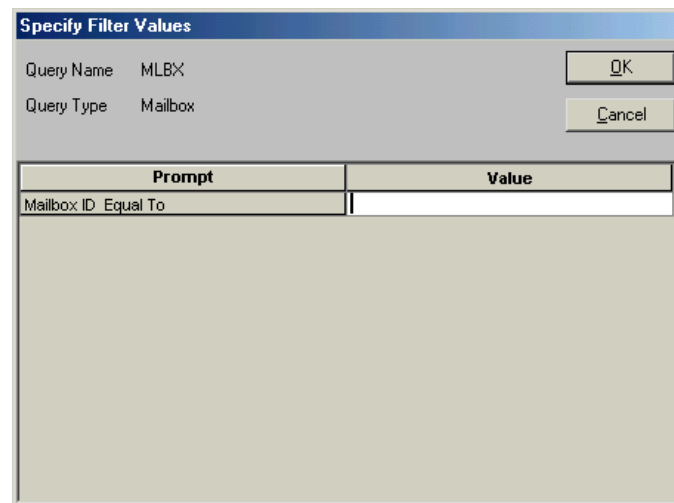
The Query List window displays.

2. Use the Functional Area drop-down list to select a functional area within DataInterchange Client.
3. In the Query Type list box, click the name of the item you want to work with.
4. Click the specific query you wish to execute and then click Run.

DataInterchange Client immediately executes the query and displays the results.

If the query filter calls for a prompt, the Specify Filter Values dialog box displays on top of what will be the results display.

Steps 5 and 6 apply only if a query filter has been specified in the query.



The dialog box titled "Specify Filter Values" contains the following elements:

- Query Name: MLBX
- Query Type: Mailbox
- Buttons: OK and Cancel
- A table with two columns: Prompt and Value.

Prompt	Value
Mailbox ID Equal To	

5. Press Tab to move to the Value field and enter a value appropriate to the Parameter Prompt.



NOTE: Many database management systems are case sensitive. Use the correct case when entering your values.

6. Click OK to display the results of the query.

The results of the query display in the list window.

Reports

The DataInterchange Client Report function lets you preview and print the results of a query. In essence, a report displays information for printing on paper or previewing on screen. Printing a report allows you to create a permanent copy of any information you choose.



NOTE: DataInterchange Client includes a run-time version of Crystal Reports** Version 5 for using the layouts that come with DataInterchange Client. To create your own report layouts, you must purchase the full Crystal Reports product.

For information on creating queries, see Chapter 24, “Queries,” on page 285.

About reports

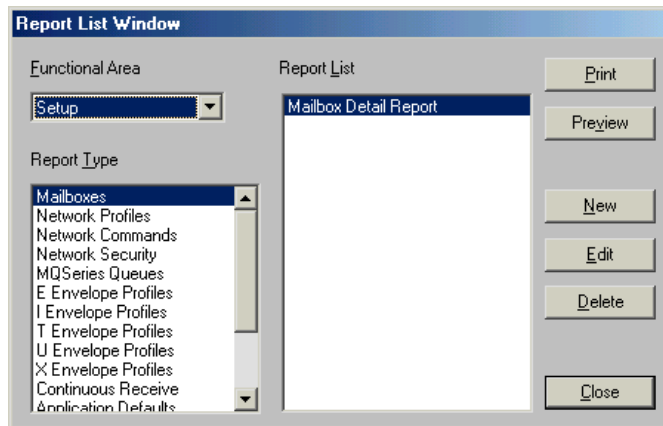
A report is a combination of a query and a report layout that is printed on paper rather than displayed in a DataInterchange Client window.

The main purpose of a report is to create a paper document that can make it easier for you to share and/or retain certain key information stored in the database. You can create management reports for circulation, or permanent records for filing, as needed.

Setup overview

When you create a report, you associate a query you have already created with a report layout that has also already been created, and name it.

You begin all report procedures in the Report List window, which you access by selecting the Open Report List command from the File menu.



The Functional Area drop-down list at the top left of the window allows you to select a functional area within the DataInterchange Client. The Report Type list box allows you to select the type of item within the functional area.

The Report List list box in the middle of the window displays all reports that have been defined for the selected list window. The buttons on the right side of the window allow you to print or preview an existing report, create a new report, edit or delete an existing report, and, finally, close the window when you finish working with it.

To create a new report or edit an existing report, you work in the Report Editor window. This window allows you to name reports, select the query to associate with a report as well as select a specific layout to use for the report.

Working with reports

Once you have decided what information you want to include in your report (and verified that there is a query to provide that information), you can create a report. You can also print or preview any report, edit it, and, when you are finished with a report, you can delete it.

Creating a report

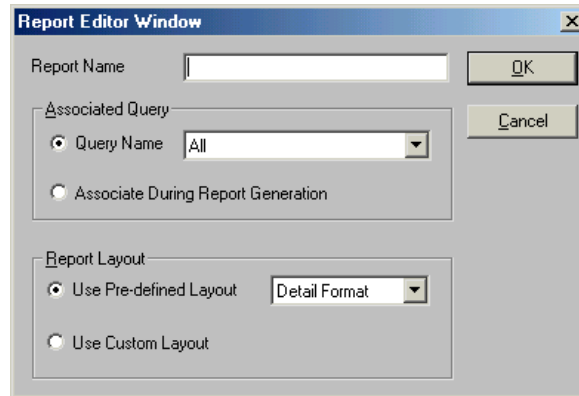
Creating a report is a fairly straightforward process in which you name the report, pick a query to associate with the report and specify a layout for the report.

◆ To create a report:

1. Choose the Open Report List command from the File menu.
The Report List window displays.
2. Use the Functional Area drop-down list to select a functional area within DataInterchange Client.
3. In the Report Type list box, click the specific list window you want to work with.

4. Click New.

The Report Editor window displays.



5. Type in a name for the report.
6. In the Associated Query section, you have two choices: You can select the name of an existing query to permanently associate with this report, or you can choose to be prompted for the name of an existing query at the time you go to print the report.
7. In the Report Layout section, you also have two choices: You can choose one of the predefined report layouts that come with DataInterchange Client, or you can choose a custom report layout you have created using Crystal Reports.



NOTE: DataInterchange Client includes a runtime version of Crystal Reports for using the layouts that come with DataInterchange Client. To create your own report layouts, you must purchase the full Crystal Reports product. After you create a report format, copy the resulting *.RPT file into the /CRW sub-directory located with the DataInterchange Client install directory.

8. Once you have completed the specifications for the new report, click OK. You are returned to the Report List window.

At this point, you can create additional reports, print or preview the report you've just created, or click Close to close the Report List window.

Editing a report

You can change the information displayed in a report by changing the query on which the report is based. If you want to change the list of items generated by the report, you must edit the query on which the report is based or associate a new query with the report. To change the appearance or layout of the report, you must use the Crystal Reports product.

◆ To edit a report:

1. Choose the Open Report List command from the File menu.

The Report List window displays.

2. Use the Functional Area drop-down list to select a functional area within DataInterchange Client.
3. In the Report Type list box, click the type of item you want to work with.
 - a. If you have chosen a report created by DataInterchange Client, you can click Edit to view information pertaining to that report.
 - b. If you have chosen a report that you have created, you can click Edit to edit information in that report.

The Report Editor window displays.

4. At this point, you can follow steps 6 through 8 of the procedure for creating a new report. See “Creating a report” on page 296.
5. When you are finished making the desired changes to the report, click OK.

DataInterchange returns to the Report List window.

At this point, you can choose another report to edit. If you are finished, click Close to close the Report Editor window.

Deleting a report

◆ To delete a report:

1. Click the report in the Report List window to highlight it.
2. Click Delete.

DataInterchange Client asks you to confirm the deletion.

3. If you are sure you want to delete the report, click Yes.

DataInterchange Client deletes the report.



NOTE: This does not delete the report layout associated with the report.

Printing or previewing a report

The main reason you create a report is to obtain a printed copy of the results of a query, either for circulation to others, or for creating a permanent printed record if necessary. Previewing the report allows you to view the report pages before they are printed. You can also download the report into various types of files through the Print Preview screen.

◆ To print a report:

1. Choose the Open Report List command from the File menu.
The Report List window displays.
2. Use the Functional Area drop-down list to select a functional area within DataInterchange Client.
3. In the Report Type list box, click the type of item you want to work with.
4. Make a selection in the Report List list box and then click Print. The report is automatically sent to the currently selected printer.



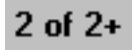






◆ To preview a report:

1. Choose the Open Report List command from the File menu.
The Report List window displays.
2. Use the Functional Area drop-down list to select a functional area within DataInterchange Client.
3. In the Report Type list box, click the type of item you want to work with.
4. Make a selection in the Report List list box and then click Preview.
The report is displayed on screen, showing you how it would display on paper.
5. To print the report, click Print.
6. To close the screen, click Close.



NOTE: If you use the Print Preview function, you see a screen from Crystal Reports. Table 79 provides brief descriptions of the Print Preview controls.

Table 79. Print Preview Screen

This button. . .	Does this:
	Displays the first page of multi-page reports.
	Displays the previous page of multi-page reports.
	Tells you the number of the page currently on display, out of the total number of pages which have been viewed. If all of the pages have not been viewed, a plus sign displays after the second number. (For example, if you are viewing the third page and you have viewed up through page five and there are more than five pages, this display will read 3 of 5+.)
	Stops building long reports.
	Displays the next page of multi-page reports.
	Displays the last page of multi-page reports.
	Prints the selected item.
	Allows you to download the report to a variety of files and formats, including e-mail, HTML, Microsoft Word**, and Lotus Notes**. See the <i>Crystal Reports User's Guide</i> for details.
	Changes the size of the page you are viewing.

Transaction Store

DataInterchange Client provides the ability to view the Transaction Store, which is created and maintained by DataInterchange Host. You can only view the contents of the Transaction Store using DataInterchange Client if you are in client-server mode.

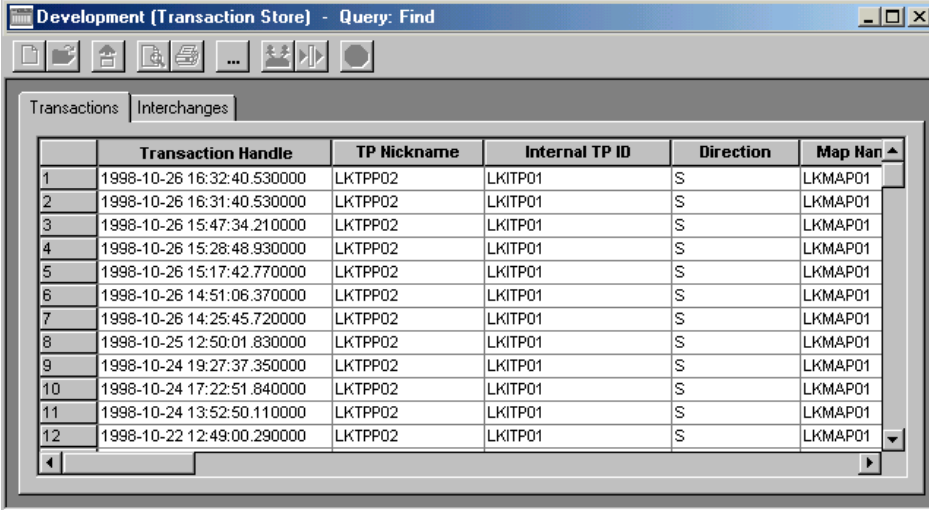
For a complete description of the Transaction Store, see the *DataInterchange Administrator's Guide*.

About Transaction Store

The Transaction Store is created by and resides on DataInterchange Host so that you can maintain a history of all of your translation activities and track those activities. DataInterchange Client allows you to view information in the Transaction Store if you are using client-server mode. The main tools for doing this are DataInterchange Client's default Transaction Store reports and its query functions.

Client overview

You view the Transaction Store through the Transaction Store List window, which you access by clicking the Transaction Store button on the DataInterchange Client Navigator bar.



	Transaction Handle	TP Nickname	Internal TP ID	Direction	Map Name
1	1998-10-26 16:32:40.530000	LKTPP02	LKITP01	S	LKMAP01
2	1998-10-26 16:31:40.530000	LKTPP02	LKITP01	S	LKMAP01
3	1998-10-26 15:47:34.210000	LKTPP02	LKITP01	S	LKMAP01
4	1998-10-26 15:28:48.930000	LKTPP02	LKITP01	S	LKMAP01
5	1998-10-26 15:17:42.770000	LKTPP02	LKITP01	S	LKMAP01
6	1998-10-26 14:51:06.370000	LKTPP02	LKITP01	S	LKMAP01
7	1998-10-26 14:25:45.720000	LKTPP02	LKITP01	S	LKMAP01
8	1998-10-25 12:50:01.830000	LKTPP02	LKITP01	S	LKMAP01
9	1998-10-24 19:27:37.350000	LKTPP02	LKITP01	S	LKMAP01
10	1998-10-24 17:22:51.840000	LKTPP02	LKITP01	S	LKMAP01
11	1998-10-24 13:52:50.110000	LKTPP02	LKITP01	S	LKMAP01
12	1998-10-22 12:49:00.290000	LKTPP02	LKITP01	S	LKMAP01

The Transaction Store List window contains two tabs, Transactions and Interchanges. To view information on Transactions or Interchanges, click the appropriate tab to display a list of Transaction Store items.

The Transactions List window displays a list of all the transactions in the database regardless of whether they have been enveloped. This list window, however, does not contain envelope information, nor does it display all of the information you can view using DataInterchange Host.

The Interchanges List window displays only enveloped transactions and includes envelope information. It also includes the interchange control numbers so that you can view details related to the network and functional acknowledgment of each transaction.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to “Creating a query” on page 287.

Host Setup

DataInterchange maintains the Transaction Store automatically during translation, unless you have specified otherwise through an Application Defaults profile. For any given application, you can choose to store:

- All transactions
- Only those transactions that are successful
- Only those transactions that fail
- No transaction information at all

For more information see Chapter 6, “Application Defaults profiles,” on page 75. For general information on the Transaction Store, see the *DataInterchange Administrator's Guide*.



NOTE: Translations done with the PERFORM TRANSFORM command do not use Transaction Store.

Using Transaction Store queries

The Transaction Store contains a vast amount of data. DataInterchange Client provides you with the tools you need to limit the number of transactions that display as a result of running a query.

The Transaction Store List window runs a Find query as the default query (unlike other DataInterchange Client list windows, which run an All query). The Find query allows you to limit the number of transactions that display as a result of the query. When a Find query runs, it displays the Specify Filters Value dialog box, which is described in “Using filters in queries” on page 289.

Table 80, “Transaction Store Transaction Find Query Field Descriptions,” on page 303, describes the fields for the Transactions query.

Table 81, “Transaction Store Interchange Find Query Field Descriptions,” on page 304, describes the fields for the Interchanges query.

The last word of each field name specifies what type of filter is being run on this field.

- The “Like” operator displays all values that resemble the value typed in the field.
- The “Equal to” operator displays all values that are equal to the value typed in the field.
- The “Between” operator displays all values between the value typed in the From field and the value typed in the To field.



ATTENTION: You must enter dates in the same format that you have defined for Windows. If you enter a date in a different format than the Windows default, you will receive errors when running queries.

Table 80. Transaction Store Transaction Find Query Field Descriptions

In this field. . .	Type:
Transaction Handle Between [from value]	<p>The first in a range of IDs the Transaction Store assigns to a transaction. The transaction handle is a 26-byte character field that uniquely identifies each transaction. It is the key around which DataInterchange database tables are built. This ID is a concatenation of a date, time, and a sequence number, as follows:</p> <p><i>yyyy-mm-dd-hh.mm.ss.nnnnnnn</i></p> <p>For send transactions, the date and time indicate when the transactions were translated to the standard. For receive transactions, the date and time indicate when the transactions were de-enveloped.</p> <p>Because it is difficult to know the exact transaction handle, you can type just the date, or date and time.</p>
Transaction Handle Between [to value]	<p>The last in a range of IDs the Transactions Store assigns to a transaction. See the description above for contents and syntax.</p>
TP Nickname Like	<p>The nickname of a trading partner identified in the trading partner profile. The Like operator allows you to use the % wildcard, as described in Table 78, “Selected Relation Operators,” on page 291.</p>

Table 80. Transaction Store Transaction Find Query Field Descriptions

In this field. . .	Type:
Direction (S/R)	S for a send transaction; R for a receive transaction.
Standard Transaction ID Equal To	The name of the standard transaction you want to view.
Network ID Equal To	The Network ID of the transactions you want to view.
Internal TP Nickname Like	The Internal Trading Partner ID you want to view.
Application Control Number Like	The Application Control Number you want to include in your query. The value must match exactly the application control value in the data, including upper and lower casing of characters.
Map Name Like	The name of the map upon which the transactions you want to view are based.
Test Transaction (Y/N)	Y indicates that you want to view test transactions; N indicates that you do not.
Date Created Between [from value]	The beginning date from which you want to see transactions.
Date Created Between [to value]	The ending date to which you want to see transactions.

Table 81. Transaction Store Interchange Find Query Field Descriptions

In this field. . .	Type:
Transaction Handle Between [from value]	<p>The first in a range of IDs the Transactions Store assigns to a transaction. The transaction handle is a 26-byte character field that uniquely identifies each transaction. It is the key around which DataInterchange database tables are built. This ID is a concatenation of a date, time and a sequence number, as follows:</p> <p><i>yyyy-mm-dd-hh.mm.ss.nnnnnnn</i></p> <p>For send transactions, the date and time indicate when the transactions were translated to the standard. For receive transactions, the date and time indicate when the transactions were de-enveloped.</p> <p>Because it is difficult to know the exact transaction handle, you can type just the date, or date and time.</p>
Transaction Handle Between [to value]	The last in a range of IDs the Transactions Store assigns to a transaction. See the description above for contents and syntax.
TP Nickname Like	The nickname of a trading partner identified in the trading partner profile. The Like operator allows you to use the % wildcard, as described in Table 78, "Selected Relation Operators," on page 291.
Direction (S/R)	S for a send transaction; R for a receive transaction.
Standard Transaction ID Equal To	The name of the standard transaction you want to view.

Table 81. Transaction Store Interchange Find Query Field Descriptions

In this field. . .	Type:														
Network ID Equal To	The Network ID of the transactions you want to view.														
Interchange Control Number Like	The interchange control numbers of the transaction you want to select. If you entered 0 in the Envelope type field, the interchange control number is the same as the group control number.														
Group Control Number Like	The group control numbers of the transactions you want to select. This field applies only to transactions enveloped as part of a functional group.														
Transaction Control Number Like	The transaction set control numbers of the transactions you want to select.														
Application Control Number Like	The application control numbers of the transactions you want to select.														
Functional Ack Expected (Y)	Enter Y in this field and leave the Functional Ack NOT Expected field blank to display all the transactions for which you expect to receive functional acknowledgments.														
Functional Ack NOT Expected (Y)	Enter Y in this field and leave the Functional Ack Expected field blank to display all the transactions for which you do not expect to receive functional acknowledgments.														
Functional Ack Received (Y)	Enter Y in this field and leave the Functional Ack NOT Received field blank to display all the transactions for which you have received functional acknowledgments.														
Functional Ack NOT Received (Y)	Enter Y in this field and in the Functional Ack Received field blank to display all the transactions for which you have not received functional acknowledgments.														
Functional Ack Status Code (A/E/M/P/R/X)	<p>This field allows you to display transactions by Functional Acknowledgment Status Code. Valid values are:</p> <table> <tr> <th>Code</th><th>Description</th></tr> <tr> <td>A</td><td>Transactions for which functional acknowledgments have been accepted</td></tr> <tr> <td>E</td><td>Transactions for which functional acknowledgments are accepted, but errors are present</td></tr> <tr> <td>M</td><td>Rejected—message authentication code failed</td></tr> <tr> <td>P</td><td>Partially accepted</td></tr> <tr> <td>R</td><td>Transactions for which functional acknowledgments have been rejected</td></tr> <tr> <td>X</td><td>Rejected contents after description could not be analyzed</td></tr> </table>	Code	Description	A	Transactions for which functional acknowledgments have been accepted	E	Transactions for which functional acknowledgments are accepted, but errors are present	M	Rejected—message authentication code failed	P	Partially accepted	R	Transactions for which functional acknowledgments have been rejected	X	Rejected contents after description could not be analyzed
Code	Description														
A	Transactions for which functional acknowledgments have been accepted														
E	Transactions for which functional acknowledgments are accepted, but errors are present														
M	Rejected—message authentication code failed														
P	Partially accepted														
R	Transactions for which functional acknowledgments have been rejected														
X	Rejected contents after description could not be analyzed														
Map Name Like	The name of the map upon which the transactions you want to view are based.														

Using Transaction Store reports

DataInterchange Client is shipped with default Transaction Store reports that allow you to display the most common types of business information. You access Transaction Store reports through the Report List window, as described in “About reports” on page 295. Select Transaction Store in the Functional Area drop-down list.

Transaction Store Reports run a query, either the default Find query or one you create, to display data.

Table 82 describes default Transactions Reports.

Table 83 describes default Interchanges Reports.

Table 82. Default Transaction Report

This report. . .	Displays:
List of Transactions	A list of transactions as defined in the query you ran to create the report.
Pending Functional Acknowledgments	A list of transactions that have been sent but for which functional acknowledgment transactions have not been received from trading partners.
Transactions Detail	Details of transactions selected in the query you ran to create the report.
Transactions Status Summary	A summary of transactions selected in the query you ran to create the report.

Table 83. Default Interchanges Reports

This report. . .	Displays:
Interchanges Detail	Details of transactions selected in the query you ran to create the report.
Interchanges Status Summary	A summary of the transactions selected in the query you ran to create the report.
List of Interchanges	A list of interchanges as defined in the query you ran to create the report.
Pending Functional Acknowledgments	A list of transactions that have been sent but for which functional acknowledgment transactions have not been received from trading partners.

Event Logs

DataInterchange Client provides the ability to view the data from event logs. Multiple event logs can be defined in DataInterchange. Each Activity Log profile refers to a different event log, for which the data is stored in common data storage, but handled as functionally independent data.

About event logs

Event Log is a menu item under View on the tool bar. The default query for event log is a find query that displays a dialog box allowing the user to enter selection criteria for determining which event logs, user IDs, dates, etc. should be used to filter the list of event log entries. You may change the default query using the Preferences option. You can also use the query facility to create custom queries to limit Event Log searches.

Viewing event logs

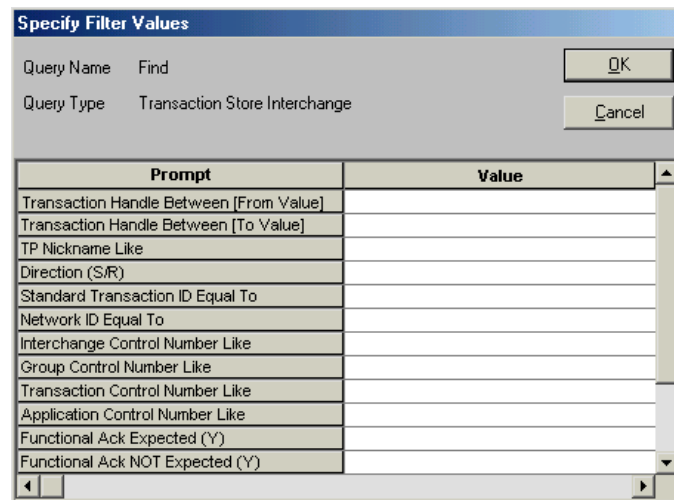
The results of an event log query are shown in the list window. Each row contains information about an event log entry, and each column contains data stored about that entry. The Event Log List window also contains the date, time, and user ID of the entry. From the displayed list of event logs, you can select an individual event log for viewing, or a group of them for deleting or printing.

To display additional columns, click the scroll bar on the bottom of the screen to scroll to the right or left. To alter the columns that display on the screen, or to change which query is executed to produce the list, click Modify Window Properties (...). To create new queries, refer to “Creating a query” on page 287.

◆ To view an entry

1. Select Event Log from the View menu on the tool bar.

The query dialog box displays.

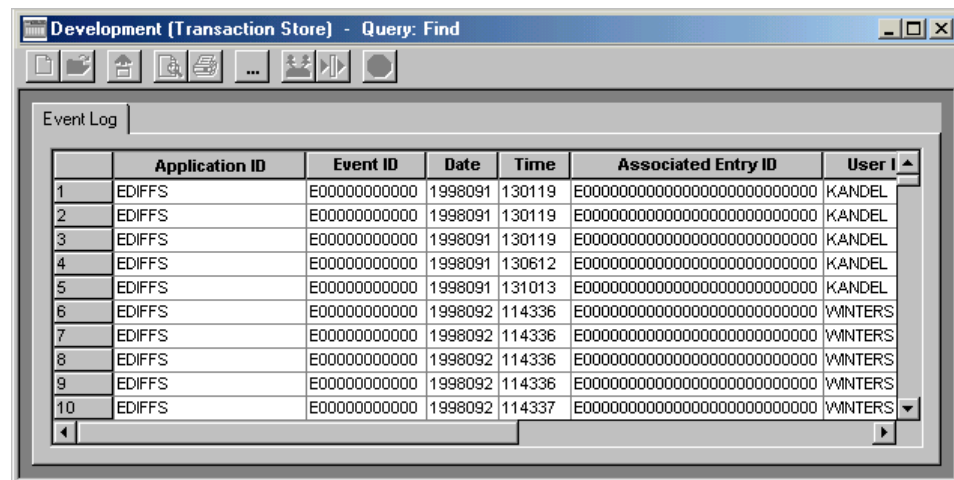


The dialog box titled "Specify Filter Values" contains the following fields and buttons:

- Query Name: Find
- Query Type: Transaction Store Interchange
- Buttons: OK, Cancel
- Table with columns: Prompt, Value

Prompt	Value
Transaction Handle Between (From Value)	
Transaction Handle Between (To Value)	
TP Nickname Like	
Direction (S/R)	
Standard Transaction ID Equal To	
Network ID Equal To	
Interchange Control Number Like	
Group Control Number Like	
Transaction Control Number Like	
Application Control Number Like	
Functional Ack Expected (Y)	
Functional Ack NOT Expected (Y)	

2. Enter your selection criteria and click OK.
3. The Event Log List window displays.



The window titled "Development (Transaction Store) - Query: Find" displays the Event Log List. The table contains the following data:

	Application ID	Event ID	Date	Time	Associated Entry ID	User I
1	EDIFFS	E00000000000	1998091	130119	E0000000000000000000000000000000	KANDEL
2	EDIFFS	E00000000000	1998091	130119	E0000000000000000000000000000000	KANDEL
3	EDIFFS	E00000000000	1998091	130119	E0000000000000000000000000000000	KANDEL
4	EDIFFS	E00000000000	1998091	130612	E0000000000000000000000000000000	KANDEL
5	EDIFFS	E00000000000	1998091	131013	E0000000000000000000000000000000	KANDEL
6	EDIFFS	E00000000000	1998092	114336	E0000000000000000000000000000000	WINTERS
7	EDIFFS	E00000000000	1998092	114336	E0000000000000000000000000000000	WINTERS
8	EDIFFS	E00000000000	1998092	114336	E0000000000000000000000000000000	WINTERS
9	EDIFFS	E00000000000	1998092	114336	E0000000000000000000000000000000	WINTERS
10	EDIFFS	E00000000000	1998092	114337	E0000000000000000000000000000000	WINTERS

4. Double-click the Event Log entry in the list window that you want to view.

The detail dialog box displays. Click  for field descriptions.

Using the 841 transaction set

The ASC-X12 Specifications/Technical Information transaction set (841) defines a way for trading partners to exchange technical information the same way they exchange EDI transactions. This technical information, which can be graphic, image, or audio, can contain binary data. The binary data can assume any value in the range hex 00 through hex FF.

In the syntax of X12 transaction sets, data elements that are separated by delimiters are combined into a segment that is identified by a segment ID and terminated with a segment delimiter. The binary data introduced by the 841 transaction set causes problems for this syntax because the binary data may contain a value that matches a segment delimiter. Translators and networks that support the 841 transaction set must have a way to identify binary data and determine its length so that it does not interfere with parsing the rest of the envelope. Special care must be taken if you want to send and receive files between DataInterchange and translators on other platforms. Not all operating systems support the record types MVS uses. For more information, see “Format specifications” on page 312.

The BIN segment ID

The binary data is identified with a BIN segment ID, which notifies the parser that data following the segment ID is binary. Although the parser must always treat the BIN segment as if it contained binary data, the segment can contain normal text. The first data element of the BIN segment contains the length of the binary data so that the parser knows the amount of data to pass without interference. The first character after the binary data should be BIN segment terminator. Any other value is a syntax error that ends parsing for the envelope.

The BIN segment ID triggers the special binary processing. Although the 841 transaction set is the only one that uses the BIN segment, binary processing is not limited to the X12 standard. In addition, DataInterchange applies this special processing to all envelope types.

Length of the BIN segment

The value in the length data element of the BIN segment can be up to 15 characters long, which means the maximum length of a BIN segment is 999,999,999,999 bytes (fifteen 9s). However, the maximum size that DataInterchange supports is 2,147,483,647 bytes, which is the maximum signed integer value that a fullword can hold. The binary segment is a repeating segment with an unlimited number of repetitions, so as far as the EDI standard is concerned, it is unlimited.

The EFI segment

For the 841 transaction set, an Electronic Format Identification (EFI) segment precedes a group of repeating BIN segments. The EFI segment provides information about the binary data file. DataInterchange provides special processing for only the following data elements in the EFI segment:

- File name
- Block type
- Record length
- Block length

Data elements such as *security technique* and *compression technique* do not receive special processing. When data compression is used, the sender must compress the data before translation, and the receiver must decompress it after translation. Any security that occurs is for the entire transaction or for the group containing the transaction. DataInterchange does not provide a special security interface for data in the BIN segments.

Use of the four data elements listed above are covered later in the discussions of send and receive processing.

The following information may help explain how DataInterchange treats two of them: file name and block type.

File name: The EDI standard essentially defines file name as free format with a maximum length of 64 characters. Its format depends on the computer operating system being used. Accordingly, DataInterchange treats this element as a fully qualified data set name, including the owner ID. A file name value with a length greater than 44 characters (the maximum length of a data set name) will be ignored.

Block type: The standard defines block type as free format with a maximum length of 4 characters. Its value indicates the organization of data in the BIN segments. Examples are fixed length, variable length, and spanned.

However, the definition does not provide any codes to represent the organizations, such as F for fixed and V for variable. In the absence of standard codes, DataInterchange interprets the block type as a string of characters that can have the following values:

Value:	Description:
A	ASA printer control characters
B	Block
F	Fixed
M	Machine code printer control characters
S	Spanned
U	Undefined
V	Variable

Send processing for the binary segment

This section covers mapping a file or an application field to a binary segment.

Mapping data from a file to a binary segment

The primary intent of the 841 transaction set and the BIN segment is to transmit files between trading partners. Therefore, DataInterchange provides a way for you to map a *file* to a data element in the EDI standard. Normally, you would map a data element to an application field. Mapping data from a file to the BIN segment requires a new application data type, FN (file name).

The FN data type has special meaning only when it is mapped to the BIN02 data element of the BIN segment. At all other times, the FN data type is treated as an alphanumeric (AN) field. The special meaning for the FN data type when mapped to the BIN02 element is as follows:

- Rather than moving the field containing data to BIN02, DataInterchange moves the entire contents of a file named by the field to the BIN02 data element.
- DataInterchange automatically supplies the length (BIN01), based on the amount of data read from the file.

Aside from the special processing described above, the mapping of an FN field is the same as any other field, including the use of a translation table or user exit routine. For example, the application may not know the data set name of a file to be transmitted. The application may know it by a coded name. A translation table or a user exit provides a way of transforming the coded name to a data set name. You can also use a literal if the application data does not contain a value at all or if the value it supplies is all blanks. Because the name of the file mapped to the BIN segment can also be specified in the EFI segment, you should use the same field for mapping both segments.

If an EFI segment is built and it contains a value in the file name data element, that file name is used if the field mapped to the BIN02 data element is all blanks and no literal value is supplied.

The format of data in an FN type field is:

type:name

where:

type

is an optional value that indicates the type of file being provided. Valid values are:

Value:	Description:
DD	Data definition name (ddname)
DS	Data set name
MQ	MQSeries Queue profile member name
VS	VSAM entry sequenced data set
TD	Transient data queue
TM	Temporary storage queue (main)
TS	Temporary storage queue (auxiliary)

VS, TD, TM, and TS are for CICS only. The default is DS in MVS and TS in CICS.

name

is either a ddname or data set name based on the value of *type*. It must conform to the conventions for its type. A ddname has a maximum length of 8. A data set name has a maximum length of 44.

name can also contain the literal &IV, which tells DataInterchange to substitute a value that represents the current binary file being processed. Each time a binary file is processed, DataInterchange increments this internal number. Thus, a specification of DD:EDIBF&IV equates to a ddname value of EDIBF1 the first time a binary file is processed, EDIBF2 the second time a binary file is processed, and so on.

The BIN segment, as mentioned earlier, can have a rather impressive length, and you may think that the entire file can be put into a single BIN segment. This is not always true. The number of BIN segments required to hold any file depends on the size of the file and the format of records in the file.

Format specifications

A file should be transmitted in a form that permits the receiver to recreate an exact copy of the file.

For MVS based systems, a file consists of individual records and each record consists of some number of characters. Records may either have a fixed format where each record has the same number of characters, or a variable format where each record has a variable number of characters. For variable length records, the number of characters in the record is maintained in a record header which is the first 4 bytes of the record and has a format of LLXX, where LL is the number of characters in the record (including the length of the LLXX field) and XX is reserved.

Files on other systems (such as Windows or OS/2) do not have record boundaries but are treated as a stream of characters with record boundaries (if any) established by the program that is reading the stream. DataInterchange provides 8 different ways to combine data from a file into BIN segments so that the receiver of the file can recreate an exact copy of the file being sent. The 8 different ways represent combinations of how records are put into binary segments (either combined or individually) and the format of the record header within the BIN segment. There are four possible header formats for records within the BIN segment:

1. No header - the data is put into the binary without a header.
2. MVS format - each record in the binary segment is preceded by a record header of the form LLXX.
3. SHORT format - each record in the binary segment is preceded by a record header of the form LL.
4. LONG format - each record in the binary segment is preceded by a record header of the form LLLL.

DataInterchange, by default, transmits all files with a fixed format by combining the records into a single binary segment without any record headers. Because the file has a fixed format, the record headers are not needed for the receiving side to reconstruct the file.

DataInterchange, by default, transmits all files with a variable format by sending each record within the file as a single binary segment. Again record headers are not used because they are not necessary to be able to reconstruct the file.

If the defaults are not satisfactory, (for example, you want to send a file with variable length records as a stream of data in a single binary segment) then a special literal value can be supplied at mapping time that overrides the defaults. This literal is specified when mapping the binary element in the BIN segment (BIN02). You can either type it in the literal field on the mapping screen, or map an application field to the BIN02, which at run time supplies the literal to DataInterchange. This literal consists of a series of keywords shown below. Except for the first keyword (BinSpec), you can specify them in any order. In the descriptions below, the keywords are shown with acceptable abbreviations in CAPS:

Keyword:	Description						
BinSpec	Signals that binary specifications follow.						
BINary(<i>x</i>)	Indicates how the records should be placed into a binary segment. Valid values for <i>x</i> are: <table> <tr> <th>Value:</th><th>Description</th></tr> <tr> <td>C</td><td>Indicates records should be combined into a single binary segment (default for fixed record lengths).</td></tr> <tr> <td>R</td><td>Indicates each record should be a binary segment (default for variable record lengths).</td></tr> </table>	Value:	Description	C	Indicates records should be combined into a single binary segment (default for fixed record lengths).	R	Indicates each record should be a binary segment (default for variable record lengths).
Value:	Description						
C	Indicates records should be combined into a single binary segment (default for fixed record lengths).						
R	Indicates each record should be a binary segment (default for variable record lengths).						
Format(<i>y</i>)	Indicates the type of header that should precede each record in the binary segment. Valid values for <i>y</i> : <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>N</td><td>No header should be used (default).</td></tr> </table>	Value	Description	N	No header should be used (default).		
Value	Description						
N	No header should be used (default).						

Keyword:	Description
V	LLXX header, where LL is the number of bytes in the record and XX is binary zeros.
S	LL header, where LL is the number of bytes in the record.
L	LLLL header, where LLLL is the number of bytes in the record.
LIMIT(<i>nnnnn</i>)	If records are being combined, then <i>nnnnn</i> is the maximum size for a binary segment. After a binary segment reaches this limit, it is stopped and a new binary segment is started. The default limit is over 2 gigabytes or the amount of virtual storage available to the program, whichever is lower.

Examples

Sending PC executable files

In order to send a Windows or OS/2 executable file received from a personal computer back to a personal computer, use the combined and no headers options:

```
BINSPEC BINARY(C) FORMAT(N)
```



NOTE: A variable length file sent with the above options cannot always be recreated, because the records have been combined and no record length information has been added. This format is needed to send data to some PC translators.

Sending variable length files to PCs

In order to send a variable length file (such as a LIST3820) to a personal computer, so that none of the record length information is lost, even if all of the records are combined into one file, use the combined and any format option other than N.

```
BINSPEC BINARY(C) FORMAT(S)
```

In order to convert a file formatted as described above back to its original format, use the same options on receive processing.

Sending a file to a system with a limit

Some systems have a limit to the size of a binary segment they can process. In order to limit the size of the data inside the BIN02 element to 1000 bytes, use the following option:

```
BINSPEC LIMIT(1000)
```



NOTE: A variable length file with records larger than 1000 bytes sent with the above options cannot always be recreated, because the records have been split and no record length information has been added. The format parameter could also be used to add record length information.

Mapping an application field to a binary segment

Although using a file name is the primary way of providing data for a binary segment, you may find it useful to provide the data by simply passing it to DataInterchange in application fields, the way all other application data is passed. If you do this, the application provides the length of the binary data and the binary data itself.

However, DataInterchange has a maximum length of 999 bytes for application fields, which is significantly less than the 999,999,999,999,999 defined by the standard or the 2,147,483,547 that DataInterchange allows. DataInterchange knows that it is dealing with binary data, and that some special processing is needed to determine the length of data being passed, gather it, and put it into a BIN segment. The easiest way to pass this binary data to DataInterchange is with a data format such as this:

```
BINARY_STRUCTURE1ST999Repeating structure
  LENGTHN0 1515-byte length field
  BINARY_DATA1ST 99Repeating structure
    BINARY_FIELDCH 999 Binary data field
```

Both BINARY_STRUCTURE1 and BINARY_DATA1 are passed separately. Now the application provides DataInterchange with a BINARY_STRUCTURE1 and as many BINARY_DATA1 structures as necessary to satisfy the length specified in the LENGTH field. DataInterchange builds one BIN segment for each occurrence of BINARY_STRUCTURE1.

The BIN segment has a length value as specified by the LENGTH field and contains data from the BINARY_DATA1 structures.

If the BINARY_DATA1 structures you provide exceed the length you specified, DataInterchange ignores the extras. If you provide fewer than needed, binary zeros are added to the BIN segment until the LENGTH value is satisfied. The number of repetitions you specify for the structures is not important. DataInterchange accepts as many as the application passes.

As another example, suppose the application wants to pass in data from the file and have structures built in the same way described above for variable-length files. The normal variable-length record format for MVS records is a binary halfword containing the length of the data followed by the data itself. The data format definition required to accomplish this is:

```
BINARY_STRUCTURE2ST 999Repeating structure
  LENGTH10 22-byte binary length
  BINARY_DATA2ST32756Repeating structure
    BINARY_FIELDCH 1 Binary data field
```

BINARY_DATA2 is defined as physically part of its parent. Its maximum use count is 32756, because that is the maximum logical record length for MVS physical sequential files.

The only difference between this example and the previous one is in the way data is passed to DataInterchange. In the first case, BINARY_STRUCTURE1 is passed followed by multiple BINARY_DATA1 structures. In the second case, only BINARY_STRUCTURE2 is passed because BINARY_DATA2 is defined as physically part of BINARY_STRUCTURE2.

There is a trade-off in choosing between the two cases. Structures always have a fixed length in DataInterchange. Therefore, when DataInterchange is presented with BINARY_STRUCTURE2, it always expects to get 32758 bytes. On the receive side, it always creates 32758 bytes. In effect, each

variable-length record is expanded to its maximum length. The method used with BINARY_STRUCTURE1 conserves main storage at the expense of making the application break up the data into 999-byte chunks, in order for DataInterchange to put it back together.

You may ask, "But a structure can repeat only 32757 times, and a field can be only 999 bytes long, for a maximum record size of 32,724,243 bytes? What if I want to pass in more than that as a single structure?" DataInterchange assumes that if the field mapped to the binary data element of the BIN segment is the first field of a structure, then the length of the *structure*, not the length of the field, is used to move data. Furthermore, if the structure is defined to be physically part of its parent, and its parent is defined to be physically part of its parent, and it has no other fields, DataInterchange uses the length of the parent structure. For example:

BINARY_STRUCTURE3	ST	999	Repeating structure
LENGTH	N0	15	15-byte length field
BINARY_DATA3	ST	10000	Repeating structure
BINARY_DATA3A	ST	10000	Repeating structure
BINARY_DATA3B	ST	10000	Repeating structure
BINARY_FIELD1	CH	999	Binary data field
BINARY_FIELD2	CH	1	Binary data field

BINARY_DATA3, BINARY_DATA3A, and BINARY_DATA3B are all defined to be physically part of their parent. Using this definition and mapping the binary data field in the standard to BINARY_FIELD1, the translator arrives at a length value of 1000 (the length of BINARY_FIELD1+BINARY_FIELD2) * 10000 (the number of times BINARY_DATA3B repeats) * 10000 (the number of times BINARY_DATA3A repeats) * 10000 (the number of times BINARY_DATA3 repeats). This yields an effective length of 10 to the 15th (1 greater than allowed by the standard).

Of course this structure could not really be used because the maximum length allowed by DataInterchange is 2,147,483,647. Nevertheless, the example illustrates the technique of creating an effective length greater than 999.

Receive processing for the binary segment

This section covers mapping a binary segment to a file or an application field.

Mapping data from a binary segment to a file

Receiving presents a set of problems that are not present on the send side. During send processing, it is reasonable to assume that senders know what they are sending, when they are sending it, and the data set names of the files they are sending. Receivers, on the other hand, may have little control over what they receive, when they receive it, and how many files they receive at any one time. Therefore, DataInterchange must be capable of creating files when necessary.

As with sending, if you want the binary data passed to you in a file rather than the application fields, map the binary data to a field in the application defined with the FN data type. You will not receive the binary data in the field; instead you will receive the name of the data set the data was written to.

On the send side, you can provide the file name in an application field buffer or as a literal value. You can also provide a literal on the receive side. If you provide a literal for an FN data type that is mapped to the binary data element of the BIN segment (BIN02), DataInterchange gives the literal

special processing. Normally, a literal mapped for receiving is used only if the standard data element is blank. The special processing for this literal is that you can use it to provide the file name and allocation parameters that DataInterchange needs to create the file on your system.

The literal has this format:

type:name parameters

where:

type

is an optional value that indicates the type of file being provided. Valid values are:

Value:	Description:
DD	Data definition name (ddname)
DS	Data set name
MQ	MQSeries Queue profile member name
VS	VSAM entry sequenced data set
TD	Transient data queue
TM	Temporary storage queue (main)
TS	Temporary storage queue (auxiliary)

VS, TD, TM, and TS are for CICS only. The default is DS in MVS and TS in CICS.

name

is either a ddname or data set name based on the value of *type*. It must conform to the conventions for its type. A ddname has a maximum length of 8 characters. A data set name has a maximum length of 44 characters.

name can also contain one of the following special literals to substitute other values in this field:

Literal:	Substitutes:
&IV	A value that represents the current binary file that is being processed. An <i>F</i> is inserted in front of this value if the IV value begins another level of qualification on the data set name.
&U	The current user ID. A period is inserted after the user ID to force another level of qualification on the data set name.
&D	The current date as a <i>yymmdd</i> value. <i>D</i> is inserted in front of this value if it begins another level of qualification on the data set name.
&T	The current time as an <i>hhmmss</i> value. A <i>T</i> is inserted in front of this value if it begins another level of qualification on the data set name.
&E	The current file name from the EFI segment. Another level of qualification is forced both before and after the EFI file name.

name is optional. If you do not provide it, the file name from the EFI segment is used. If an EFI segment does not exist or does not contain a file name field, a default data set name &U.BIN&V.D&D.T&T is used. The name that is used is returned to the application in the FN field. The maximum length of a data set name is 44 characters, and truncation occurs if necessary.

parameters

is a series of keywords and values that supply the data needed to create a new data set on your system. Except for the first keyword (ALL), you can specify them in any order. The keywords are shown with an acceptable abbreviation in CAPS:

Keyword:	Description:																
ALLocate	Signals that allocation parameters follow.																
TRacKs	Allocation unit is tracks.																
CYLinders	Allocation unit is cylinders. This is the default value if TRK or BLK is not specified.																
BLocKs(<i>b</i>)	Allocation unit is BLOCKS with an average block length of <i>b</i> .																
Space(<i>p</i> , <i>s</i>)	Primary space quantity of <i>p</i> (default value of 10). Secondary quantity of <i>s</i> (default value of 10).																
UNit(<i>vvvv</i> .)	<i>vvvv</i> is the unit name for the allocation. Default value is SYSDA.																
UCount(<i>n</i>)	<i>n</i> is the number of units for the allocation. Default value is 1.																
Lrecl(<i>l</i>)	<i>l</i> is the logical record length. Default value is 32756 or the value taken from the EFI segment.																
Blksize(<i>b</i>)	<i>b</i> is the block size. Default value is 32760 or the value taken from the EFI segment.																
Recfm(<i>xxxx</i>)	<p><i>xxxx</i> is the record format for records in the file. Default value is VB or the value taken from EFI segment. Valid values are:</p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>A</td><td>ASA printer control characters</td></tr> <tr> <td>B</td><td>Block</td></tr> <tr> <td>F</td><td>Fixed</td></tr> <tr> <td>M</td><td>Machine code printer control character</td></tr> <tr> <td>S</td><td>Spanned</td></tr> <tr> <td>U</td><td>Undefined</td></tr> <tr> <td>V</td><td>Variable</td></tr> </table>	Value	Description	A	ASA printer control characters	B	Block	F	Fixed	M	Machine code printer control character	S	Spanned	U	Undefined	V	Variable
Value	Description																
A	ASA printer control characters																
B	Block																
F	Fixed																
M	Machine code printer control character																
S	Spanned																
U	Undefined																
V	Variable																
LIKE(<i>dname</i>)	<i>dname</i> is the name of some other data set from which the LRECL, BLKSIZE, and RECFM are copied.																

DataInterchange first tries to open a given data set name. If the attempt to open the data set is successful, the file is used, and any data in the file is overlaid by the new data. If the attempt fails, DataInterchange tries to create a new file using the allocation value (or defaults) shown above. The new data set has a specification of (NEW,CATLG,CATLG), and all unused space in the data set is released when the data set is closed. For DataInterchange for CICS, if the attempt to open

the data set fails, a unique temporary storage queue is created, and the data is written to this queue. Default values are overridden by values from the EFI segment, which in turn are overridden by literal values supplied at mapping time.

If the data being received is being written to a fixed-length file, DataInterchange assumes the binary segment consists of multiple records without any record headers. If the data being received is being written to a variable-length file, DataInterchange assumes each binary segment contains data for a single record without any record headers.

DataInterchange takes data from the binary segments and adds them to the file. For each BIN segment, the amount of data defined by the logical record length of the file is extracted and written to the file. This continues until the amount of data left in the BIN segment is less than or equal to the logical record length. If the amount of data left in a binary segment is less than the logical record length, fixed-length records are padded with binary zeros; variable-length records are written with the shortened logical record length. Data from the end of one BIN segment is never combined with the data from the beginning of the next BIN segment. At a minimum, each BIN segment defines the beginning of a new record in the file.

If the defaults previously described are not desired, then the binary specifications described in “Format specifications” on page 312 (BINary and Format) can be provided as part of the allocation parameters in the receive literal value.

Mapping data from a binary segment to an application field

The considerations described for mapping from an application field to a binary segment also apply when mapping data from a binary segment to an application field. See “Mapping an application field to a binary segment” on page 315.

Data Transformation mapping commands and functions

In addition to allowing you to map a source element to a target element by dragging and dropping, the data transformation map editor includes a powerful set of commands. This allows you to use conditional logic such as if/then/else, qualify repeating elements, save values in variables, create complex expressions, and many other functions. This document describes the command language used in the data transformation map editor to perform these tasks.

When you create a map, you are basically creating a sequence of commands. Each command is attached to an object in the source document, such as a segment, data element, field, XML element, etc. Whenever the translator finds an object in the source document, it will execute any commands associated with it. You can also add commands before and after objects in the source document. The translator will execute these commands, even if the elements before and after are not found. You can create the commands by dragging and dropping a source element to a target element (or a variable) or vice versa. This results in a **MapTo** or assignment command being created. You can also create commands by right clicking on a node in the map command display to start the mapping command wizard.

Below is a description of the various commands and functions, along with other important features of the mapping language.

Map Variables

Map variables are used like variables in any programming language. They are an integral part of the DataInterchange mapping command language. Variables are used to hold and manipulate values assigned to them by the user. DataInterchange supports three types of variables: *local*, *global*, and *special variables*.

Local variables are unique to the map they are defined in. A local variable must be defined to a map before it can be used in that map. Local variables have a scope of *document* or *loop*. During translation, local variables defined with a scope of *document* will be created at the start of every document and deleted at the end of the document. Variables defined with a scope of *loop* will be created and initialized whenever a new loop iteration is started, and destroyed at the end of each

loop iteration. A loop variable does not disturb the value of another variable with the same name at another level of looping. Local variables will be maintained within the map in which they are defined. A user may add, delete and alter the properties of any local variable.

Special variables are a group of predefined variables used by DataInterchange. They function much like local or global variables, except they each have a special purpose. A user can view properties of a special variable, but no changes can be made. Special variables always start with "DI", which is reserved.

Special variables are write-only. That means you can set the value of a special variable, but cannot read it or use it in an expression.

The following special variables are supported:

Table 84. Supported special variable

Name	Scope	Type	Length	Description
DIOutFile	Document	Character	8	Specifies the name of the file to which the translated data will be written. It will override any value that was specified in the data transformation rule or data format definition.
DIOutType	Document	Character	2	Specifies the file type to which the translated data will be written to. It will override any value that was specified in the data transformation rule or data format definition.

Naming Variables

Variable names can be up to 30 characters long and may contain the characters 0-9, A-Z, a-z and the special characters "@#\$_". Special variables always start with "DI". Local variables cannot start with a number, dollar sign, or ampersand. Variables cannot be the same as any reserved word (command name, function name, or other reserved word).

Names can be in mixed case. Case is not used in determining uniqueness, for instance, the variable "MyValue" is the same as the variable "MYVALUE".



NOTE: During translation, DataInterchange may use variables whose names start with \$R to hold intermediate values. These variables are not visible in the map, but may appear in messages during translation.

Literals

The data transformation map editor uses the term "literals" differently from the send/receive map editor. In the data transformation map editor, the term "literal" refers to a static value, such as a default value. The "special literals" used in send and receive maps are replaced by "mapping commands" in data transformation maps.

Literals can be either numeric or character strings. Character strings may be contained in either single or double quotes. For example:

"This is a character literal"

'This literal contains "quotes" within'

Numeric literals are made up of a sequence of digits, and may include a decimal point. (Negative numbers may also be used, but the negative sign is considered separate from the numeric literal.) For example:

99

1.23

Comments

The data transformation map editor allows the user to enter comments by creating comment nodes in the mapping commands window. A comment group node allows the user to add several comments within a single node.

Keywords

This section identifies keywords that have special meaning to DataInterchange. In addition to the keywords listed in this section, all mapping commands, logical operators, comparison operators, and arithmetic operators are considered keywords.

False Used as a boolean value to test boolean expressions and set boolean variables.

True Used as a boolean value to test boolean expressions and set boolean variables.

Specifying a Path

Paths are used to identify a source or target element in various mapping commands. To specify a source or target path in a command, simply drag the element from the source or target tree display to the command window.

The path indicates the position of a compound element or a simple element in the source or target document definition. Paths are used in various mapping commands to identify the compound element or simple element involved in the command. Paths are syntax dependent. For instance, all of the path examples below use paths from EDI standard X12, version 3, release 6, transaction 837.

A path can include compound element identifiers and simple element identifiers. The path begins with a backslash to indicate that it starts at the root of the document. Each element of a path is separated by a backslash. The path is always terminated by two backslashes. An example of a path is:

\T 2\L 2300\S CLM 130\C C023 5\E 1332 2\\

This path translates to: table 2, loop Id 2300, CLM segment at position 130, composite data element C023 at position 5, and component data element 1332 at position 2.

The following path refers to the CLM segment at position 130 within table 2 and loop Id 2300:

\T 2\L 2300\S CLM 130\\



NOTE: Source document elements are read-only. You cannot update values in the source document. Target document elements are write-only. You may not read the value that has been assigned to an element in the target document.

Forward and Reverse References

DataInterchange always has a *current position* in the source data. The current position is the path just located and/or obtained from the source data.

In source data, DataInterchange is able to locate paths or simple elements before or after the current position. Doing this does not change the current position. The forward or reverse reference can be helpful in examining simple data or determining if a particular path exists.

For repeating items (loops, segments, elements, etc.), you may not refer to a previous or later occurrence of the item. For example, if you are currently processing the second occurrence of a repeating segment, you may not refer to an element from the first or third occurrence of the segment. If you have not yet started processing a repeating item (loop, segment, element, etc.), the first occurrence of the item is considered the current occurrence.

A more efficient option to using references is to save the information you need in variables. Then utilize the variable in place of the forward or reverse reference.

Forward and reverse references can be specified in expressions. A reference is specified by using a path.

Data Types Supported by DataInterchange mapping commands and functions

DataInterchange supports a wide variety of data types for simple elements. When the input data is parsed, the values are kept internally as one of the following data types:

character	Character data
int	Integer data
real	Floating-point data
boolean	Boolean data
binary	Binary data

Variables can also be any of these data types. Literals are always either real or character. Implicit type conversion is done by the translator when an element, variable, or literal is assigned to another type of element or variable, or is used in a function or command that expects a different type. Integer values can be converted to real or character; real values can be converted to integer or character; character values can be converted to integer or real values. If an invalid type conversion is attempted, for example trying to assign a boolean variable to a character variable, an error will be issued during the control string compilation.

Many of the commands and functions expect numeric values for their arguments or return a numeric value. All functions, commands, and expressions process numbers internally using data type real. However, since real values are implicitly converted to/from integers, integer variables and elements can be used in place of any of the real arguments or return values described below.

Expressions

Expressions are commonly used in the mapping command language. The basic form of an expression is:

token operator token [operator token [operator token [...]]]

Where:

- token* can be one of the following:
- A variable, either local, global, or special
 - A path identifying a source simple element
 - A numeric constant, such as 1024
 - A string constant, such as 'ABCD'
 - A mapping function
- operator* can be one of the following types of operators:
- Logical
 - Comparison
 - Arithmetic
 - Unary

Quotation marks must be used at the start and end of a string constant. Either single (') or double quotes (") may be used, but whichever is used to start the string constant must also be used to end it.

All variables, literals, and simple source and target elements have a specific data type. The translator will convert the value to the appropriate type for the expression if needed. If it cannot convert the value to the appropriate type because the types are incompatible (for example, boolean to binary), an error will be issued during the control string compilation. If it cannot convert the value to the appropriate type because the values are incompatible, (for example "abc" to real), the translator will issue an error or warning during translation.

Logical Operators

Data transformation maps allow logical operators: AND, OR and NOT.

AND returns a value of **True** if both conditions are true. **False** is returned when either condition is false.

OR returns a value of **True** if either condition is true. **False** is returned when neither condition is true.

NOT reverses the boolean result of an expression. For logical expressions, NOT(*expression*) will return **False** if the result of *expression* is **True**. NOT(*expression*) will return **True** if *expression* is **False**.

NOT has a short form that can be specified as an exclamation point (!). For instance, the following are valid:

```
NOT(expression)
!(expression)
NOT(myBoolFlag)
!myBoolFlag
```

Case is not relevant with the logical operators. They can be entered as uppercase, lowercase, or mixed.

Comparison Operators

Comparison operators tell the DataInterchange translator to compare two numeric objects. DataInterchange processes comparison operators the same way it handles logical operators. If the comparison is true, **True** is returned. If the comparison is false, **False** is returned. The following comparison operators are supported:

Table 85. Comparison operators

Operator	Alternate	Description
EQ	=	The first value is the same as the second
GE	>=	The first value is greater than or equal to the second value
GT	>	The first value is greater than the second value
LE	<=	The first value is less than or equal to the second value
LT	<	The first value is less than the second value
NE	!= or <>	The first value is not equal to the second value

Case is not relevant with the comparison operators. They can be entered as uppercase, lowercase, or mixed.



NOTE: The comparison operators are for numeric (integer and real) data types only. Character strings can be compared using the string comparison functions.

Arithmetic Operators

Data transformation maps allow addition, subtraction, multiplication and division of numeric values.

Data transformation maps also allow an additional operator; modulus. The modulus operator (%) will return the remainder of the first operand divided by the second.

Division by zero using the division operator ("/") will produce an error. Division by zero using the modulus operator ("%") will also produce an error.

Unary Operators

The following describes the unary (single component) operator supported by the DataInterchange translator:

Attribute	Description
-	Changes the sign of the expression it precedes. For example, if <i>var1</i> equaled 9, specifying <i>-var1</i> returns -9. It does not change the value of <i>var1</i> .

Order of Precedence

During processing, all expressions are evaluated from left to right. The order of precedence is:

1. Unary minus (-)
2. Multiple (*), Divide (/)
3. Modulus (%)
4. Addition (+), Subtraction (-)
5. Relational Operators (GT, GE, LT, LE, EQ, NE)
6. Logical NOT Operator
7. Logical AND Operator
8. Logical OR Operator

Precedence can be overridden by using parentheses within an expression. For example, $2+3*5$ will equal 17 because the multiplication is done first, then the addition. $(2+3)*5$ equals 25 because the parentheses indicate that the addition is done first, then the multiplication.

Assignment

A value can be assigned to any variable or any target simple element. This is accomplished using an assignment statement. An assignment statement has the following format:

target = expression

Where:

target Is any defined variable or simple element in the target document definition.

expression Is any valid expression.

DataInterchange will attempt to convert the result from *expression* to the same data type of *target*, if needed. If it is unable to make the conversion, an error will be issued.

Conditional Commands

If / Elself / Else / Endif

The **If** command is used to conditionally perform one or more mapping commands. The **If** command uses the following format:

```
If (condition)
    mapping commands
[Elself (condition)
    mapping commands]
[Elself (condition)
    mapping commands]
[Else
    mapping commands]
EndIf
```

Where:

condition Any valid expression that evaluates to **True** or **False**.

mapping commands One or more mapping commands.

The **If** command marks the beginning of the **If** condition block. **EndIf** is used to mark the ending of the **If** condition block.

When the **If** command is encountered by the translator, the *condition* will be evaluated. When *condition* evaluates to **True**, the mapping commands immediately following the **If** command will be executed. If *condition* evaluates to **False**, the translator will look for an **Elself** statement within the **If** condition block. If an **Elself** statement is found, its associated *condition* will be evaluated. When *condition* evaluates to **True**, the mapping commands immediately following the **Elself** statement will be executed. If *condition* evaluates to **False**, the translator will look for the next **Elself** statement within the **If** condition block. When all **Elself** statements have been tested and evaluated to **False**, the translator will look for an **Else** statement within the **If** condition block. If an **Else** statement is found, the mapping commands immediately following the **Else** statement will be executed.

The **ElseIf** and **Else** statements are optional. When present, an **Else** statement always precedes the **EndIf** statement. **ElseIf** statements always follow the **If** statement and precede the **Else** statement (if present) or the **EndIf** statement (when the **Else** statement is not present).

The translator will examine the **If** and **ElseIf** statements sequentially until an associated *condition* evaluates to **True**. Once a *condition* has evaluated to **True**, only the mapping commands associated with the corresponding **If** or **ElseIf** statement will be executed. After the mapping commands have been executed, the translator will exit the **If** condition block and resume processing after the **EndIf** statement. If no *condition* evaluates to **True**, the mappings commands associated with the **Else** statement will be executed and then translator will resume processing after the **EndIf** statement. If no *condition* evaluates to **True** and there is no **Else** statement, then no mapping commands within the **If** condition block will be executed and the translator will resume processing after the **EndIf** statement.

Commands

Mapping commands all perform a specific action as described below. The command name is not case sensitive. For example, the **Error** command could also be specified as **ERROR**. Most commands can take any expression for their arguments, as long as the expression evaluates to the appropriate type (or its result can be converted to the appropriate type). The only exception is when the argument is specified as a *sourcePath* or *targetPath*. For those commands, the argument must be a source or target path. Below is a description of each of the supported commands.

Error

The **Error** command is used to issue an error condition. It allows you to establish your own errors for a translation. Typically, the error is issued from within an **If** conditional block. The **Error** command uses the following format:

Error(real level, real code, char text)

Where:

level	Indicates the severity of the error. It should be a value 0, 1, or 2.
code	Is the unique error code that should be associated with the error. This can be any value from 5000 to 5999.
text	Is the string value, which will be included in an error message issued by DataInterchange when this command is executed.

The message is issued as a TR0026 error message. Within this message, *text* and *code* will be included.

The *level* that is assigned becomes the extended return code from the translator, and thus the JCL condition code in the DataInterchange utility. If *level* exceeds the acceptable error level specified in the document usage, then the translation will not be successful.

MapTo

The **MapTo** command can be used several different ways:

- It can be used to associate a repeating compound element in the source document definition to a corresponding repeating compound element in the target document definition.
- It can also be used to move data from a repeating simple element in the source document definition to a corresponding repeating simple element in the target document definition.
- Finally, it can be used to move data from a non-repeating simple element or variable in the source document definition to a corresponding simple element in the target document definition.

A **MapTo** command is generated when you drag a source element to a target element or a target element to a source element. The **MapTo** command uses the following format:

MapTo(targetpath [, expression])

Where:

- targetpath** Identifies the path in the target document definition that is being mapped.
- expression** Is the expression whose value will be placed into the simple element identified by *targetpath*. This expression can be any data type, as long as it is compatible with the target element that it is being mapped to.

When placed on a repeating or nonrepeating compound element and only *targetpath* is specified, the compound element in the source document definition will be associated with the compound element identified by *targetpath* in the target document definition. Each occurrence of the source compound element will result in the creation of a corresponding target compound element.

When placed on a repeating simple element and only *targetpath* is specified, the simple element in the source document definition will be associated with the simple element identified by *targetpath* in the target document definition. Each occurrence of the source simple element will result in the creation of a corresponding target simple element and data being moved from the source simple element to the target simple element. This is equivalent to the assignment statement:

targetpath = (current source element)

When placed on a non-repeating simple element and only *targetpath* is specified, the simple element in the source document definition will be associated with the simple element identified by *targetpath* in the target document definition. Encountering the source simple element will result in the creation of a corresponding target simple element and data being moved from the source simple element to the target simple element. This is equivalent to the assignment statement:

targetpath = (current source element)

When *targetpath* is specified with *expression*, the result of *expression* will be moved to the simple element identified by *targetpath* in the target document definition. This is equivalent to the assignment statement:

targetpath = expression

Qualify / Default

The **Qualify** command is used on repeating compound or repeating simple elements in the source document definition. It is used to indicate that a specific iteration or iterations of the elements are to be handled differently than other iterations of the element. For instance, you may want to say that the first iteration of a loop is handled differently than all other iterations of the loop. The **Qualify** command uses the following format:

Qualify(bool boolExpr)

Where:

boolExpr Is the boolean value or expression that, when True, indicates the mapping commands within the qualification will be executed by the translator.

boolExpr can indicate if this is a particular occurrence of a repeating compound or simple element. Values of variables or simple elements in the source document definition can also be checked to determine when to execute the qualification. For instance, you can check to see if a specific simple element has a value of "ABC". If it does, the mapping commands within the qualification will be executed by the translator.

Example 1:

Qualify (StrComp(Table 1 Loop 1 M 98, "ZZ") = 0)

In this example, the value contained in simple element "Table 1 Loop 1 M 98" is compared to the literal "ZZ". If the function returns 0 (values are equal), then the mapping commands within the qualification will be executed.

Example 2:

Qualify(Occurrence() = 2 OR Occurrence() = 3)

In this example, the mapping commands within the qualification will be executed if it is the second or third occurrence of the simple or compound element.

Example 3:

Qualify(Occurrence() = 2 AND StrComp(Table 1 Loop 1 M 98, "ZZ") = 0)

In this example, the mapping commands within the qualification will be executed if it is the second occurrence of the current compound source element and simple element "Table 1 Loop 1 M 98" is "ZZ".

The Occurrence and StrComp functions are described in more detail in the Functions section. Any boolean expression or function may be used as the expression on the **Qualify** command.

The **Default** command may be used to specify the commands that should be executed if none of the **Qualify** expressions evaluate to True.

SetProperty

The **SetProperty** command is used to set a special processing property of the target message. Various special properties are defined to control things such as the EDI envelope fields or the XML prolog. The **SetProperty** command uses the following format:

```
SetProperty(char propertyName, char propertyValue)
```

Where:

propertyName Is the property name to be set in the target message.

propertyValue Is the value that it should be set to.

See the section on "Message Properties" for a list of the property names that can be specified.

Functions

All functions take 0 or more arguments as input and return a value. The data type of the return value, as well as the number and data types of the arguments varies from one function to the next. Appropriate type conversions will be done implicitly if needed (and possible), just as they are done for the expressions.

Some functions have optional parameters. If the optional parameters are omitted from the function call, a default value will be used for that argument.

Most functions can take an expression as an argument, as long as the result of the expression is (or can be converted to) the correct data type. For example, the Char function converts a value to a character string. The command:

```
Var1 = Char ( 1 + 2 )
```

Is equivalent to :

```
Var1 = Char ( 3 )
```

The only time you cannot use an expression as an argument is when the argument is identified as a source or target path. For example, you may not pass an expression to the **Created** function.

The input arguments for a function are never modified by the function. The return value and argument list for each of the functions is described below.

The function names are not case sensitive. The function name **Char** is the same as **CHAR**.

Char

The **Char** function returns a character representation of a numeric value. The **Char** function uses the following format:

```
char Char(real value)
```

Where:

value Is the value that will converted to a character value. It can be data type real or int. (Data type character is allowed, but does not cause any conversion.)

Results:

The character representation of *value*.

This function is not normally needed, since the conversion is generally done implicitly. However, it can be used to force the conversion to a character value, or to clarify when the conversion is done.

Concat

The **Concat** function concatenates one character string to another. The **Concat** function uses the following format:

```
char Concat(char value1, char value2)
```

Where:

value1 Is the value that is appended to.

value2 Is the value that is appended to the end of *value1*.

Results:

The concatenated string.

Example:

```
chVar = Concat("abc", "def")
```

Would set chVar to "abcdef".

Created

The **Created** function is used to determine if the specified path was created in the target data. The **Created** function uses the following format:

```
boolean Created(targetPath)
```

Where:

targetPath Identifies the path in the target document that is being checked.

Results:

True When the specified path has been created in the target data

False When it has not been created in the target data.

This function can only be used with paths in the target document definition. If the element is within a repeating compound element (such as a repeating loop or segment), then only the current (or most recent) iteration is searched for the *targetPath*.

This function can only be used with paths that can occur in the target document definition.

Example:

```
If (!Created(\T 2\L 2300\S CLM 130\C C023 5\\))
    mapping command
EndIf
```

In this example, if path “\T 2\L 2300\S CLM 130\C C023 5\\” has not been created, then *mapping command* will be executed by the translator.

Date

The **Date** function returns the system date as a character string in the format *yyyymmdd*. The **Date** function uses the following format:

```
char Date()
```

Results:

The current system date in the format *yyyymmdd*.

DateCnv

The **DateCnv** function is used to assist in converting one date format to another. The **DateCnv** function uses the following format:

```
char DateCnv(char sourceDate, char frommask, char tomask)
```

Where:

sourceDate Is the string that contains the source date.

frommask Is the mask that identifies the format of the date in *sourceDate*.

tomask Is the mask that identifies the format of the date desired in the result string.

Results:

The converted date as a character string.

The mask format for the DateCnv function is the same as the mask format used in send and receive maps. See Appendix C, “Date conversion special operators,” on page 377 for information about the values that can appear in the from and to mask.

Find

The **Find** function is used to determine if a string is contained within another string. The **Find** function uses the following format:

```
Find(char toSearch, char searchVal, real startPos)
```

Where:

toSearch Is the string to be searched.

searchVal Is the string that is to be searched for within *toSearch*.

startPos Is the starting position within *toSearch* where the search will be started.

Results:

The position where *string* occurs in the simple element or variable.

Zero is returned when:

- *searchVal* is not found within *toSearch*
- The *toSearch* value has less than *startPos* characters
- *toSearch* is empty

startPos is one-based, so a *startPos* of one will start the search from the beginning of the string, a position of two begins the search at the second character, and so on.

Examples

Assuming *var1* contains the string "ABCDEFGH",

```
Find(var1, "CDE", 1) will return 3
Find(var1, "CDE", 3) will return 3
Find(var1, "CDE", 4) will return 0
Find(var1, "CDE", 0) will cause an error to be issued
```

Found

Found is used to determine if a specified path exists in the source data. The **Found** function uses the following format:

```
boolean Found(sourcePath)
```

Where:

sourcePath Identifies the path that is being checked.

Results:

True When the specified path is found in the source data

False When it is not in the source data.

This function can only be used with paths in the source document definition. If the element is within a repeating compound element (such as a repeating loop or segment), then only the current iteration is searched for the *sourcePath*.

Example:

```
If (!Found(\T 2\L 2300\S CLM 130\C C023 5\))
    mapping command
EndIf
```

In this example, if path "\T 2\L 2300\S CLM 130\C C023 5\\" is not found, then *mapping command* will be executed by the translator.

GetProperty

The **GetProperty** function is used to get a property of the source message. This can be used to retrieve information such as EDI envelope header elements. The **GetProperty** function uses the following format:

```
char GetProperty(char propertyName)
```

Where:

propertyName Is the name of the property you want to retrieve. See "Message Properties" for a list of properties that you can retrieve.

Results:

The value associated with the specified message property.

If the *propertyName* is not set in the source message, an empty string is returned.

Example:

```
var1 = GetProperty("ISA04")
```

Will set var1 to the value that was in the ISA04 element in the X12 ISA segment.

HexEncode

The **HexEncode** function is used to encode binary data to a character format. The **HexEncode** function uses the following format:

```
char HexEncode(binary binValue)
```

Where:

binValue Is the binary value to be encoded.

Results:

An encoded character string that represents the binary data

Each byte of the binary value will be encoded as two characters in the resulting string. For example, if the input is a 4 byte binary value: 0x01020A0B, the result would be the 8-character string: "01020A0B".

HexDecode

The **HexDecode** function is used to decode a character string that contains encoded binary data. The **HexDecode** function uses the following format:

binary HexDecode(char encodedStr)

Where:

encodedStr Is an encoded string.

Results:

A binary value derived from the encoded character string

Each byte of the binary value will be derived from two characters in the encoded string. For example, if the input is the 8-character string: "01020A0B", the result would be a 4 byte binary value: 0x01020A0B.

IsEmpty

The **IsEmpty** function is used to determine if a character string contains any data. The **IsEmpty** function uses the following format:

boolean IsEmpty(char value)

Where:

value Is the value to check.

Results:

True When *value* is empty (set to an empty string: "").

False When *value* contains data.

If the value is a source element, this will also return **True** if the element is not found. The expression:

IsEmpty(value)

is equivalent to the following string comparison expression:

(StrComp(value, "") = 0)

Example:

```
If (not IsEmpty(var1))  
    mapping command  
EndIf
```

In this example, if *var1* contains a value other than an empty string, then *mapping command* will be executed by the translator.

Left

The **Left** function is used to obtain the first few characters from a string. The **Left** function uses the following format:

```
char Left(char stringIn, real length)
```

Where:

stringIn Is the input string that characters are copied from.

length Is the number of characters to be copied.

Results:

A character string containing the leftmost characters of *stringIn*.

Only available characters from *stringIn* will be copied. Therefore, the returned string will contain *length* characters unless *stringIn* has less than *length* characters. See Example 2 below.

Example 1:

```
var1 = left("ABCDEFGH",3)
```

After this code has been executed, *var1* will equal "ABC".

Example 2:

```
var1 = left("ABCD",6)
```

After this code has been executed, *var1* will equal "ABCD" because there are not enough characters in the *stringIn* argument to fulfill the request.

Length

The **Length** function is used to determine the length of a string. The **Length** function uses the following format:

```
real Length(char value)
```

Where:

value Is a string, usually a variable or a simple element in the source document definition.

Results:

The length of the character string *value*.

Example:

```
If (Length(var1) < 5)  
    mapping command  
EndIf
```

In this example, if *var1* has a length that is less than 5, then *mapping command* will be executed by the translator.

Lower

The **Lower** function is used to convert a string to lowercase. The **Lower** function uses the following format:

char Lower(char value)

Where:

value Is a string, usually a variable or a simple element in the source document definition.

Results:

A copy of *value*, with all characters converted to lowercase.

Example:

```
var1 = Lower("ABC")
```

In this example, *var1* will be set to "abc".

Number

The **Number** function converts a character value to a real value. The **Number** function uses the following format:

real Number(char value)

Where:

value Is the character value that will be converted to a real value. (Data type real or int are also allowed, but do not cause any conversion.)

Results:

The (data type) real representation of *value*.

This function is not normally needed, since the conversion is generally done implicitly. However, it can be used to force the conversion to a numeric value, or to clarify when the conversion is done.

NumFormat

The **NumFormat** function formats a real number or integer as a character string using a specified number of decimal positions. Unused digits are either truncated or rounded. The **NumFormat** function uses the following format:

```
char NumFormat(real value, real decimals[, char flag])
```

Where:

- | | |
|-----------------|---|
| value | Is the number that will be formatted as a character string. |
| decimals | Is the number of decimal positions that should be included in the returned value. |
| flag | Indicates whether unused digits are rounded or truncated. Specify "ROUND" to round unused digits or "TRUNCATE" to truncate unused digits. These can be shortened to "R" or "T". If no flag is specified, the value defaults to "ROUND". |

Results:

The character representation of *value*, containing the specified number of decimal places.

Example 1:

```
var1 = NumFormat(1234.567, 2, "T")
```

This example will set *var1* to "1234.56".

Example 2:

```
var1 = NumFormat(1234.567, 2, "R")
```

This example will set *var1* to "1234.57".

Occurrence

The **Occurrence** function is used to obtain the current occurrence number of:

- The current repeating compound element or simple element
- A specific repeating compound element or simple element

Occurrence can be shortened to **Occur**. The **Occurrence** function uses the following format:

```
real Occurrence([sourcePath])
```

Where:

- | | |
|-------------------|--|
| sourcePath | Is the path that refers to a compound or simple element in the source document definition. If no <i>sourcePath</i> is specified, the current compound or simple element in the source document is assumed. |
|-------------------|--|

Results:

The current occurrence number of the specified element

When **Occurrence** is specified with no parameters, it will return the occurrence number of the current compound or simple element. When *sourcePath* is specified, it will return the current occurrence number of the specified repeating compound element or simple element. The

specified element must be the current element, a parent of the current element, or one of the other elements on the path to the current element (such as the grandparent of the current element)

Example 1:

```
If (Occurrence (\Table 2\10 M PO1 Loop\)) = 2)
    mapping command
EndIf
```

In this example, if this is the second occurrence of the PO1 loop, then *mapping command* will be executed by the translator.

Example 2:

```
If (Occurrence() != 1)
    mapping command
EndIf
```

In this example, if this is not the first occurrence of the current source element, then *mapping command* will be executed by the translator.

Overlay

The **Overlay** function is used to overlay a portion of a string with data from another string. The **Overlay** function uses the following format:

```
char Overlay(char string1, char string2, real position [, real length])
```

Where:

string1	Evaluates to a string that will be overlaid with data from <i>string2</i>
string2	Evaluates to a string that will overlay data in <i>string1</i>
position	Indicates the position where the overlay will begin in <i>string1</i>
length	Is an optional parameter that indicates the number of characters to be overlaid in <i>string1</i> . If omitted or less than 0, all of the characters following <i>position</i> will be overlaid until all characters from <i>string2</i> have been used.

Results:

The resulting string *position* is one-based, so a position of one will begin overlaying characters at the beginning of *string1*, a position of two begins overlaying at the second character, etc. If the length of *string1* has less than *position* characters, then blanks will be used to fill the positions between the end of *string1* and the starting position.

Example:

```
var1 = overlay("ABCDEFGF", "WXYZ", 3, 3)
```

After this code has been executed, *var1* will equal "ABWXYFG".

Right

The **Right** function is used to obtain the last few characters from a string. The **Right** function uses the following format:

```
char Right(char stringIn, real length)
```

Where:

stringIn Is the input string that characters are copied from.

length Indicates the number of characters to be copied.

Results:

A character string containing the rightmost characters of *stringIn*.

Only available characters from *stringIn* will be copied. Therefore, the returned string will contain *length* characters unless *stringIn* has less than *length* characters. See Example 2 below.

Example 1:

```
var1 = Right("ABCDEFGH",3)
```

After this code has been executed, *var1* will equal "EFG".

Example 2:

```
var1 = Right("ABCD",6)
```

After this code has been executed, *var1* will equal "ABCD" because there are not enough characters in the *stringIn* argument to fulfill the request.

Round

The **Round** function rounds the input value to the specified number of decimal places. The **Round** function uses the following format:

```
real Round(real value, real decimals)
```

Where:

value Is the value that will be rounded.

decimals Is the number of decimals to round to.

Results:

A real number rounded to the specified number of decimal places.

For instance, Round(4321.556, 2) will return a result of 4321.560000. Round(4321.556, 0) will return a result of 4322.000000.



NOTE: This function returns a numeric (real) value. If the return value will be assigned to a character string and trailing zeros are to be removed, the NumFormat function should be used.

StrComp

The **StrComp** function is used to compare two character strings. The **StrComp** function uses the following format:

```
real StrComp(char string1, char string2)
```

Where:

string1 Is the first string to be compared.
string2 Is the second string to be compared.

Results:

-1 If *string1* < *string2*
0 If the strings are equal
1 If *string1* > *string2*

Example:

```
If (StrComp(var1, "ABC") = 0)  
mapping command  
EndIf
```

In this example, if *var1* is "ABC", then *mapping command* will be executed by the translator.

StrCompI

The **StrCompI** function is used to compare two character strings without regard to case. The **StrCompI** function uses the following format:

```
real StrCompI(char string1, char string2)
```

Where:

string1 Is the first string to be compared.
string2 Is the second string to be compared.

Results:

-1 If *string1* < *string2* (case insensitive)
0 If the strings are equal (case insensitive)
1 If *string1* > *string2* (case insensitive)

Example:

```
If (StrCompI(var1, "ABC") = 0)  
mapping command  
EndIf
```

In this example, if *var1* is "ABC", "abc", "Abc", etc., then *mapping command* will be executed by the translator.

StrCompN

The **StrCompN** function is used to compare the first *length* characters of two character strings. The **StrCompN** function uses the following format:

real StrCompN(char string1, char string2, real length)

Where:

- string1** Is the first string to be compared.
- string2** Is the second string to be compared.
- length** Is the number of characters to compare.

Results:

- 1** If the first *length* characters of *string1* < the first *length* characters of *string2*
- 0** If the first *length* characters of the two strings are equal
- 1** If the first *length* characters of *string1* > the first *length* characters of *string2*

Example:

```
If (StrCompN(var1, "ABC", 3) = 0)
    mapping command
EndIf
```

In this example, if the first 3 characters of *var1* are "ABC", then *mapping command* will be executed by the translator.

StrCompNI

The **StrCompNI** function is used to compare the first *length* characters of two character strings without regard to case. The **StrCompNI** function uses the following format:

real StrCompNI(char string1, char string2, real length)

Where:

- string1** Is the first string to be compared.
- string2** Is the second string to be compared.
- length** Is the number of characters to compare.

Results:

- 1 If the first *length* characters of *string1* < the first *length* characters of *string2* (case insensitive).
- 0 If the first *length* characters of the two strings are equal (case insensitive).
- 1 If the first *length* characters of *string1* > the first *length* characters of *string2* (case insensitive).

Example:

```
If (StrCompNI(var1, "ABC", 3) = 0)
    mapping command
EndIf
```

In this example, if the first 3 characters of *var1* are "ABC", "abc", "Abc", etc., then *mapping command* will be executed by the translator.

SubString

The **SubString** function is used to obtain a portion of a string. The **SubString** function uses the following format:

```
char SubString(char string, real position [, real length])
```

Where:

- string** Is the string value.
- position** Indicates the starting position of the substring within the *string*.
- length** Is an optional parameter that indicates the number of characters to be copied into the substring. If this argument is omitted or the length is less than 0, all remaining characters to the end of the string are copied.

Results:

The resulting character string returned by the function.

position is one-based, so a *position* of one will start copying characters from the beginning of *string*, a position of two begins copying data from the second character, etc.

Only available characters from the *string* value will be copied. Therefore, the substring will contain *length* characters unless *string* has less than *position* plus *length* minus one characters. See example 2 below. If the *string* value has less than *position* characters, an empty string will be returned.

Example 1:

```
var1 = substring("ABCDEFGH",3,3)
```

After this code has been executed, *var1* will equal "CDE".

Example 2:

```
var1 = substring("ABCD",3,3)
```

After this code has been executed, *var1* will equal "CD" because there are not enough characters in the *string* argument to fulfill the request.

Time

The **Time** function returns the system time as a character string in the format *hhmmss*. The **Time** function uses the following format:

```
char Time()
```

Results:

The system time as a character string in the format *hhmmss*.

Translate

The **Translate** function attempts to locate a specified string in a translation table. If the string is found in the table, the corresponding value in the table is returned. The **Translate** function uses the following format:

```
char Translate(char table, char direction, char valueIn [, boolean error[, char default]])
```

Where:

table	Is the string that identifies the translation table.
direction	Is either "SOURCE" or "TARGET". This can be shortened to either "S" or "T".
valueIn	Is the string value that will be looked up in the translation table.
error	Indicates whether a not found condition will result in a warning message. If True , a warning message will be issued if the value is not found in the table. If False , no warning message will be issued. The default is True .
default	Is a string that provides a default value that is returned when <i>valueIn</i> is not found in the translation table.

Results:

The resulting character string.

table must be a valid *Forward Translation* table. If *table* does not exist, a warning will be issued and it will be treated as a not found condition.

A translation table contains two values for each entry in the table, one called the *Local Value* and the other called the *Standards or Trading Partner Value*. When *direction* is specified as "SOURCE", an attempt is made to find *valueIn* in the *Local Value* column in the table. If it is found, the value from the corresponding *Standards or Trading Partner Value* column is returned by the function. When *direction* is specified as "TARGET", an attempt is made to find *valueIn* in the *Standards or Trading Partner Value* column in the table. If it is found, the value from the corresponding *Local Value* column is returned by the function.

In a *Forward Translation* translation table, the *Local Value* column must be unique. The *Standards or Trading Partner Value* column does not have to be unique. A *Forward Translation* translation table used in the **Translate** command will always produce predictable results when SOURCE is specified as the direction. The results are also always predictable when TARGET is specified as the direction if the values in the *Standards or Trading Partner Value* column are unique. If the values are not unique in the *Standards or Trading Partner Value*, the results may not be predictable when TARGET is specified as the direction.

If *valueIn* is not found in the translation table, a warning message will be issued if *error* is **True**. Also, if *valueIn* is not found in the translation table, the *default* value will be returned if specified. If a *default* value is not specified, an empty string ("") is returned.

Example:

```
var1 = Translate("MYTABLE", "S", stringVar, False, "ABC")
```

This command will get the data contained in variable *stringVar* and attempt to locate that value in the *Local Value* column of translation table "MYTABLE". If the value is not found in the translation table, a warning message is not issued and *var1* will be set to "ABC". If the value is found in the translation table, *var1* will be set to the value in the corresponding *Standards or Trading Partner Value* column.

TrimLeft

The **TrimLeft** function is used to remove unwanted leading characters from a string. The **TrimLeft** function uses the following format:

```
char TrimLeft(char value, [char trimList])
```

Where:

- value** Is the string value that is to be trimmed.
- trimList** Is an optional string value that includes all the characters that are to be removed. If this is not specified, all leading whitespace characters are removed.

Results:

The resulting character string.

All characters contained in *trimList* will be removed from the beginning of the *value* string. The operation ends when the first character not contained in *trimList* is encountered. If *trimList* is omitted, whitespace is removed from the beginning of the *value* string. Whitespace includes blanks, carriage returns, line feeds and tab characters.

Example 1:

```
var1 = TrimLeft(" ABCD")
```

In this example, *var1* will equal "ABCD" after the function has been executed.

Example 2:

```
var1 = TrimLeft("ZZYDABCD", "DYZ")
```

In this example, *var1* will equal "ABCD" after the function has been executed. All characters "D", "Y" and "Z" will be removed from the beginning of *value* until an unspecified character is encountered.

TrimRight

The **TrimRight** function is used to remove unwanted trailing characters from a string. The **TrimRight** function uses the following format:

```
char TrimRight(char value, [char trimList])
```

Where:

- | | |
|-----------------|--|
| value | Is a string value that is to be trimmed. |
| trimList | Is an optional string value that includes all the characters that are to be removed. If this is not specified, all trailing whitespace characters are removed. |

Results:

The resulting character string.

All characters contained in *trimList* will be removed from the end of the *value* string. The operation ends when the first character not contained in *trimList* is encountered. If *trimList* is omitted, whitespace is removed from the end of the *value* string. Whitespace includes blanks, carriage returns, line feeds and tab characters.

Example 1:

```
var1 = TrimRight("ABCD  ")
```

In this example, *var1* will equal "ABCD" after the function has been executed.

Example 2:

```
var1 = TrimRight("ABCDZZY", "AYZ")
```

In this example, *var1* will equal "ABCD" after the function has been executed. All characters "A", "Y" and "Z" will be removed from the end of the string until an unspecified character is encountered.

Truncate

The **Truncate** function is used to truncate a number to the specified number of decimal places. The **Truncate** function uses the following format:

```
real Truncate(real value, real decimals)
```

Where:

- | | |
|-----------------|---|
| value | Is the value that will be truncated. |
| decimals | Is the number of decimals to truncate to. |

Results:

A real number truncated to the specified number of decimal places.

The real number *value* is truncated to *decimals* decimal places - no rounding occurs. For instance, **Truncate**(4321.556, 2) will return a result of 4321.550000. **Truncate**(4321.556, 0) will return a result of 4321.000000.



NOTE: This function returns a numeric (real) value. If the return value will be assigned to a character string and trailing zeros are to be removed, the NumFormat function should be used.

Upper

The **Upper** function is used to change a string to upper case. The **Upper** function uses the following format:

```
char Upper(char value)
```

Where:

value Is a string, usually a variable or a simple element in the source document definition.

Results:

A copy of *value*, with all characters converted to uppercase.

Example:

```
var1 = Upper("abc")
```

In this example, *var1* will be set to "ABC".

Validate

The **Validate** function attempts to locate a specified string in a validation table. The **Validate** function uses the following format:

```
boolean Validate(char table, char value [, boolean error])
```

Where:

table Is a string that identifies the validation table.

value Is a string value that will be looked up in the validation table.

error Indicates whether a not found condition will result in a warning message. If **True**, a warning message will be issued if the value is not found in the table. If **False**, no warning message will be issued. The default is **True**.

Results:

True If the string resulting from *value* is found in the validation table.

False Is returned if the string is not found in the validation table

table must be a valid *Code List* table. If *table* does not exist, a warning message will be issued and it will be treated as a not found condition.

The *value* string is used to search the validation table. If the string is found in the validation table, **True** is returned by the function. If the string is not found in the validation table, **False** is returned.

Example:

```
If (Validate("MYTABLE",\Table 1\20 M BEG\1 M 353\))
```

```
    mapping command
```

```
Else
```

```
    Error(1, 5001, , "Invalid value specified")
```

```
EndIf
```

This code will get the data contained in simple element “\Table 1\20 M BEG\1 M 353\” and attempt to locate that value in the validation table “MYTABLE”. If the value is found in the validation table, the *mapping command* will be executed. If the value is not found in the validation table, an error is issued with error code 5001.

Message Properties

Target Document Properties

The following table identifies properties that are specific to target documents. These properties can be set using the SetAttribute command.

Attribute	Description
DIProlog	Used to override the default XML prolog in the target document.

EDI Envelope Standard Generic Properties

The EDI envelope standard generic properties allow you to get or set information in the EDI envelope standard segments. They have generic names that may apply to any of the EDI standard types. For example, IchgSndrId is used for the interchange sender ID, regardless of the EDI standard type.

These properties may be available when the source document is received within an EDI envelope standard. The properties can be obtained using the GetProperty function. If you request a property that is not present, an empty string is returned.

These values may also be set for the target EDI document using the SetProperty command. If set, they will override the values specified in the envelope profile.

The following table identifies the EDI envelope standard generic properties.

Attribute	Description
IchgCtlNum	Interchange control number
IchgSndrId	Interchange sender ID
IchgRcvrId	Interchange receiver ID
IchgDate	Interchange date
IchgTime	Interchange time
IchgPswd	Interchange password

Attribute	Description
IchgUsgInd	Interchange usage indicator
IchgAppRef	Interchange application reference
IchgVerRel	Interchange version/release
IchgGrpCnt	Number of groups in interchange
IchgCtlTotal	Control total from interchange trailer segment
IchgTrxCnt	Number of transactions in interchange
GrpCtlNum	Group control number
GrpFuncGrpId	Functional group ID
GrpAppSndrId	Group application sender ID
GrpAppRcvrId	Group application receiver ID
GrpDate	Group date
GrpTime	Group time
GrpPswd	Group password
GrpVer	Group version
GrpRel	Group release
GrpTrxCnt	Number of transactions in group
TrxCtlNum	Transaction control number
TrxCode	Transaction code
TrxVer	Transaction version
TrxRel	Transaction release
TrxSegCnt	Number of segments in the transaction

EDI Envelope Standard Specific Properties

The EDI envelope standard specific Properties also allow you to get or set information in the EDI envelope standard segments. These have names that are specific to a particular EDI standard, and specify a particular segment and element number. For example, ISA04 is used for the fourth element of the ISA segment in an X12 envelope.

All of these properties are used to obtain the value of common data contained in an EDI envelope standard segment. These properties may be available when the source document is received within an EDI envelope standard. These properties can be obtained using the `GetProperty` function. If you request a property that is not present, an empty string is returned.

These values may also be set for the target EDI document using the `SetProperty` command. If set, they will override the values specified in the envelope profile.

The following table identifies the EDI envelope standard specific properties.

Attribute	Description
ISA nn	“nn” is 01 through 16. Use attribute ISA01 through ISA16 to obtain information from each element in the ISA segment contained in the envelope.
GS nn	“nn” is 01 through 08. Use attribute GS01 through GS08 to obtain information from each element in the GS segment contained in the envelope.
ST nn	“nn” is 01 or 02. Use attribute ST01 and ST02 to obtain information from each element in the ST segment contained in the envelope.
SE nn	“nn” is 01 through 02. Use attribute SE01 and SE02 to obtain information from each element in the SE segment contained in the envelope.
GE nn	“nn” is 01 through 02. Use attribute GE01 and GE02 to obtain information from each element in the GE segment contained in the envelope.
IEA nn	“nn” is 01 through 02. Use attribute IEA01 and IEA02 to obtain information from each element in the IEA segment contained in the envelope.
UNB nn	“nn” is 01 through 18. Use attribute UNB01 through UNB18 to obtain information from each element in the UNB segment contained in the envelope.
UNG nn	“nn” is 01 through 13. Use attribute UNG01 through UNG13 to obtain information from each element in the UNG segment contained in the envelope.
UNH nn	“nn” is 01 through 09. Use attribute UNH01 through UNH09 to obtain information from each element in the UNH segment contained in the envelope.
UNT nn	“nn” is 01 through 02. Use attribute UNT01 and UNT02 to obtain information from each element in the UNT segment contained in the envelope.
UNE nn	“nn” is 01 through 02. Use attribute UNE01 and UNE02 to obtain information from each element in the UNE segment contained in the envelope.
UNZ nn	“nn” is 01 through 02. Use attribute UNZ01 and UNZ02 to obtain information from each element in the UNZ segment contained in the envelope.

Attribute	Description
BGnn	“nn” is 01 through 07. Use attribute BG01 through BG07 to obtain information from each element in the BG segment contained in the envelope.
EGnn	“nn” is 01 through 04. Use attribute EG01 through EG04 to obtain information from each element in the EG segment contained in the envelope.

Advanced send and receive mapping

This appendix describes how to map your application data to an EDI standard transaction set. This appendix assumes that you are familiar with your application data layout and have already defined your application data to DataInterchange. It also assumes you are familiar with the EDI standard transaction set you are using.

Using accumulators

You can use accumulators to count occurrences of an event, such as counting the detail line items in a purchase order to provide a hash total. You can also use accumulators to total fields for control purposes, such as totaling the quantity field to cross check the number of items sent or received.

Accumulators can apply to individual transactions or to all transactions in a translation session.

Valid accumulators are:

Name:	Description:
T0--T9	Transaction accumulators that apply to one transaction. They are reset at the beginning of each transaction.
G0--G9	Global accumulators that apply to an entire translation session. They are reset at the beginning of each translation session.

Basic accumulator actions are adding, incrementing, zeroing, and mapping. These actions can be combined as follows:

Action:	Description:
A	For sending, adds to the accumulator the EDI standard data just mapped. For receiving, adds to the accumulator the EDI standard data received. The data must be numeric.
I	Increments the accumulator by 1.
M	Maps the accumulator to the EDI standard data element (send) or the application field (receive).
Z	Zeroes the accumulator.
MI	Maps the accumulator, then increments it.
IM	Increments the accumulator, then maps it.
MZ	Maps the accumulator, then zeroes it.
AM	Adds to the accumulator, then maps it.
MA	Maps the accumulator, then adds to it.

For send transactions, accumulator actions will not be processed unless one of the following occurs:

- Data is generated for the EDI standard data element.
- The variable is mapped.

For receive transactions, accumulator actions will not be processed unless one of the following occurs:

- The data element associated with the accumulator is received.
- The accumulator is mapped and at least the segment containing the data element is received.

To map both an accumulator and a received value for the same data element, map the data element to a field. Then use the Repeat button in the Mapping Data Element Editor to create another mapping occurrence of the data element. In the new mapping occurrence of the data element, map from the accumulator to a field.

Accumulators have the following limitations:

- Each accumulator holds a maximum of 31 digits.
- Each data element mapping can support up to 4 accumulators.
- You can use up to 10 transaction accumulators for a transaction.
- You can use up to 10 global accumulators for an entire translation session.

Using literals

For send transactions, literals let you supply data that is not in your application data. For receive transactions, literals let you supply data required by your application that is not received with the standard data. DataInterchange offers a variety of options for using literals. This section provides details on each option available, including rules for use, keywords, and syntax.

Transactions are always processed starting with the first data element of the first segment and proceeding to the last data element of the last segment, as documented in the standard. This is true for both send and receive processing.

Using literals for send mapping

When mapping literals for sending data, you can map both an application field and a literal to the same data element. The literal value is used if one of the following conditions occur:

- The record containing the application data field was provided, but the application data field does not supply a value or contains all blanks.
- The application data field supplies a value that does not match a value in the validation or translation table specified in the mapping.
- The application data field supplies a value, but the conversion from the application data type to the EDI standard data type fails.

DataInterchange attempts a conversion for any application data field defined as a numeric field. The conversion removes leading and trailing blanks, leading zeros before the decimal, trailing zeros after the decimal, and changes the decimal point if the application decimal notation is different from the EDI standard decimal notation. If the data types are numeric, the conversion fails if the data contained anything other than a number or a decimal point. The literal value is used if the conversion fails. See “Validation during mapping” on page 396 for more information.

- If a translation table is specified, the literal value is checked against this table. If no matching entry is found, the translator logs a warning message in the event log, then uses the literal value. For information about the event log, see Chapter 27, “Event Logs.”

Segment creation for send mapping

When an EDI standard segment is mapped with a combination of literal values and application data, the application data determines if the segment is produced. When the values from the application data produce data for the segment, the EDI standard segment is created. When the values in the application data do not produce data for the segment, such as when the records are not found, the application fields contain all blanks or zeros, or the application field values are not found in the associated validation or translation tables, then the EDI standard segment is not created. An exception is when zeros are passed to a mandatory numeric data element, in which case the segment is created. If this is not desirable, then pass blanks in the application data instead, or use the logic discussed below to suppress the segment.

In some cases it may be desirable to suppress a segment altogether depending on input application data values. It is possible to suppress a segment using *IF* logic. When all contributing data element mappings for a segment contain conditional *IF* statements and all conditions are false, the segment is not produced. A contributing data element mapping is one that may produce output in the corresponding data element; such as specifying an application field name or a literal

value, or specifying *USE* on a variable. Noncontributing data element mappings, such as *SAVE* or *SET*, do not have any effect on segment creation. Refer to Table 86 on page 360 for details on *IF*, *USE*, *SAVE*, and *SET* usage.



NOTE: If the segment being suppressed begins a loop of segments, the entire loop is suppressed.

Using literals for receive mapping

If a data element mapping provides a literal, the literal value is used if one of the following conditions occur:

- The EDI standard data includes the segment being mapped, but the data element within the segment does not supply a value.
- The EDI standard data element supplies a value that does not match a value in the validation or translation table specified in the mapping.
- The EDI standard data element supplies a value, but the conversion from the EDI standard data type to the application data type fails. DataInterchange attempts a conversion for any EDI standard data field defined as a numeric field. If the data types are numeric and the EDI standard data field contains nonnumeric characters, the conversion would fail and the literal value would be used.
- The EDI standard data element supplies a value, but the &FORCE special literal is used to force the literal into the application field regardless of the EDI standard data element's contents.

Format of literal data

When using literals in send or receive mapping:

- For data types BN, Bn, HX, Hn, IT, In, Ln, PD, Pn, ZD, and Zn, the translator converts the literal before placing it in the standard data (send) or the application data (receive). For sending, it converts the literal to character data. For receiving, it converts the literal to the application data type. For example, if the data type is binary, the translator converts the literal to binary, then moves it to the application field.
- Do not type a decimal point when it is implied. For example, if you want to use a default value of 9.99 for a field defined as data type P2 (packed number with two implied decimal positions), enter 999 as the value of the literal.
- Enter literal values for hexadecimal fields as hexadecimal strings. For example, if the application field is defined as a one-byte hexadecimal field and you want to use a default value of X'FF', enter FF as the value of the literal. For receiving, the translator converts each two bytes of literal value to a single byte of application data.

- You can specify a literal value of zero to move a value into the standard field. DataInterchange generally removes leading zeros from an application field so that an application field containing nothing but blanks or zeros will not result in a value for the data element. A value of zero is treated the same as all other literal values when determining if a segment should be created. The &ZEROSIG special literal may also be used to indicate that zeros within the application field are significant.

In translation and validation tables, enter numeric values left-justified and formatted according to the application data format. For example, if the data format defines a field as R2, enter the value 7 as 7.00 or the value 7.1 as 7.10.

Accumulator literals

The following literals allow access to the local and global accumulators. You can map an accumulator using the accumulator action, or you can map it using the literal associated with the accumulator. Using the literal allows a user exit to gain control. An accumulator has a data type of R. The following list describes the literals:

Literal:	Description:
&T n	Where n can be 0 through 9, identifies the accumulators T0 through T9.
&G n	Where n can be 0 through 9, identifies the accumulators G0 through G9.

Conditional processing literals

Conditional processing lets you define how you want data processed, based on rules you establish. The following terms will be used in discussing conditional processing:

Term	Description
Named variable	A name used to represent data whose value can be changed while a program is running. You supply the name you want to use.
Expression	A sequence of instructions which can consist of named variables, literals, operators, and constants. When processed, this sequence of instructions provides a single value.
Constant	A value that does not change.
Operator	A symbol that represents an operation to be done. DataInterchange uses arithmetic operators, Boolean operators, comparison operators, unary operators, relational operators, and special operators.
Operation	An action performed on one or more data items such as multiplying, comparing or moving.
Value	A data value you provide. You can use any of the special literals that provide a data value; for example, &DATE, &TIME, &E, &ICN. This includes &T0 through &T9 and &G0 through &G9 to get the values for global and local accumulators.

Term	Description
Default value	A default data value you provide. This value should not be enclosed in quotation (") marks. You can use any of the special literals that provide a data value; for example, &DATE, &TIME, &E, &ICN. This includes &T0 through &T9 and &G0 through &G9 to get the values for global and local accumulators.



NOTE: When using conditional processing on a data element used to qualify a loop or repeating segment, the first occurrence of the qualifying data element mapping must specify a literal or application data field.

Literal keywords

Table 86. Literal Keywords

Keyword	S/R	Keyword Description and Syntax
&ACFIELD	S/R	<p>Syntax: &ACFIELD</p> <p>Substitutes the application control value established by the mappings of the AC field from the data format or the concatenation of the fields specified during creation of the transaction mapping.</p> <p>Note: Literals specified with &LIT and variables specified with &VAR in the mapping are not available in the AC field until the end of the translation process and will not appear in the AC field data that is moved using the &ACFIELD keyword.</p>

Keyword	S/R	Keyword Description and Syntax
&ASSERT n	S/R	<p>Syntax: &ASSERTn(<i>expression</i>)</p> <p>The action associated with this literal is only executed if the <i>expression</i> is false and the assertion level is not greater than n. The &ASSERT keyword can be combined with:</p> <ul style="list-style-type: none"> A mapping between an application data field and EDI standard data elements The &SET, &SAVE, and &USE keywords An error condition; for example, &ASSERTn followed by &ERR. <p>The differences between &ASSERT and &IF are:</p> <ul style="list-style-type: none"> &ASSERT is a statement about the transaction that is expected to be true. For example, the number of items processed in the transaction should equal the total number of items claimed to be in the transaction (value from CTT segment). If the &ASSERT is not true, special action should take place. An &ASSERT is usually associated with an &ERR condition, but this is not required. With &IF, if the expression is true, special action should take place. An &IF is usually associated with a mapping or named variable, but this is not required. <p>&ASSERT has 10 levels (&ASSERT0 through &ASSERT9). Level 9 assertions are always executed. Assertions at level 0 (&ASSERT0) through 8 are controlled by the assertion level used to start the translation. Thus, it is possible to turn off assertions, but &IF conditions are always checked.</p> <p>When using the API, the assertion level is set in the ASSERTLVL field of the translator control block. When using the DataInterchange Utility, the assertion level is set with the ASSERTLVL keyword on the PERFORM command. See the <i>DataInterchange Programmer's Reference</i> for additional information on the API.</p> <p>See “Expressions” on page 372 for more information.</p> <p>For an example of using this keyword, see “Example 8:” on page 388.</p>
&DATE	S/R	<p>Syntax: &DATE</p> <p>Substitutes the system date. The length of the date field in the EDI standard data (send) or application data (receive) determines whether the date is formatted as <i>yyyymmdd</i> or <i>yymmdd</i>. The date is then edited as requested by the date edit specified in mapping.</p> <p>You can use the &DATE keyword as source data for any of the EDI standard data types.</p> <p>You can also combine the &DATE keyword with the &IFDATA, &IFNODATA, or &FORCE keywords.</p>

Keyword	S/R	Keyword Description and Syntax
&DEFERRED	S	<p>Syntax: &DEFERRED &USE <i>variable</i></p> <p>Allows you to signal that at a later time a value will be set using the same name as specified in the &DEFERRED &USE mapping. When the value is set, the data is put into the mapped segment.</p> <p>For example, if you had an HDR segment that contained a total number of items field, you could use &DEFERRED &USE TOTITEMS while mapping the HDR segment field.</p> <p>NOTE: It is not recommended that the value be set within the same segment mapping. This will produce unexpected results. If the entire segment is mapped based on the use of deferred mapping, the segment could be produced without data.</p> <p>At the end of the mapping, you would then repeat map some field with &SET TOTITEMS X, which will move the literal value 'X' into the HDR segment field. If X is used as a variable, the mapping would be &SET TOTITEMS &E(X), which would move the contents of the variable X into the HDR segment field.</p>
&E	S/R	<p>Syntax: &E(<i>expression</i>)</p> <p>The <i>expression</i> is evaluated and the result used as if it had been entered directly as a literal value. This keyword allows calculations without using a user exit. See “Expressions” on page 372 for more information.</p> <p>For examples of using this keyword, see “Example 7:” on page 388.</p>

Keyword	S/R	Keyword Description and Syntax
&ERR	S/R	<p>Syntax: &ERR(<i>level,code,facode,text</i>)</p> <p>Allows you to establish your own errors for a transaction. This literal keyword can be used in one of three ways:</p> <ul style="list-style-type: none"> • On the Literal line itself • &IF (expression) • &ASSERT (expression) <p>NOTE: If you specify a field in the application name field, you cannot use &ERR on this occurrence of the element mapping.</p> <p><i>level</i> is the severity of error where 1=data element, 2=segment, 3=transaction.</p> <p><i>code</i> is the unique error code that should be associated with the error. This value can range from 0 to 999. DataInterchange automatically adds 5000 to separate this value from DataInterchange detected errors.</p> <p><i>facode</i> is the functional acknowledgment error code that should be associated with this error (receive only).</p> <p><i>text</i> is some text that is included in an error message logged by DataInterchange if this error is detected.</p> <p>If an &ERR special literal is executed (normally controlled by either an &IF or an &ASSERTion), then DataInterchange will log message TR0026. Within this message, the <i>text</i> and <i>code</i> will be identified. The <i>level</i> that you assign in the &ERR becomes the extended return code from the translator and thus the JCL condition code in the DataInterchange Utility. If <i>level</i> exceeds the acceptable error level specified in the transaction usage, then the translation will not be successful and the application data will not be returned. The value of <i>code</i> plus 5000 will also be added to the list of errors for the transaction. These are available to the API programs in the ERRCODE field of the translator control block.</p> <p>For an example of using this keyword, see “Example 8:” on page 388.</p>
&FORCE	R	<p>Syntax: &FORCE <i>value</i></p> <p>Forces a literal value into an application field regardless of the EDI standard data element's contents. A literal value specified for receive mapping is normally used only if the EDI standard data element does not contain any data, or if an error occurs while processing the data (see “Using literals for receive mapping” on page 358).</p> <p>For the &FORCE keyword to be effective, you must use it in a data element of a segment that is present in the input transaction.</p>
&FORMAT	S/R	<p>Syntax: &FORMAT</p> <p>Substitutes the data format ID.</p> <p>You can also use the &FORMAT keyword with the &IFDATA, &IFNODATA, or &FORCE keywords.</p>

Keyword	S/R	Keyword Description and Syntax
&IF	S/R	<p>Syntax: &IF(<i>expression</i>)</p> <p>The action associated with this literal is only executed if the <i>expression</i> is true. The &IF literal can be combined with:</p> <ul style="list-style-type: none"> • A mapping between application fields and EDI standard data elements • A request to SET/SAVE/USE a named variable • An error condition (for example, an &IF followed by &ERR) <p>The differences between &ASSERT and &IF are:</p> <ul style="list-style-type: none"> • &ASSERT is a statement about the transaction that is expected to be true. If the &ASSERT is not true, special action should take place. <p>With &IF, if the expression is true, special action should take place. An &IF is usually associated with a mapping or named variable, but this is not required.</p> <ul style="list-style-type: none"> • &IF conditions are always checked. <p>See “Expressions” on page 372 for more information. For an example of using this keyword, see “Example 5a:” on page 386, “Example 5b:” on page 386, and “Example 5a:” on page 386.</p>
&IFDATA	S/R	<p>Syntax: &IFDATA <i>value</i></p> <p>For sending, uses a literal value only if an application field contains data. For example, a segment has a pair of data elements for a qualified value and its qualifier. The application data has a corresponding field for the qualified value, but not for the qualifier. You want to supply the qualifier with a literal, but only when the application supplies a qualified value. You can do this using the &IFDATA keyword with the Application field name and Literal fields on the Map Data Element panel (TP10). &IFDATA can be used in combination with &SET or &SAVE keywords.</p> <p>The test performed by the translator to determine whether or not a source field contains data is different depending on the data type of the source field. For numeric data types, zeros are not considered significant, in which case the assumption is made that the field does not contain data. &IFDATA can be used in combination with &ZEROSIG if you want a zero value to be considered significant. &ZEROSIG must precede the save keyword.</p> <p>&IFDATA is allowed on receive but only when used in combination with the &SET or &SAVE keywords.</p> <p>For an example of using this keyword, see “Example 2:” on page 384.</p>
&IFNODATA	S/R	<p>Syntax: &IFNODATA <i>value</i></p> <p>Uses a literal value only if an application field does not contain data. If you do not use a keyword before the literal, &IFNODATA is used by default.</p> <p>&IFNODATA can be used in receive transactions, but only when used in combination with the &SET keyword.</p>

Keyword	S/R	Keyword Description and Syntax
&IFNOVAR	S/R	<p>Syntax: &IFNOVAR</p> <p>This keyword is only valid when used with &SAVE, &LSAVE, &SET, or &LSET:</p> <p>&IFNOVAR &SAVE <i>variable</i> &IFNOVAR &LSAVE <i>variable</i> &IFNOVAR &SET <i>variable</i> &IFNOVAR &LSET <i>variable</i></p> <p>The named variable will only be created if it does not already exist. If the named variable already exists, the data in it is not overlaid.</p> <p>For an example of using this keyword, see “Example 3:” on page 384.</p>
&LOOPBREAK	S	<p>Syntax: &LOOPBREAK</p> <p>Use when an outer and an inner loop are qualified on the same record, and when you want the inner loop to be generated more than once. By putting a &LOOPBREAK in the inner loop, then the inner loop will be repeated until the &LOOPBREAK condition is met. The &LOOPBREAK condition is established using &IF; for example,</p> <p>&IF((A < B) AND (X > Y)) &LOOPBREAK</p> <p>NOTE: You cannot enter an application field name on the same mapping occurrence that uses &LOOPBREAK.</p>
&LOOPCHECK	S	<p>Syntax: &LOOPCHECK</p> <p>This keyword is very similar to &LOOPBREAK, but &LOOPCHECK does all the conditional processing automatically, based on the application field name used in the mapping that specifies &LOOPCHECK. DataInterchange will save this field value the first time the loop is created, and will continue to create inner loops until a record with a different non-blank field value is found.</p> <p>NOTE: You cannot enter an application field name on the same mapping occurrence that uses &LOOPCHECK.</p>

Keyword	S/R	Keyword Description and Syntax
&LSAVE	S/R	<p>Syntax: &LSAVE <i>variable</i><,<<i>position</i>><,<<i>length</i>> <<i>default value</i>></p> <p>Saves the value from the current data element in a named variable, but only for the duration of the loop instance. A value saved in an outer loop is available within associated inner loops. A repeating segment is considered to be a loop; therefore, values saved in one instance of a repeating segment are not available in subsequent segment iterations.</p> <p>The value to be saved is normalized prior to being stored in the variable. The normalization that takes place depends on the data type of the source field being saved. For character data types A, AC, AN, CH, or ID, any trailing blanks are removed from the source value. If the value in the field (defined with a character data type) contains all numbers, the data will be treated as a numeric data type. Leading zeros will be removed from numeric data types. For numeric data types, the source value has all leading and trailing blanks removed, it is converted to a REAL value, and all leading zeros before the decimal and trailing zeros after the decimal are removed. If a source field is numeric and contains a zero value, the variable will contain null. If a value of zero needs to be saved, &ZEROSIG must be used in combination with the save. &ZEROSIG must precede the &LSAVE keyword.</p> <p><i>position</i> is optional, but if provided, it indicates the position within the current variable where the information should be saved. You can use an asterisk (*) to save at the end of the variable; for example, &LSAVE <i>name</i>,*. If <i>position</i> is not specified, the data replaces the current value of the variable.</p> <p><i>length</i> is optional, but if provided, it indicates the length of data that should be saved. You can use an asterisk to save the total length of the data being supplied; for example, &LSAVE <i>name</i>,*,*.</p> <p>For example, if you want data to be saved starting in position 4 and the data you want saved is 5 characters long, the position and length would be stated as 4,5. Stating both the position and length, you can combine more than one data element into a single named variable. Special processing may be needed when combining data elements with different data types into a single named variable. See “Example 9:” on page 390 and “Example 10:” on page 391.</p> <p>The <i>default value</i> is also optional, and if provided, the value is saved in the <i>variable</i> when the current data element contains no value.</p> <p>If the named variable does not already exist, it is created. If it already exists, the data in the existing variable is overlaid with the new data. If the data element is empty and no default value is supplied, the named variable is created but it contains no data.</p> <p>A variable established with &LSAVE does not affect the value of a variable with the same name at any other looping level. If the variable was created outside the loop with LSAVE or SAVE, and the variable is created again inside the loop, the USE for the variable inside the loop will use the value which was saved inside the loop. If the USE for the variable is outside the loop, then the value which was saved outside the loop will be used.</p> <p>For an example of using this keyword, see “Example 3:” on page 384.</p>

Keyword	S/R	Keyword Description and Syntax
&LSET	S/R	<p>Syntax: &LSET <i>variable</i><,<<i>position</i>><,<<i>length</i>> <i>value</i> The named variable is created or overlaid with the stated value, but only for the duration of the loop instance. A value set in an outer loop is available within associated inner loops. A repeating segment is considered to be a loop; therefore, values set in one instance of a repeating segment are not available in subsequent segment iterations.</p> <p><i>position</i> is optional, but if provided, it indicates the position within the current variable where <i>value</i> should be set. You can use an asterisk to set <i>value</i> at the end of the variable, for example, &LSET name,* <i>abcd</i>. If <i>position</i> is not specified, <i>value</i> replaces the current value of the variable.</p> <p><i>length</i> is optional, but if provided, it indicates the length of <i>value</i> that should be set. You can use an asterisk to set the total length of <i>value</i>, for example, &LSET name,*,* <i>abcd</i>. Special processing may be needed when combining data elements with different data types into a single named variable. See “Example 9:” on page 390 and “Example 10:” on page 391.</p> <p>A variable established with &LSET does not affect the value of a variable with the same name at any other looping level. If the variable was created outside the loop with LSET or SET and the variable is created again inside the loop, the USE for the variable inside the loop will use the value which was set inside the loop. If the USE for the variable is outside the loop, then the value which was set outside the loop will be used.</p> <p>When zeros are passed in a DT application field and mapped to a DT element, they are considered significant data, and are populated to the element. In other words, DataInterchange treats DT data as character versus numeric.</p>
&LSID	R	<p>Syntax: &LSID <i>value</i></p> <p>Identifies the instance of the LS loop, where <i>value</i> is equal to the LS01 value in the EDI standard data. This special literal is required only if the translator has no other way to determine which LS loop is provided.</p>
&SAMEAS	S/R	<p>Syntax: &SAMEAS <i>seqno</i></p> <p>Indicates when a mapping for a current data element should be exactly the same as the data element identified by <seqno>.</p> <p>For example, if the mapping for element 2 of the POC segment should be exactly the same as the mapping for element 1 of the POC segment, then specify the special literal value of '&SAMEAS 1' when mapping element 2. This results in the mapping for element 2 being the same as element 1, which is useful when qualifying (Q/S) data elements on a receive mapping.</p>

Keyword	S/R	Keyword Description and Syntax
&SAVE	S/R	<p>Syntax: &SAVE <i>variable</i><,<i>position</i>><,<i>length</i>> <<i>default value</i>></p> <p>Saves the value from the current data element (inbound) or application field (outbound) in a named variable. The value to be saved is normalized prior to being stored in the variable. The normalization that takes place depends on the data type of the source field being saved. For character data types A, AC, AN, CH, or ID, any trailing blanks are removed from the source value. For numeric data types, the source value has all leading and trailing blanks removed, it is converted to a REAL value, and all leading zeros before the decimal and trailing zeros after the decimal are removed. If a source field is numeric and contains a zero value, the variable will contain null. If a value of zero needs to be saved, &ZEROSIG must be used in combination with the save. &ZEROSIG must precede the &SAVE keyword.</p> <p><i>position</i> is optional, but if provided, it indicates the position within the current variable where the information should be saved. You can use an asterisk to save at the end of the variable; for example, &SAVE name,*. If <i>position</i> is not specified, the data replaces the current value of the variable.</p> <p><i>length</i> is optional, but if provided, it indicates the length of data that should be saved. You can use an asterisk to save the total length of the data being supplied, for example, &SAVE name*,*.</p> <p>For example, if you want data to be saved starting in position 4 and the data you want saved is 5 characters long, the position and length would be stated as 4,5. Stating both the position and length, you can combine more than one data element into a single named variable. Special processing may be needed when combining data elements with different data types into a single named variable. See “Example 9:” on page 390 and “Example 10:” on page 391.</p> <p>The <i>default value</i> is also optional, and if provided, the value is saved in the <i>variable</i> when the current data element contains no value.</p> <p>If the named variable does not already exist, it is created. If it already exists, the data in the existing variable is overlaid with the new data. If the data element is empty and no default value is supplied, the named variable is created but it contains no data.</p> <p>For examples of using this keyword, see “Example 1:” on page 384.</p> <p>When zeros are passed in a DT application field and mapped to a DT element, they are considered significant data, and are populated to the element. In other words, DataInterchange treats DT data as character versus numeric.</p>

Keyword	S/R	Keyword Description and Syntax
&SET	S/R	<p>Syntax: &SET variable< ,position>< ,length> value</p> <p>The named variable is created or overlaid with the stated value. If no default value is specified, &SET clears the variable.</p> <p><i>position</i> is optional, but if provided, it indicates the position within the current variable where value should be set. You can use an asterisk to set value at the end of the variable, for example, &SET name,* abcd. If no default value is specified, &SET clears the variable.</p> <p><i>length</i> is optional, but if provided, it indicates the length of value that should be set. You can use an asterisk to set the total length of value; for example, &SET name,*,* abcd. Special processing may be needed when combining data elements with different data types into a single named variable. See “Example 9:” on page 390 and “Example 10:” on page 391.</p> <p>When zeros are passed in a DT application field and mapped to a DT element, they are considered significant data, and are populated to the element. In other words, DataInterchange treats DT data as character versus numeric.</p>
&THANDLE	R	<p>Syntax: &THANDLE</p> <p>Substitutes the DataInterchange archive key. Can be used to assist in mapping the SAP IDOC. It enables mapping of the DataInterchange archive key to the SAP IDOC for inbound processing. The length of the THANDLE field is 20 characters and is formatted as YYYYMMDDHHMMSSnnnnnn. It is the concatenation of the date, time, and a sequence number to ensure uniqueness.</p>
&TIME	S/R	<p>Syntax: &TIME</p> <p>Substitutes the system time. The length of the time field in the EDI standard data (send) or application data (receive) determines whether the time is formatted as <i>hhmm</i> or <i>hhmmss</i>.</p> <p>You can use the &TIME keyword as source data for any of the EDI standard data types.</p> <p>You can also combine the &TIME keyword with the &IFDATA, &IFNODATA, or &FORCE keywords.</p>
&TPID	S/R	<p>Syntax: &TPID</p> <p>Substitutes the value of the internal trading partner ID.</p> <p>You can also use &TPID keyword with the &IFDATA, &IFNODATA, or &FORCE keywords.</p>
&TPNICKN	S/R	<p>Syntax: &TPNICKN</p> <p>Substitutes the value of the trading partner nickname.</p> <p>You can also use &TPID keyword with the &IFDATA, &IFNODATA, or &FORCE keywords.</p>

Keyword	S/R	Keyword Description and Syntax
&USE	S/R	<p>Syntax: &USE <i>variable</i><,<<i>position</i>>><,<<i>length</i>>> <<i>default value</i>></p> <p>For inbound transactions, the value of the named variable is the source of data for the application field. The value of the EDI standard data element being mapped is ignored.</p> <p>For outbound transactions, the value of the named variable is used to provide data for the EDI standard data element being mapped. An application field cannot be specified.</p> <p>The named variable must be saved or set using the appropriate keyword before you can use it with this keyword.</p> <p><i>position</i> is optional, but if provided, it indicates the position within the variable from which the data should be retrieved.</p> <p><i>length</i> is optional, but if provided, it indicates the length of data that should be retrieved. You can use an asterisk to move all data beginning at the location specified by the position parameter through the end of the variable, for example, &USE var 3,*.</p> <p><i>default value</i> is optional, and if provided, the default value is used when the variable contains no value.</p> <p>NOTE: For numeric elements, a variable value of zero causes DataInterchange to use the default value. In the case where no default is specified, as in literal = &USE X, there will be no output if variable X contains zero. At times, however, zero needs to be considered significant. In these cases, the user should specify the default, as in literal = &USE X 0.</p> <p>For an example of using this keyword, see “Example 1:” on page 384.</p>
&ZEROSIG	S	<p>Syntax: &ZEROSIG <<i>default value</i>></p> <p>Use the keyword &ZEROSIG to indicate that a zero in an application field is significant when mapping to optional or conditional data elements. Without use of this special literal, DataInterchange considers a zero value being mapped to an optional or conditional data element as insignificant and produces no output during translation. You can combine the use of &ZEROSIG with &IFDATA, &IFNODATA, &SAVE, and &LSAVE keywords. &ZEROSIG must precede the other keyword. &ZEROSIG can also be used with a default literal to indicate that the application field is used if it contains a value, including zero, but the default literal is used if the field is blank. Binary data types preclude this because blanks represent real values.</p>

Named variables

For receiving transactions, one way to use data in one data element for multiple application fields is to map the data element to each application field, using the repeat mapping capability. The disadvantage is that the application structures are always created, whether or not they are needed. Sometimes, you do not want the structure to be created unless some other data within the transaction is present. *Named variables* let you save the value of a data element until the time when the application structure should be created. Then when you are mapping the data element that creates the structure, you can use the repeat mapping capability to map the value in the named variable.

Named variables are also critical to the use of expressions and conditional processing. See “Expressions” on page 372 and “Conditional processing literals” on page 359 for additional information. Data values from an application field for outbound processing or standard data elements for inbound processing are not directly available for use in an expression. The values must first be saved to a named variable using for example, the &SAVE literal, then the named variable can be used within the expression.

A variable name can be the same as your application field name, up to 16 characters, but it cannot start with any of the following:

- A numeric digit (0 through 9).
- The letter P. These variables are reserved for future use.
- The letters DI. These variables are reserved for DataInterchange.
- An ampersand (&), so they do not get confused with special literals.
- A left parenthesis, so they do not get confused with the start of an expression.

A variable name cannot contain any of the special characters designated as Arithmetic Operators or Alternate Comparison Operators. Use of these special characters within a variable name will produce unpredictable results.

The first character of a variable name determines the life span or scope of the variable:

- If the first letter is anything other than G, the variable has transaction scope. The variable is deleted after the transaction is translated. This is the same scope as local accumulators (T0 through T9). For more information, see “Using accumulators” on page 355.
- If the first letter is G, the variable has translator session scope. The variable is not deleted until the session with the translator is terminated. This is the same scope as global accumulators (G0 through G9). For more information on accumulators, see “Using accumulators” on page 355.
- To create a variable that will exist only for the duration of the loop in which it was created, use the &LSAVE or &LSET literal keywords. When the loop repeats or terminates, any variable created using these keywords is deleted. A variable established with &LSAVE or &LSET does not disturb the value of a variable with the same name at any other looping level.

Variable names are not case sensitive. *TOTALITEMS* and *totalitems* are the same variable. Special processing may be needed when combining data elements with different data types into a single named variable. See “Example 9:” on page 390 and “Example 10:” on page 391.

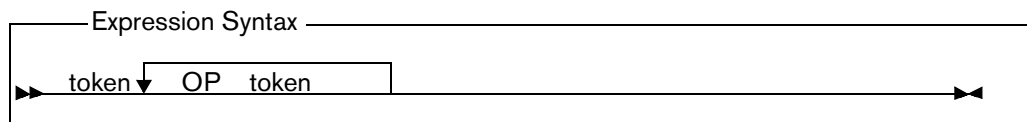
You can combine the substring capability in mapping with the substring capability of named variables. For example, you can use sub strings to get the first 4 bytes of a standard data element, then use the *position* or *length* options to put the data in a specific place in the named variable.

Expressions

Special literals that use expressions are:

- &IF
- &ASSERT
- &E

The syntax for an expression is:



Where *token* can be either:

- A named variable, such as TOTALS
- A numeric constant, such as 1024
- A text constant, such as 'ABCD'



NOTE: You must use quotation marks around text constants to distinguish them from named variables. You can use single or double quotes (' or "), but whichever is used to start the text constant must also be used to end it.

and where *OP* is one of the following types of operators:

- Boolean
- Comparison
- Arithmetic
- Unary
- Special

By default, the data type of a variable is implicitly assigned. If the contents are all numeric digits, the data type is assumed to be numeric and is treated as such in subsequent comparisons. This assignment does not take into consideration the data type of the source data field that originally supplied the value for the variable. It is based simply on the contents of the variable. If a variable contains any nonnumeric characters, the variable data type is assumed to be character. You can override this implicit data type assignment with the use of the CHAR or NUMBER operator. For more information, see the operators CHAR and NUMBER below.

Term	Description
CHAR	The CHARACTER operator forces DataInterchange to treat a value as a character value rather than a numeric value.
NUMBER	The NUMBER operator forces DataInterchange to treat a value as a numeric value rather than a character value. DataInterchange will normally treat a value in quotes as a character value and, if both operands look like character values then what might be thought of as a numeric operator will be treated as a string operator. Thus, the expression &E('12' + '34') will yield '1234' because both operators are flagged as character data. If what you really wanted was an arithmetic addition rather than a string concatenation then you could use &E(NUMBER('12') + NUMBER('34')) which will yield 46 (as would &E(12 + 34)).

Boolean operators

DataInterchange has two Boolean operators: AND and OR. AND returns a value of 1 if both conditions are true, or a value of 0 if either condition is not true. OR returns a value of 1 if either condition is true, or a value of 0 if both conditions are not true. AND and OR must be entered in uppercase because they are case sensitive.

Comparison operators

Comparison operators tell DataInterchange to compare two objects.

DataInterchange processes comparison operators the same way it does Boolean operators. If the comparison is true, it returns a value of 1. If it is not true, it returns a value of 0.

DataInterchange has the following comparison operators:

Operator:	Alternate:	Description:
EQ	=	The first value is the same as the second value.
GE	>=	The first value is greater than or equal to that of the second value.
GT	>	The first value is greater than the second value.
LE	<=	The first value is less than or equal to that of the second.
LT	<	The first value is less than the second value.
NE	!=	The values are not equal, because they are either different data types or different values.



NOTE:

- If you use the alphabetic operators, such as EQ, they must be uppercase.
- Comparing a numeric constant or variable to a character constant or variable results in an invalid comparison. This produces a false result for all comparisons with the exception of Not Equal (NE or !=), in which case a true result is returned.
- If you enclose a numeric constant in quotation marks ("), a numeric data type is still assumed for the data. Therefore, comparison to a variable that contains numeric data will not fail.

Arithmetic operators

In the following descriptions, single quote marks (') surround absolute values. DataInterchange has the following arithmetic operators:

Operator:	Description:
+	$a + b$ has a value equal to the sum of the two. For numeric values, the sum is numeric. For example, $2 + 2 = 4$. For nonnumeric values, the sum is a concatenation of the values. For example, 'AB' + 'CD' equals 'ABCD'. The use of quotation marks around a numeric constant has special meaning when used during addition. The enclosed value will be treated as character data and concatenated to the second value. For example, '19' + 940323 = 19940323.

Operator:	Description:
-	$a - b$ has a value equal to the difference between the two. For numeric values, the result is numeric. For example $4 - 2 = 2$. For nonnumeric values, the result is a deconcatenation of the two values. For example, 'ABCD' - 'CD' equals 'AB'. The use of quotation marks around a numeric constant has special meaning when used during subtraction. The enclosed value will be treated as character data and deconcatenated from the second value. For example, $19940323 - '23' = 199403$.
*	$a * b$ has a value equal to the multiplication of a and b , if both a and b are numeric. For example, $4 * 4 = 16$. If either variable is nonnumeric, the result is created by concatenating the value of the b after each character in a . For example, 'ABCD'*'Z' equals 'AZBZCZDZ'.
/	a / b has a value equal to the division of a by b , if both a and b are numeric. For example, $16 / 4 = 4$. Division by 0 yields a 0 value. If either variable is nonnumeric, the result is created by removing each occurrence of b from a . For example, 'ACDBCD'/'CD' equals 'AB'.
CHAR	The CHARacter operator forces DataInterchange to treat a value as a character value rather than a numeric value.
NOT	An operator to reverse the Boolean value of an expression. The exclamation point (!) is a short hand for the NOT operator. NOT(value) has the following meanings: <ol style="list-style-type: none"> 1. If value is numeric, then NOT(value) is 1 if the value is 0 and 1 otherwise. Thus, NOT(0) yields 1 and NOT(1) yields 0. 2. If value is a string value, then NOT(value) is 1 if the string has no length and zero otherwise. Thus, NOT('abc') yields 0 and NOT('') yields 1.
NUMBER	The NUMBER operator forces DataInterchange to treat a value as a numeric value rather than a character value. DataInterchange will normally treat a value in quotes as a character value and, if both operands look like character values then what might be thought of as a numeric operator will be treated as a string operator. Thus, the expression &E('12' + '34') will yield '1234' because both operators are flagged as character data. If what you really wanted was an arithmetic addition rather than a string concatenation then you could use &E(NUMBER('12') + NUMBER('34')) which will yield 46 (as would &E(12 + 34)).
RD or:	$a \text{ RD } n$ has a value equal to a rounded to n decimal places if a is numeric. For example, $4321.556 \text{ RD } 2$ equals 4321.56. If a is nonnumeric, the result is created by taking the first n characters from a . For example, 'ACDBCD' RD 3 equals 'ACD'. If n is less than or equal to zero it is interpreted to be a request to remove leading blanks. Thus, ' ABC' RD 0, yields 'ABC'. If using the character string RD for rounding instead of the special character:, it must be entered using uppercase.

Operator:	Description:
TU or;	a TU n has a value equal to a truncated to n decimal places if a is numeric. For example, 4321.556 TU 2 is 4321.55. Notice that the number 6 is dropped from the result and not rounded. If a is nonnumeric, the result is created by taking the last n characters from a . For example, 'ACDBCD' TU 3 yields 'BCD'. If n is less than or equal to zero it is interpreted to be a request to remove trailing blanks. Thus, 'ABC ' TU 0 yields 'ABC'. If using the character string TU instead of the special character;, it must be entered using uppercase.

Unary operator

The following describes the unary (single component) operator:

Operator:	Description:
-	&E(- a) changes the sign of a . If a does not exist, the value is 0.

Special operators

The following list describes the special operators:

Operator:	Description:
UE	<p>&E(a UE 'MYPROG') returns a value from the user-written program MYPROG.</p> <p>NOTE: As mentioned under the descriptions of "&LSAVE" on page 366 and "&LSET" on page 367, variables are normalized to REAL value equivalents. When using this special operator, the variable value that is passed is converted back to its Nn numeric format.</p>
TS	<p>&E(a TS 'MYTABL') translates the local value a to the standard value using the MYTABL translation table.</p> <p>NOTE:</p> <ol style="list-style-type: none"> As discussed under the descriptions of "&LSAVE" on page 366 and "&LSET" on page 367, variables are normalized to REAL value equivalents. When using this special operator, the variable value that is passed is converted back to its Nn numeric format. If the translation table specified does not exist, or the value passed does not match an entry in the table, then no data is produced from this instruction and no errors or exceptions are issued.

Operator:	Description:
TL	<p>&E(<i>a</i> TL 'MYTABL') translates the standard value <i>a</i> to the local value using the MYTABL translation table.</p> <p>NOTE:</p> <ol style="list-style-type: none"> As discussed under the descriptions of “&LSAVE” on page 366 and “&LSET” on page 367, variables are normalized to REAL value equivalents. When using this special operator, the variable value that is passed is converted back to its <i>Nn</i> numeric format. If the translation table specified does not exist, or the value passed does not match an entry in the table, then no data is produced from this instruction and no errors or exceptions are issued.
IN	<p>&E(<i>a</i> IN 'MYTABL') returns a value equal to 1 if <i>a</i> exists in the MYTABL validation table.</p> <p>NOTE:</p> <ol style="list-style-type: none"> As discussed under the descriptions of “&LSAVE” on page 366 and “&LSET” on page 367, variables are normalized to REAL value equivalents. When using this special operator, the variable value that is passed is converted back to its <i>Nn</i> numeric format. If the validation table specified does not exist, or the value passed does not match an entry in the table, then no data is produced from this instruction and no errors or exceptions are issued.
SC	<p>&E(<i>a</i> SC <i>max.dec</i>) scales a real number to a maximum of <i>max</i> digits with a maximum of <i>dec</i> decimal places, truncating unused digits. For example:</p> <p>&E(1234.56 SC 4.2) yields 1234 &E(1234.56 SC 6.2) yields 1234.56 &E(1234.56 SC 6.1) yields 1234.5 &E(1234.56 SC 8.1) yields 1234.5 &E(1234.56 SC 8.8) yields 1234.56 &E(1234.56 SC 3.3) yields 1234.56 &E(1234.56 SC 4.5) yields 1234</p> <ul style="list-style-type: none"> If <i>max</i> is less than the number of significant digits to the left of the decimal point, the SC is ignored. If <i>dec</i> is greater than <i>max</i>, <i>dec</i> is set equal to <i>max</i>. <p>SC applied to strings provide a substring capability. When applied to strings, the format is &E('STRING' SC <i>pos.len</i>). For example:</p> <p>&E('ABCDEF' SC 4.2) yields 'DE' &E('ABCDEF' SC 1.5) yields 'ABCDE' &E('ABCDEF' SC 9.1) yields '' &E('ABCDEF' SC 1.9) yields 'ABCDEF' &E('ABCDEF' SC .1) yields 'A' &E('ABCDEF' SC 5) yields 'E'</p> <p>If you do not provide <i>pos</i> or <i>len</i>, DataInterchange uses 1 for that value.</p>

Operator:	Description:
SR	<p>&E(<i>a</i> SR <i>max.dec</i>) scales a real number to a maximum of <i>max</i> digits with a maximum of <i>dec</i> decimal places, rounding the value. For example:</p> <p>&E(1234.56 SR 4.2) yields 1235 &E(1234.56 SR 6.2) yields 1234.56 &E(1234.56 SR 6.1) yields 1234.6 &E(1234.56 SR 8.1) yields 1234.6 &E(1234.54 SR 8.1) yields 1234.5</p>
IS	<p>&E(<i>a</i> IS <i>pattern</i>) has a value equal to <i>a</i> but establishes a <i>pattern</i> for the data within <i>a</i>. This <i>pattern</i> only has meaning when using the TO operator (next). All variables have a default pattern of "ABCDEFGHIJKLMNOPQRSTUVWXYZ".</p>
TO	<p>&E(<i>a</i> TO <i>pattern</i>) has a value that is created by matching the pattern associated with <i>a</i> with the <i>pattern</i> in this expression. A character from TO <i>pattern</i>, if located in the IS <i>pattern</i> and if found the corresponding character from <i>a</i>, is moved to the result field. If a match cannot be found, the character from the TO <i>pattern</i> is moved to the result field. For example:</p> <ul style="list-style-type: none"> To reverse a string: &E('PLEH' IS 'ABCD' TO 'DCBA') yields 'HELP'. To insert delimiters: &E('HHMM' IS 'ABCD' TO 'AB:CD') yields 'HH:MM'. To remove delimiters: &E('HH:MM' IS 'ABCDE' TO 'ABDE') yields 'HHMM'. <p>The last three examples all show the use of both the IS and TO operators for clarity. The IS operator is not really necessary because all variables automatically have the default pattern described in the IS operator. Therefore:</p> <p>&E('PLEH' TO 'DCBA') yields 'HELP'</p> <p>&E('THISAMEG ' TO 'ABCDICDIEIFGDDEHG') yields 'THIS IS A MESSAGE'.</p>

Date conversion special operators

DataInterchange allows for any-to-any date conversions. The format of the any-to-any date conversion operator is:

&E(*variable* FD *mask* TD *mask*)

where:

variable The value to be converted.

FD From Date operator which signals that the following variable establishes the mask that describes the date within *variable*.

TD To Date operator which signals that the following variable establishes the mask that describes the date that is wanted.

mask The mask that describes the FROM or TO date. A mask consists of the following symbols which identify the date provided or date wanted. Symbols may be in upper or lower case. Any value in the mask that is not one of the symbols below is expected to be physically part of the source data (FD) or will become physically part of the result data (TD).

CC Century

YY	Year
MM	Month of year
DD	Day of month
D	Day of month as a single character, if possible
HH	Hour of day
MM	Minute of hour
II	Minute of hour
	MM can be used when it immediately follows HH as in HHMM; however, if you want minute followed by hour, you must use IIHH, because MMHH would be interpreted as month of year and Hour of day.
SS	Second of minute
WW	Week of Year (1 through 52)
K	Day of Week (Monday=1, Tuesday=2, etc.)
	D can be used if it immediately follows WW as in WWD; however, if you want day of week followed by week, you must use KWW, because DWW would be interpreted to be day of month and Week of year.
JJJ	Julian day of year
Q	Quarter (1,2,3,4)
E	Semester (1,2)
ZZZ	Time zone
TM	Textual month (i.e., January, February, etc.)
	TM may be followed by the name of a translate table to convert a textual month to a numeric month (FD), or from a numeric month to a textual month (FD). If a table name is not provided, the default table names are DIMONTXT to translate from text to numeric and DIMONNUM to translate from numeric to text. A table name is indicated using parenthesis, for example; TM(<i>tablename</i>), where <i>tablename</i> must be a constant.

After processing the From Date and before creating the To Date, the following processing will be done.

1. SS (seconds) will be defaulted to 0.
2. CC (century) will be defaulted to 19 when the YY (year) is greater than 10 and to 20 otherwise.
3. JJJ (julian day) will be created based on WW (week of year) and K (day of week) if not otherwise provided and WW and K were provided.
4. JJJ (julian day) will be created based on MM (month of year) and DD (day of month) if not otherwise provided and MM and DD were provided.
5. JJJ (julian day) will be used to determine WW (week of year), K (day of week), if either WW or K was not provided.

6. JJJ (julian day) will be used to determine MM (month of year), DD (day of month), if either MM or DD was not provided.
7. Q (Quarter) will be determined based on MM (month of year) if not otherwise provided.
8. E (Semester) will be determined based on MM (month of year) if not otherwise provided.

The following are some examples using CONSTANTS for all values.

- Simple example to remove delimiters.
`&E('96/06/07' FD 'YY/MM/DD' TD 'YYMMDD')` yields '960607'
- Simple example to remove delimiters and rearrange
`&E('96/06/07' FD 'YY/MM/DD' TD 'MMDDYY')` yields '060796'
`&E('96/06/07 EDT' FD 'YY/MM/DD ZZZ' TD 'ZZZ MMDDYY')` yields 'EDT 060796'
- Change delimiters and convert from one form to another
`&E('96/06/07' FD 'YY/MM/DD' TD 'YY:JJJ')` yields '96:157'
- Textual month to Numeric month
`&E('June 7, 1996' FD 'TM D, CCYY' TD 'YYMMDD')` yields '960607'
- Numeric month to Textual month
`&E('060796' FD 'MMDDYY' TD 'TM D, CCYY')` yields 'June 7, 1996'
- Numeric month to Textual month with a special translate table that uses abbreviations for the months
`0 FD 'MMDDYY' TD 'DDTM(ABBREV)CCYY')` yields '07JUN1996'

Newer releases of X12 and EDIFACT standards contain segments with variable date/time formats. The format is determined by a qualifier value in the segment. DataInterchange provides two tables for dynamically translating the qualifier into a mask. Table EDIXDTMSK is used for X12 and table EDIEDTMSK is used for EDIFACT.

Assume the following X12 data is received in the DTP segment (Date/Time period):

- DTP**D8*19940927!
 Where D8 is the date/time period format qualifier (CCYYMMDD) and 19940927 is the date/time period.

Assume your application requires the date in YYMMDD format. You map the DTP segment as follows:

- DTP03 - &SAVE Qual
 Results in Qual is D8
- DTP03 - &SAVE Date
 Results in Date is 19940927
- DTP03 - &FORCE &E(Date FD (QUAL TS 'DIXDTMSK') TD 'YYMMDD')
 Results of (Qual TS 'DIXDTMSK') is CCYYMMDD
 Results of &FORCE is 940927

Order of precedence

During processing, all expressions are evaluated from left to right. The order of precedence is:

1. Unary minus (-)
2. Rounding (RD) and truncating (TU)
3. Special operators (UE, TS, TL, IN, IS, TO, SC, SR, FD, TD)
4. Multiply (*), Divide (/)
5. Addition (+), Subtraction (-)
6. Relational Operators (GT, GE, LT, LE, EQ, NE)
7. Boolean (AND)
8. Boolean (OR)

Precedence can be overridden with parentheses embedded within an expression. For example, $\&E(2+3*5)$ equals 17 because the multiplication is done first, then the addition. $\&E((2+3)*5)$ equals 25 because the parentheses indicate that the addition is done first, then the multiplication.

DI variables

DataInterchange has reserved the prefix DI for variables that will be reserved for use DataInterchange to accomplish special functions. The following DI variables are currently available and are used with the `&SET` keyword:

DIAPPPFILE	<p>This variable may be used to change the name of the file to which the translation application data will be written during a Receive Translate. It will override any value that was used in the receive usage or in the application data format definition. It provides the capability for data that is being received to influence the final destination for the data. For example, the statement</p> <pre>&SET DIAPPPFILE SPECIAL</pre> <p>would force the current transaction to be written to the application file identified by the ddname SPECIAL.</p>
DIAPPTYPE	<p>This variable sets the application file type that corresponds with the file name provided by DIAPPPFILE.</p>
DIAUTOCC	<p>Allows automatic century manipulation for both inbound and outbound translation. Century will be automatically added or removed from the date using the length of the standard data element or application field. For example, the following statement uses a value of 1:</p> <pre>&SET DIAUTOCC 1</pre>
DICCCTRL	<p>Use to remove the century control year from the translator and allow selection of the century control year. If year is greater than 10, century is 19; otherwise century is 20. The century control year is 10. For example, if year is less than 95, century is 20, control year is 95:</p> <pre>&SET DICCCTRL 95</pre>

DICUSERDATA	<p>This variable is used to set the data value that will be inserted in the TRCB field <code>cuserdata</code>. This field is copied to any output 'C' record before the 'C' record is written. Received data can be placed into a named variable in any combination up to 256 bytes. Then use the reserved variable <code>DICUSERDATA</code> anytime the value of the named variable needs to be placed into the TRCB. For example, suppose a named variable <i>tvar</i> has been created and filled with the data from a previously mapped data element.</p> <p>That data can be placed in the TRCB <code>cuserdata</code> field by including the following in a map:</p> <pre>&SET DICUSERDATA &E(TVAR)</pre>
DIERRFILTER	<p>This variable can be used to control which errors are actually meaningful to you at a point in time during a translation. A description of the error filter can be found in the DataInterchange Programmer's Reference under the section "Error Filtering" and under the description for "ERRFILTER."</p>
DIEXPTRACE	<p>This variable, when given a nonzero value (<code>&SET DIEXPTRACE 1</code>), causes DataInterchange to create a TRACE of the results of all expression evaluations. When tracing is active, DataInterchange will write out message TR0411 to the PRTFILE for each expression. The message will show the expression being evaluated and the result of the evaluation. Tracing will remain active until the <code>DIEXPTRACE</code> is given a zero value (<code>&SET DIEXPTRACE 0</code>).</p> <p>NOTE: The TR0410 and TR0411 messages always occur as the first messages for a transaction. They are not merged with any other error messages for the transaction.</p>
DIMAPCHAIN	<p>This variable can be used when an inbound transaction is required by more than one application program. It allows more than one mapping to be executed for the specified transaction. The last value given to <code>DIMAPCHAIN</code> in a mapping will establish the application sender ID value that will be used to locate the next mapping to execute. For example, if MAPABC had this coded:</p> <pre>&SET DIMAPCHAIN APPLICATIONB</pre> <p>the inbound transaction would be translated using map MAPABC, and then it would be translated using the map that is associated with application sender ID APPLICATIONB. The <code>DIMAPCHAIN</code> command will cause all maps indicated by each <code>DIMAPCHAIN</code> command to be translated, whereas the <code>DIMAPSWITCH</code> command will stop translating the map that has the <code>DIMAPSWITCH</code> variable in it, and literally switch to the new map indicated in the command.</p>

DIMAPSWITCH	<p>This variable can be used when data being received needs to be inspected before it can be determined exactly what mapping should be done against the transaction. It allows you to switch the map that is being executed dynamically based on the data that is being received. A map could be created to initially look at the data being received. Only those data elements necessary to make a mapping decision would be mapped. DataInterchange would determine the real map to be used by interpreting values resulting from conditional logic expression. For example, a map would contain conditional logic expression:</p> <pre>&IF(X > Y) &SET DIMAPSWITCH APPLICATIONA</pre> <p>Here, if X is greater than Y, the mapping identified with an application sender ID value of APPLICATIONA would be used to translate the transaction.</p>
DISAPSEQ	<p>This variable can be used to allow saving of the SAP IDOC record sequence number on the first error encountered during outbound processing. The sequence number may be provided through the application or using the DataInterchange accumulators. Variable DISAPSEQ is captured in the SAP status record to indicate the first record in error. For more information, see the <i>DataInterchange Programmer's Reference</i>.</p>
DIVALLEVEL	<p>This variable can be used to control the level of validation done. It can have the same values as the validation level specified in a usage record, which are: 0 (no validation), 1 (validation tables activated), and 2 (validation tables plus type checking). Any value other than 0, 1, or 2 will be treated as a 0.</p>
DIVALTTYPE	<p>This variable can be used to control the data types for which data type checking is done (validation level of 2). The types that may be specified are DT, TM, N, R, CH, AN, A, and HX. They must be specified in uppercase and separated by a comma. Any value specified that is not in the list above will be ignored. For example, to activate DT, TM and HX validation, the following could be done:</p> <pre>&SET DIVALTTYPE DT,TM,HX</pre>
DIVARTRACE	<p>This variable, when given a nonzero value (&SET DIVARTRACE 1), causes DataInterchange to create a TRACE of all accesses to variables. When tracing is active, DataInterchange will write out message TR0410 to the PRTRFILE for each variable access. The message will indicate the variable being accessed and its current value. Tracing will remain active until the DIVARTRACE is given a zero value (&SET DIVARTRACE 0).</p>

Mapping techniques for literal keywords

As you will see in the following examples, deciding where to save variables, execute expressions, and subsequently use variables is fundamental. To determine where these operations should be done, it is essential to understand the order in which the translator executes element mapping instructions. As stated earlier, transactions are always processed starting with the first data element of the first segment and proceeding to the next element of the same segment, after which the next segment is processed, and so on. If one element has repeat mappings, the instructions are executed in a top-down fashion. It is equally important to understand the difference between a mapping that may contribute to the output versus one that does not. For example, specifying **&SAVE variable** or **&SET variable** will not contribute to the output directly. Only direct mappings such as specifying an Application Field Name, forcing a literal value, specifying **&USE variable**, or evaluating an **&IF** expression, may contribute to the output. Determining where to save and use variables is different for inbound and outbound translation.

For send processing, proper placement is easy for mappings that contribute to the output because you know which element requires the result. Proper placement for a mapping that does not contribute is not as obvious because there may be no data element relationship with this action. In most cases, you will save an application value into a variable, check or manipulate the variable, then use it. In other cases, it may not be appropriate to perform all these actions in successive repeat element mappings. For instance, you may have two independent looping structures (records) and you need to save a value from a particular iteration in the first loop. This particular value must be saved while the translator is processing this first loop. The saved variable can then be inspected and manipulated in the second repeating loop to provide the desired result.

In summary, the location of a mapping that saves a variable can be far from a corresponding mapping that actually uses the variable. The best technique for deciding where to save values is to do so at or near processing points in the map when the translator is working on the corresponding record.



NOTE: Typically, the map for saving a value into a variable will precede the event of actually using the variable. A feature called **&DEFERRED &USE** can be utilized in the event that output is necessary in an earlier segment than where the final result will be set.

For receive processing, the situation is reversed. It is easy to decide where to save variables because you already know which element value in the input you need to work with. Proper placement for a mapping that contributes to your output application data is not as obvious because there may be no data element relationship with this action. In most cases, you will save a data element value into a variable, check or manipulate the variable, then use it in your application record. In other cases, it may not be appropriate to perform all these actions in successive repeat element mappings. For instance, you may need to save values from two different segments, compare them, and write the result to an application field. One input element that needs to be saved is in the header section of the EDI transaction and the second is in an outer loop (the name loop for example). The output application field that needs to be output after comparing these two variables is related to the detail loop (line item loop for example). It is most appropriate to use the result within the detail loop because this may have an independent repeating record that is not associated with the name or header segments. Hence, the comparison and use of the result should be done within this detail loop that controls the creation of the detail record. The best technique for deciding where to use variables is to do so at or near processing points in the map when the translator is working on the corresponding record.



NOTE: For receive processing, the map for saving a value into a variable must precede the event of actually inspecting or using the variable. The feature of **&DEFERRED &USE** cannot be used during inbound translation.

Examples of using literal keywords and named variables

Example 1: For an inbound transaction, if your trading partner sends you the city name, state abbreviation, and zip code in three fields, but your database puts all of this information in one 30-byte field, ADDRESS, you could use &SAVE to put the address information into the single field:

1. Map the city name data element using the literal **&SAVE citystzip,1,19**.

This creates the named variable **citystzip** and places the received city in the first 19 bytes of the named variable.

2. Map the state abbreviation data element using the literal **&SAVE citystzip,20,2**.

This places the received state abbreviation in the bytes 20 and 21 of the named variable.

3. Map the zip code data element using the literal **&SAVE citystzip,22,9**.

This places the received zip code in bytes 22 through 30 of the named variable.

4. Repeat the mapping of the zip code data element. Specify application field ADDRESS and the literal **&USE citystzip**.

DataInterchange uses the value in the named variable that was concatenated in the first three steps.

Example 2: For an outbound transaction, you want to provide a telephone number, either of a specific contact (CONTACTPHONE) or of the organization (ORGPHONE). If a contact phone number is provided, you want to use it; otherwise, you want to use the organization phone number. CONTACTPHONE occurs before ORGPHONE.

1. In the first mapping of the phone number data element, specify application field CONTACTPHONE and the literal **&IFDATA &SAVE Tphone**.

This creates the named variable **Tphone** only if the CONTACTPHONE field contains data.

2. Repeat the mapping of the phone number data element. Specify application field ORGPHONE and the literal **&IFNOVAR &SAVE Tphone**.

This creates the named variable **Tphone** only if the named variable did not already exist (CONTACTPHONE did not contain any data).

3. Repeat the mapping of the phone number data element. Specify the literal **&USE Tphone**.

Example 3: For an inbound transaction, the application field NAME should receive the value from EDI standard data element 123 (CUSTNAME) in the first occurrence of the name and address loop, or from the data element (ORGNAME) in the second occurrence of the loop, if the first occurrence does not contain data.

1. When mapping the first occurrence of the loop, specify the literal **&IFDATA &SAVE Name**. Do not specify an application field.

This will create the named variable **Name** only if the first occurrence contains data.

2. When mapping the second occurrence, specify the literal **&IF(Name EQ ") &LSAVE Name**. Do not specify an application field.

This will create the named variable Name only if it does not already exist (the first occurrence did not contain any data). &LSAVE is used so that the value of Name from the first occurrence of the loop is not disturbed.

3. Repeat the mapping for the second occurrence. Specify application field NAME and the literal **&USE Name**.

This will use either the value saved in step 2 (first occurrence did not contain a value) or the value saved in step 1 (first occurrence did contain a value). All future occurrences of the loop and unrelated loops and/or segments will only see the value of Name saved in step 1.

Example 4: For an inbound transaction, the application field NAME should receive the value from EDI standard data element 123 (CUSTNAME) in the second occurrence of the name and address loop, or from the data element (ORGNAME) in the first occurrence of the loop, if the second occurrence does not contain data.

1. When mapping the first occurrence of the loop, specify the literal **&SAVE Name**. Do not specify an application field.

This will create the named variable Name.

2. When mapping the second occurrence, specify the literal **&IFDATA &SAVE Name**. Do not specify an application field.

This will overlay the named variable Name only if the second occurrence contains data.

3. Repeat the mapping for the second occurrence. Specify application field NAME and the literal **&USE Name**.

This will use either the value saved in step 2 (second occurrence contained a value) or the value saved in step 1 (second occurrence did not contain a value).

Example 5a: For an outbound transaction, either application field ORDQTY or MINQTY should supply a value for an EDI standard data element. The field with the largest value should be used. If neither ORDQTY or MINQTY contain data then a value of 100 should be used.

1. In the first mapping of the data element, specify application field ORDQTY and the literal &SAVE QTY1.

This saves the value of the application field to a named variable. If the application field does not contain a value, the variable value is zero.

2. Repeat the mapping of the data element. Specify application field MINQTY and the literal &SAVE QTY2.

This saves the value of the application field to a named variable. If the application field does not contain a value, the variable value is zero.

3. Repeat the mapping of the data element. Specify the literal &IF(QTY1 >= QTY2) &USE QTY1 100. Do not specify an application field.

This compares the values of the named variables. If QTY1 is greater than or equal to QTY2, DataInterchange will use QTY1. If neither ORDQTY or MINQTY contain a value then both QTY1 and QTY2 will have a value of 0 and will therefore be equal. However, a value of 0 is not significant, which causes DataInterchange to use the default literal value of 100.

4. Repeat the mapping of data element 123. Specify the literal &IF(QTY1 < QTY2) &USE QTY2. Do not specify an application field.

This compares the values of the named variables. If QTY1 is less than QTY2, DataInterchange will use QTY2.

Example 5b: Assume you have created a mapping similar to that of example 5a, but forgot to do step 1. Because it is necessary to define QTY1 before using it, you need to insert a mapping of this data element, making it the first occurrence of the loop. Instead of remapping all of the occurrences, perform the following:

1. Drag application field ORDQTY to the data element.
2. Open the Mapping Data Element Editor for the new mapping.
3. Enter the literal &SAVE QTY1 and press OK.
4. Drag the new mapping to the appropriate location and drop it there.

Example 6: Your application can generate three different discount rates: the regular discount (REGDISC), a volume discount (VOLDISC), and a special discount (SPECDISC). For an outbound transaction, if application field SPECDISC contains a value, you want to use it for EDI standard data element 456. However, if SPECDISC does not contain a value, then the larger value of either REGDISC or VOLDISC should be used.

1. In the first mapping of data element 456, specify application field REGDISC and the literal **&SAVE REGDISC**.

This saves the value of the application field to a named variable. If the application field does not contain a value, the variable value is zero.

2. Repeat the mapping of data element 456. Specify application field VOLDISC and the literal **&SAVE VOLDISC**.

This saves the value of the application field to a named variable. If the application field does not contain a value, the variable value is zero.

3. Repeat the mapping of data element 456. Specify application field SPECDISC and the literal **&IF(REGDISC >= VOLDISC) &E(REGDISC)**.

This mapping will be executed only if the value of REGDISC is greater or equal to the value of VOLDISC. If this is the case, then SPECDISC will be mapped to the data element. However, if SPECDISC does not contain any data, then the default literal value of **&E(REGDISC)** will be used.

4. Repeat the mapping of data element 456. Specify application field SPECDISC and the literal **&IF(REGDISC < VOLDISC) &E(VOLDISC)**.

This mapping will be executed only if the value of REGDISC is less than the value of VOLDISC. If this is the case, then SPECDISC will be mapped to the data element. However, if SPECDISC does not contain any data, then the default literal value of **&E(VOLDISC)** will be used.

Notes on Examples 5 and 6: Examples 5 and 6 illustrate the differences between **&IF**, **&USE**, and **&E**:

- **&IF** is used to determine if a mapping should be executed. The value of the **&IF** expression is not used in the mapping; it only controls the execution of the mapping. If the expression is true (nonzero value), the mapping is executed. If the expression is false (zero value) the expression is not executed. In examples 5 and 6, only one of the maps in steps 3 and 4 will be executed because the expressions are mutually exclusive.
- **&USE** indicates that a named variable should be used as the primary source for data in the mapping. An application field cannot be specified, but you can have a default literal value if the variable name being used does not contain any data.
- **&E** is used exactly the same as a literal value, but instead of having a constant literal value, **&E** allows a literal value to be computed, taken, or computed and taken from a named variable. If the last operator of an expression is a Boolean or comparison operator, then the value of the expression will either be true (1) or false (0), and you can map these values.

For example, if a data element should contain a 1 if FLD1 is greater than FLD2, or a 0 if it is not, then mapping the expression `&E(FLD1 > FLD2)` would result in either a 1 or 0 being moved to the EDI standard data element. If a data element should contain a 1 if FLD1 is greater than FLD2, or nothing if it is not, then the conditional mapping `&IF(FLD1 > FLD2) 1` would map the constant value of 1 only when FLD1 is greater than FLD2.

Example 7: For an outbound transaction, data element 321 should have a value of S if the value of application field SIZE is 6, 7, or 8; a value of M if SIZE is 9, 10, 11, or 12; and a value of L if SIZE is 13, 14, or 15. These values are specified in the translation table SIZETAB. However, for this transaction, SIZE can also be less than 6, in which case data element 321 should have a value of XS, or greater than 15, in which case data element 321 should have a value of XL.

1. In the first mapping of data element 321, specify application field SIZE and the literal `&SAVE Size`.

This saves the value of the application field to a named variable. If the application field does not contain a value, the variable value is zero.

2. Repeat the mapping of data element 321. Specify the literal `&IF(Size > 0 AND Size < 6) XS`.

This compares the value of the named variable to 6 and 0. If the value is less than 6 but greater than 0, DataInterchange uses the value XS.

3. Repeat the mapping of data element 321. Specify the literal `&IF(Size > 15) XL`.

This compares the value of the named variable to 15. If the value is greater than 15, DataInterchange uses the value XL.

4. Repeat the mapping of data element 321. Specify application field SIZE and the literal `&IF(Size >= 6 AND Size <= 15) &E(Size TS 'SIZETAB')`.

This compares the value of the named variable to 6 and 15. If the value is greater than or equal to 6, and less than or equal to 15, DataInterchange uses the translation table SIZETAB to determine the corresponding value for the named variable.

Example 8: Assume you have a lumber supply business and your trading partners are home builders in the area. The unit of measure on their orders ranges from inches to rods (one rod equals 5½ yards). You have implemented just-in-time inventory processes so if you receive an order that requires additional inventory, your order is immediately sent to your supplier. Your application stores all measurements in board feet, and therefore must convert all incoming data to board feet.

Your application fields are QUANTITY and UNITMEAS. Using the conditional processing literals, you would:

1. Map the unit of measure data element, using the literal `&SAVE UOM`. Do not specify an application field.

This saves the present value of the standard data element in the named variable UOM.

2. Repeat the mapping of the unit of measure data element, using the literal `&IF(UOM EQ 'IN' OR UOM EQ 'BF' OR UOM EQ 'RD') &SET UOMOK 1`. Do not specify an application field.

This statement enables you to determine if you have only the values you want, and if the statement is true, DataInterchange puts a 1 in the named variable UOMOK for later use.

3. Repeat the mapping of the unit of measure data element. Specify application field UNITMEAS and the literal `&IF (UOMOK EQ 1) &FORCE BF`.

This forces the value 'BF' into the application field only when the unit of measure is a valid value. The steps that follow will convert the data received into board feet.

4. Map the quantity data element, using the literal `&SAVE QTY`. Do not specify an application field.

This saves the present value of the standard data element in the named variable QTY.

5. Repeat the mapping of the quantity data element. Specify application field QUANTITY and the literal `&IF(UOM EQ 'IN') &FORCE &E(QTY/144)`.

This statement saves the value of quantity converted to board feet in the application field QUANTITY only if the incoming unit of measure was 'IN'.

6. Repeat the mapping of the quantity data element. Specify application field QUANTITY and the literal `&IF(UOM EQ 'RD') &FORCE &E(QTY*198/144)`.

This statement saves the value of quantity converted to board feet in the application field QUANTITY only if the incoming unit of measure was 'RD'.

7. Repeat the mapping of the quantity data element. Specify application field QUANTITY and the literal `&IF(UOM EQ 'BF') &FORCE &E(QTY)`.

This statement saves the value of quantity converted to board feet in the application field QUANTITY only if the incoming unit of measure was 'BF'.

8. Repeat the mapping of the unit of measure data element, using the literal `&ASSERT1(UOMOK EQ 1) &ERR(2,100, 'Invalid unit of measure')`. Do not specify an application field.

This statement creates a translation error for anything that does not meet our criteria. Assume you received something in yards. You could add a repeat mapping for yards, then retranslate. The transaction would then pass through the translation.

An alternative to using conditional processing literals would be a single mapping using a translation table. For example, you could set up a translation table UOMDIV with the following entries:

Local	Standard
IN	144
RD	1.375
BF	1

If a translation table is used, then the mapping could be reduced to the following:

1. Map the unit of measure data element, using the literal **&SAVE UOM**.
2. Repeat the mapping of the unit of measure field, using the literal **&SET divisor &E(UOM TS 'UOMDIV')**.
3. Repeat the mapping of the unit of measure field, using the literal **&IF divisor NE 0) &FORCE BF**.
4. Map the quantity data element using the literal **&SAVE QTY**.
5. Repeat the mapping of the quantity data element, using the literal **&FORCE &E(QTY/divisor)**.
6. Repeat the mapping of the unit of measure data element, using the literal **&ASSERT1(divisor NE 0) &ERR(2,100,, 'Invalid unit of measure')**.

This method might be preferred if the values of UOM are expected to change. If this happens, only the UOMDIV table needs to be updated rather than changing the mapping.

Example 9: For an outbound transaction, the application field TEST1 is an N2 data type and contains the value 100. You need to build a variable that contains "XX" in the first and second positions, "01" in the third, fourth, fifth, and sixth positions, and the value in the application field TEST1 beginning in the seventh position.

To put this information into a single variable, do the following:

1. Map a data element using the literal **&SET TVAR**.
This creates the named variable TVAR and sets the variable to null.
2. Repeat the data element and map the data element using the literal **&SET TVAR,*,2 XX**.
This places "XX" in bytes 1 and 2 of the variable TVAR.
3. Repeat the data element and map the data element using the literal **&SET TVAR,*,4 01**.
This places "01" in bytes 3 through 6 of the variable TVAR.
4. Repeat the data element and map the data element using the application field TEST1 and a literal **&SAVE TVAR,*,7**.
This places "100" in bytes 7 through 9 of the variable TVAR. The variable TVAR now contains the following:

XX01 100



NOTE: At this point, the variable TVAR has been normalized to the data type of the application field TEST1, which is defined as an N2 data type.

To map TVAR position 3 through 6 to an EDI standard data element with an ID data type, you would need to do the following because of the normalization of the variable to an N2 data type:

1. Map a data element using the literal `&E(CHAR(TVAR) SC 3,4)`.

The `&E` allows the literal to be computed. The `CHAR` forces the variable TVAR to have a data type of character for this instruction. The `SC` is a scaling function but can be used to substring character data. The resulting value would be "01".

An alternative to using the scaling function is as follows:

1. Map a data element using the literal `&E(CHAR(TVAR) TO 'CDEF')`.

The `TO 'CDEF'` is used to create the value by matching the default pattern ('ABCDEF-GHIJK...') with the 'CDEF' pattern in this expression. The resulting value would be "01".

2. Map a data element using the literal `&USE TVAR,3,4`.

The resulting value would be "0.01".

To map TVAR position 7 through 9 to an EDI standard data element with an R data type, you can move the data.

1. Map a data element using the literal `&USE TVAR,7,3`. The resulting value would be "1.00".

Example 10: For an outbound transaction, the application field TEST1 is an N2 data type and contains the value 123. Application field TEST2 is an N0 data type and contains the value 100.

To put this information into a single variable do the following:

1. Map a data element using the literal `&SET TVAR`.

This creates the named variable TVAR and sets the variable to null.

2. Repeat the data element and map the data element using the application field TEST1 and a literal `&SAVE TVAR,*,3`.

This places "123" in bytes 1 through 3 of the variable TVAR.

3. Repeat the data element and map the data element using the application field TEST2 and a literal `&SAVE TVAR,*,3`.

This places "100" in bytes 4 through 6 of the variable TVAR.

The variable TVAR now contains the following:

123100



NOTE: At this point the variable TVAR has been normalized to the data type of the application field TEST1 which is defined as an N0 data type.

To map TVAR position 1 through 3 to an EDI standard data element with an R data type, you would need to do the following because of the normalization of the variable to an N2 data type:

1. Map a data element using the literal `&SET TEMPVAR &E(CHAR(TVAR) SC 1.3)`.

The `&E` allows the literal to be computed. The `CHAR` forces the variable `TVAR` to have a data type of character for this instruction. The `SC` is a scaling function but can be used to substring character data. The resulting value would be "123".

2. Repeat the data element and map the data element using the literal `&E(TEMPVAR / 100)`.

The resulting value would be 1.23.

An alternative is as follows:

1. Map a data element using the literal `&USE TVAR,1,3`.

The resulting value would be 123.

Control data literals

Audit and control are generally high priority items. If you have the need to add the internal trading partner nickname of your trading partner to your application data for each transaction, use the `&TPID` or `&TPNICKN` literals. If you need to include the data format ID associated with the current transaction, use the `&FORMAT` literal. Finally, if you require the application control value associated with the current transaction, use the `&ACFIELD` literal. This value will only be correct after ALL the fields that comprise the application control field have been processed.

Mapping specific service segment fields (receive only)

The table below lists the literals that are provided so that every field within every service segment can be accessed using a combination of the segment ID (ISA, UNB, STX, and so forth) concatenated with a 2-byte number indicating the field within the segment wanted. The names created with this concatenation match the names of the fields defined in the E, I, T, U, and X profiles.

Invalid names (for example, ISA44) are not flagged as errors, but return no data. Using names that do not match the envelope type being received (for example, using ISA01 when UN/EDIFACT service segments (UNB) are being used) is not an error, but no data is returned.

Table 87. Service Segments

Literal:	nn Value:	Segment:
<code>&ISAnn</code>	01 through 16	ISA
<code>&GSnn</code>	01 through 08	GS
<code>&STnn</code>	01 through 02	ST
<code>&SEnn</code>	01 through 02	SE
<code>&GEnn</code>	01 through 02	GE
<code>&IEAnn</code>	01 through 02	IEA
<code>&UNBnn</code>	01 through 18	UNB

Literal:	nn Value:	Segment:
&UNGnn	01 through 13	UNG
&UNHnn	01 through 09	UNH
&UNTnn	01 through 02	UNT
&UNEnn	01 through 02	UNE
&UNZnn	01 through 02	UNZ
&STXnn	01 through 12	STX
&BATnn	01 through 01	BAT
&MHDnn	01 through 06	MHD
&MTRnn	01 through 01	MTR
&EOBnn	01 through 01	EOB
&ENDnn	01 through 01	END
&BGnn	01 through 07	BG
&EGnn	01 through 04	EG
&ICSnn	01 through 10	ICS
&ICEnn	01 through 02	ICE

Mapping generic service segment fields (receive only)

You can map received envelope data to application fields by using substitution keywords in the *Literal* field. The keywords indicate which service segment field is mapped to the application field.

The table below describes the substitution keywords you can use to map service segment fields. The Envelope Data Type column indicates the required data type for the service segment field. The EDI Standard Data Type indicates the data type that DataInterchange uses for conversions from an EDI standard data type to the application data type.

Table 88. Keywords for Mapping Envelope Data (Receive Only)

Keyword:	Envelope Data Type:	EDI Standard Data Type:	Envelope Data Mapped to Application:
&I		A	Entire interchange service segment, up to the length of the application field
&ICN	CN or IV	AN	Interchange control number
&IIS	IS, AS, or RS	A	Interchange sender ID
&IIR	IR, AR, or RR	A	Interchange receiver ID
&IDT	DT	DT	Interchange date
&ITM	TM	TM	Interchange time

Keyword:	Envelope Data Type:	EDI Standard Data Type:	Envelope Data Mapped to Application:
&IPW	PW	A	Interchange password
&IAP	AP	A	Interchange application reference
&IVR	VR	A	Interchange version/release
&IGT		N0	Interchange total number of groups
&ICT	CT	N0	Interchange control total from the interchange trailer segment
&ITT		N0	Interchange total number of transactions
&G		A	Entire group service segment, up to the length of the application field
&GCN	CN or IV	AN	Group control number
&GFG	FG	A	Functional group ID
&GAS	AS, IS, or RS	A	Group application sender ID
&GAR	AR, IR, or RR	A	Group application receiver ID
&GDT	DT	DT	Group date
>M	TM	TM	Group time
&GPW	PW	A	Group password
&GVR	VR	A	Group version
&GLV	LV	A	Group release
>T		N0	Group total number of transactions
&T		A	Entire transaction service segment, up to the length of the application field
&TCN	CN or IV	AN	Transaction control number. The &TCN keyword is valid for both send and receive. See “Mapping service segment fields (send only)” on page 395 for an explanation on the use of &TCN for application assigned control numbers.
&TTC	TC	A	Transaction code
&TVR	VR	A	Transaction version
&TLV	LV	A	Transaction release

Keyword:	Envelope Data Type:	EDI Standard Data Type:	Envelope Data Mapped to Application:
&TTS		N0	Transaction total number of segments

DataInterchange interprets any literal beginning with an ampersand (&) as a special keyword. To use a literal that begins with an ampersand, use two ampersands. The translator discards the first one and uses the remaining characters as literal data. For example, if you enter &T in the *Literal* field, the translator moves the entire transaction service segment to the application data. If you enter &&T in the *Literal* field, the translator removes the first & and uses &T as literal data. The service segments (ISA, GS, ST, UNB, UNG, and so on) are not provided in the list of segments that can be mapped for trading partner transactions. Because they are not provided in the list, a direct mapping of a field from a service segment is not possible. In order to use one of the literals from Table 86 on page 360 or Table 88 on page 393, you have to map, or repeat map, some other data element defined in the transaction. When one of the special literals is used, DataInterchange knows that the value of the data element being mapped should be ignored, and the value of the special literal should be used instead.

Restrict the mapping of service segment fields to data elements in nonrepeating segments. Select a data element in the first nonrepeating segment that you know will always be present in the received transaction data. The segment *must* be present in the data being received for the mapping instructions to be executed. Repeat the element mapping as many times as is necessary to map (or &SAVE) all of the service segment fields you need. DataInterchange recognizes the substitution keywords and moves the value from the associated service segment field rather than the transaction data element that is currently being mapped to the application field or named variable. Data conversions, table translations, and user exits are possible when service segment special literals are used.

Mapping service segment fields (send only)

During the send mapping process, the &TCN special literal may be used to indicate the application field which contains the message or transaction control number. Any field from a segment currently mapped may be chosen and either mapped or repeated, and the special literal &TCN used to identify the application field containing the control number.

- If the application field is part of a record that occurs more than once, the first record is the only one that will be used.
- It is possible to have more than one mapping which contributes to the transaction control number. The data from each mapping will be concatenated with the current data from previous mappings.
- If &DATE is used in the mapping, the format will be *yyyymmdd*. If &TIME is used in the mapping, the format will be *hhmmss*.

The transaction control number is extracted at translate time so if delayed enveloping is used and &DATE or &TIME, or both, are used to construct the control number, the date and time will be the date and time of translation and not the date and time of enveloping.

- Translation tables, validation tables, and user exits can be used during &TCN mappings the same as they can be used in any other mapping.

- The transaction control number generated by the application will be truncated to the maximum length for a control number allowed by the EDI standard, never to exceed fourteen bytes.
- It is possible to combine delayed enveloping with application assignment of control numbers. If this is done, when an envelope operation is requested, the transactions will be sorted such that all transactions for which DataInterchange will assign control numbers will occur before the transactions for which the application has assigned control numbers. Also, during an envelope operation, a switch from a transaction that requires DataInterchange to assign the control number to a transaction with application assigned control numbers will cause a new interchange to be started. Transactions with application assigned control numbers will be sorted by the value of the transaction control number.

Error message TR0115 will be displayed if the application field containing the control number was not provided, if it contained all blanks, or was otherwise invalid.

Error message TR0116 will be issued if the control number assigned is a duplicate within the group/interchange. The translator requires that message control numbers must be unique within the group. If groups are not being used, then message control numbers must be unique within the interchange.

These errors are considered level 3 errors.

You must fix the application so that the message control numbers are unique within the interchange or group.

The following table shows the type of validation done for every possible mapping between application data types (down) and EDI standard data types (across) during a TRANSLATE TO STANDARD processing.

Validation during mapping

The following list describes the type of validation used during translation, based on the type of data being validated.

Data type:	Validation:
A	The ALPHANUM validation table shipped with DataInterchange, with numeric digits (0 through 9) removed, is used to validate the data.
AN	The ALPHANUM validation table shipped with DataInterchange is used to validate the data.
BIN	A generic data type that encompasses the P, L, Z, B, I and H data types.
CH	The CHARSET validation table shipped with DataInterchange is used to validate the data.
N	Only digits, leading or trailing sign characters, and leading or trailing blanks are allowed.
R	Only digits, decimal notation, leading or trailing sign characters, and leading or trailing blanks are allowed.
DT	Must be a valid date according to the format specified during the mapping process.

Data type:	Validation:
TM	Must be a valid time.
-	Validation is done automatically because a data conversion is required.

The following table shows the type of validation done for every possible mapping between application data types (down) and EDI standard data types (across) during a TRANSLATE TO STANDARD processing.

Table 89: Type of Validation During Translate to Standard

Data type:	A:	AN:	BIN:	N:	R:	DT:	TM:
A	A	R	AN	N	R	DT	TM
AN	A	R	AN	N	R	DT	TM
AC	A	R	AN	N	R	DT	TM
CH	CH	R	CH	N	R	DT	TM
DT	DT	DT	DT	DT	DT	DT	DT
TM	TM	TM	TM	TM	TM	TM	TM
R	R	R	R	R	R	DT	TM
N	N	N	N	N	N	DT	TM
P	-	-	-	-	-	DT	TM
L	-	-	-	-	-	DT	TM
Z	-	-	-	-	-	DT	TM
B	-	-	-	-	-	DT	TM
I	-	-	-	-	-	DT	TM
FN	CH	R	CH	CH	CH	CH	CH
H	-	-	-	-	-	DT	TM

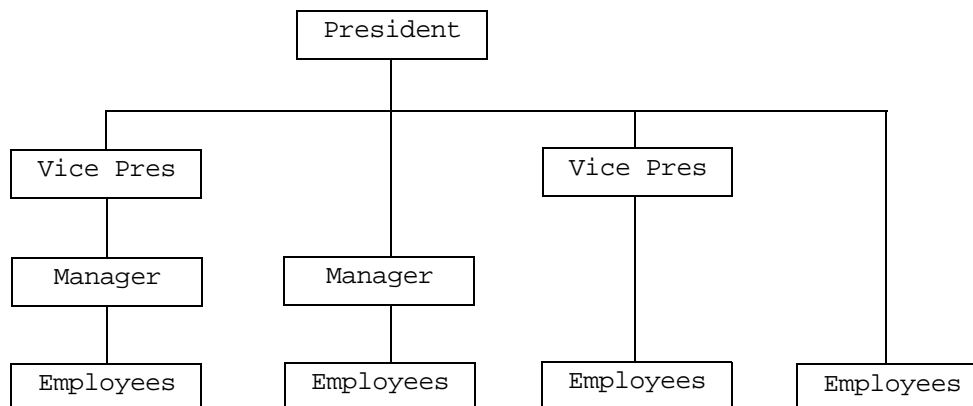
Table 90 shows the type of validation done for each possible mapping between application data types (down) and EDI standard data types (across) during TRANSLATE TO APPLICATION processing.

Table 90: Type of Validation During Translate to Application

Data type:	A:	AN:	N:	R:	DT:	TM:
A	A	A	N	R	DT	TM
AN	AN	AN	N	R	DT	TM
AC	AN	AN	N	R	DT	TM
CH	CH	CH	N	R	DT	TM
DT	DT	DT	DT	DT	DT	DT
TM	TM	TM	TM	TM	TM	TM
R	R	R	R	R	DT	TM
N	N	N	N	R	DT	TM
P	R	R	N	R	DT	TM
L	R	R	N	R	DT	TM
Z	R	R	N	R	DT	TM
B	R	R	N	R	DT	TM
I	R	R	N	R	DT	TM
FN	CH	CH	CH	CH	CH	CH
H	R	R	N	R	DT	TM

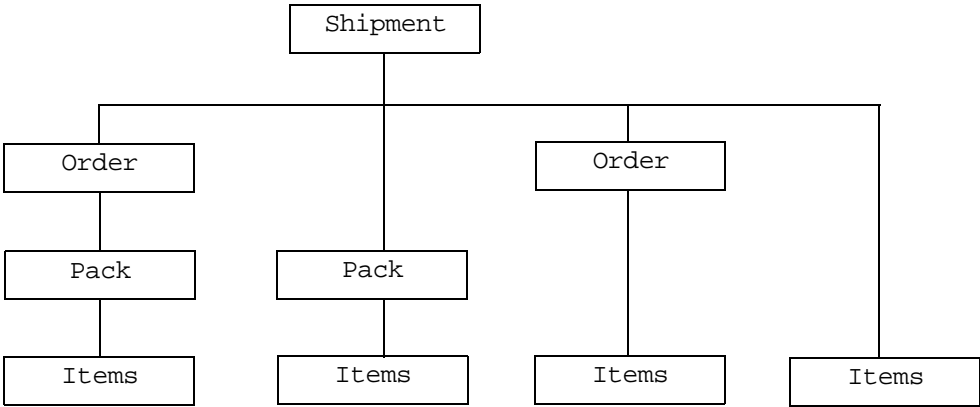
Hierarchical loops

A hierarchical loop is similar to an organization chart. Just as an organization chart shows you the various groups of people and their relationship to the whole, a hierarchical loop shows you each group of data and its relationship to the whole. For example, the following figure shows an organization chart.



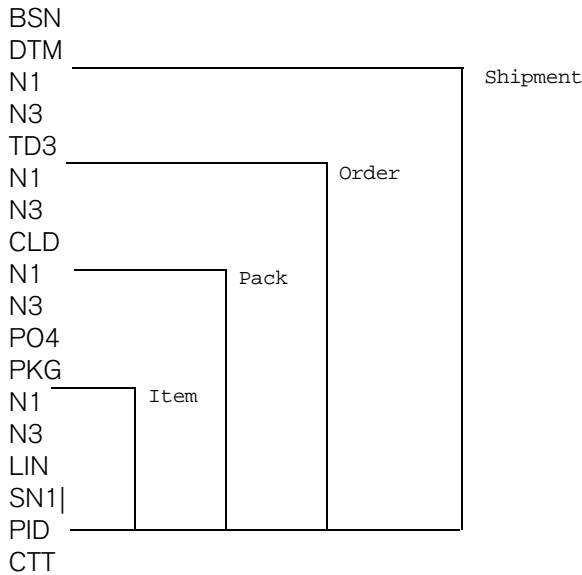
You can clearly see the different levels in the organization, and who reports to whom.

Hierarchical loops define different levels of data, which can be used in any sequence, and skipped when appropriate, allowing you to fit the loop to your data.



The HL segment

Sometimes you have to nest loops several levels deep to map all of your data, yet each level contains similar information, such as name and address. In the following figure, loops are nested four levels deep, and the N1 and N3 segments occur at the beginning of each loop.



The hierarchical level (HL) segment makes it easier to map these loops. Each HL segment contains information about the relationship of segments in a hierarchical loop to the other segments in the loop. This information is described in Table 91.

Table 91. HL Segment

Field ID	Field Name	Description
HL01	ID number	A unique number that identifies the occurrence of the HL segment. This data element is alphanumeric and has a maximum length of 12 characters. This field usually contains a sequential number that is incremented for each occurrence of the HL segment.
HL02	Parent ID	The HL01 value of the HL segment that is the parent of the current HL segment.
HL03	Level Code	A code that indicates the level of the HL segment in the current HL loop. For example, the level code could refer to the shipment, order, or item level information in the ANSI X12 Shipping Notice transaction set.
HL04	Child code	A code that indicates if the segment has subordinate segments: 1 for subordinate segments, or 0 for no subordinate segments. The default is 0.

HL01 and HL02 provide the information for DataInterchange to determine the nesting of loops within each other.



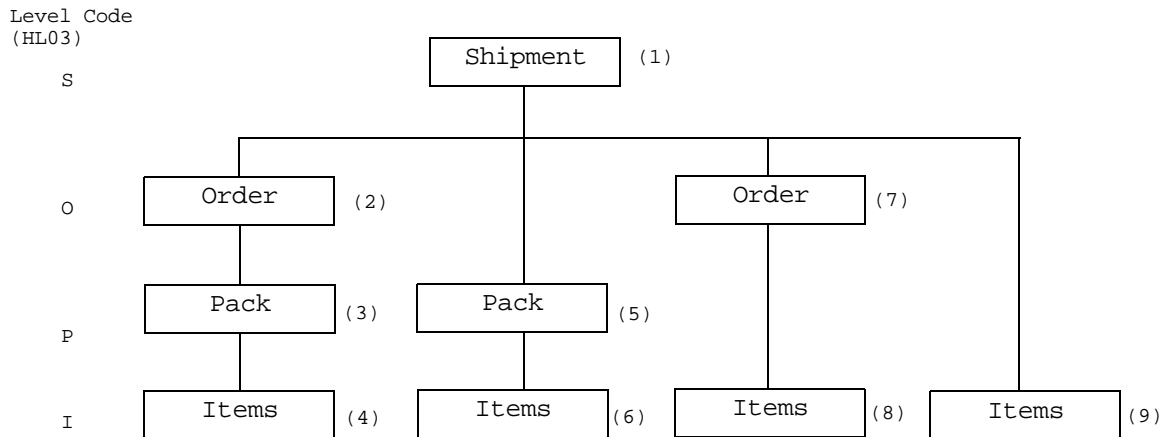
NOTE: The HL segment is not supported by all standards.

Preparing hierarchical loops

DataInterchange's HL support allows you to specify unique mapping instructions for each identifiable group of structures in a hierarchical loop. DataInterchange can handle 16 levels of nesting within the HL loop structure. To begin mapping a hierarchical level loop, follow these steps:

1. Create the hierarchy for your application data.
2. Assign node IDs to your hierarchy to uniquely identify logically grouped segments in the hierarchical loop. This ensures that DataInterchange processes your data the way you want it to be processed. The logical grouping of segments within the HL and the hierarchical relationship is defined by standards organizations or industry groups. Trading partners then agree which segments they will use and how they will be mapped.

The following diagram shows how each group of segments in your hierarchy should be numbered in a top-down, left-right order. Use this number as the node ID value.



3. Create the application data format for this hierarchical loop. The data format and the HL mapping tell DataInterchange how to build the hierarchical loop.

The application data can take one of three basic formats.

- The application data format can exactly match the hierarchy being created. This optimizes performance during translation because DataInterchange does minimal processing to match the data with the hierarchy.

```

SHIPBASE
  SHIPMENT
    ORDER1
      PACK1
        ITEMS1
      PACK2
        ITEMS2
    ORDER3
      ITEMS3
    ITEMS4
  
```

Because a structure can only occur once within a given application data format, numbers to distinguish between the different ORDER, PACK, and ITEMS structures.

- The application data format can have all the structures defined at the same level. In this case, the order of the structures passed to DataInterchange during translation helps determine how the hierarchy is built.

```
SHIPBASE
  SHIPMENT
  ORDER
  PACK
  ITEMS
```

Given the following application data:

```
SHIPMENT
ORDER
PACK
ITEMS
SHIPMENT
ITEMS
ITEMS
```

The following occurrences of the hierarchy would be built by DataInterchange during translation:

**** occurrence 1 ****

SHIPMENT,	ORDER,	PACK,	ITEMS	(based on nodes 1, 2, 3, and 4)
		PACK,	ITEMS	(based on nodes 5 and 6)
	ORDER,		ITEMS	(based on nodes 7 and 8)

**** Occurrence 2 ****

SHIPMENT,	ITEMS	(based on nodes 1 and 9)
	ITEMS	(based on node 9)

When data is passed to the translator like this, DataInterchange builds a child level of the hierarchy for each structure received in the application data between occurrences of the parent structure or structures. For example, only the ITEMS between the current SHIPMENT and the next SHIPMENT, or end of file, will be used to create this occurrence of the hierarchical loop. Any ITEMS received before the first SHIPMENT will be ignored.

- The application data can supply a structure that contains hierarchical data.

```
SHIPBASE
  HLDATA
    SHIPMENT
    ORDER
    PACK
    ITEMS
```

When data is passed to the translator like this, the fields within HLDATA define exactly how the hierarchy will be built. HLDATA must contain the value for the HL03 element, which contains the hierarchical level code. It can also contain the HL01, HL02, and HL04 elements of the HL

segment, but these are not required. If values for these fields are not present in the application data, the HL special literals, which are described in “Literal keywords for the HL segment,” may be used.

To create our sample hierarchy, the application data being passed to DataInterchange for translation could look like:

	where HLDATA contains			
	HL01	HL02	HL03	HL04
HLDATA	1		S	1
SHIPMENT				
HLDATA	2	1	0	1
ORDER				
HLDATA	3	2	P	1
PACK				
HLDATA	4	3	1	0
ITEMS				
HLDATA	5	1	P	1
PACK				
HLDATA	6	5	1	0
ITEMS				
HLDATA	7	1	0	1
ORDER				
HLDATA	8	7	1	0
ITEMS				
HLDATA	9	1	1	0
ITEMS	9	1	1	0

You can also use combinations of these basic formats. For example, you might want to provide an HLDATA structure for all but the last branch of the hierarchy. The application data format would be:

```
SHIPBASE
  HLDATA
    SHIPMENT
    ORDER
    PACK
    ITEMS
SHIPMENT2
  ITEMS2
```

Literal keywords for the HL segment

Your application data may not contain the information the translator needs to create the HL segment, but you can use the following special literals to supply the values for the HL segment.

Keyword	Description
&HLID	Supplies a sequential number for each HL segment created.
&HLPID	Supplies the HLID value for the parent of the current HL.
&HCODE	Supplies the hierarchical code associated with the current HL segment.
&HCHILD	Supplies the value 1 if the current HL segment has subordinate segments.

When you map the HL segment, you can use these literal keywords by typing them in the **Literal** field of the Map Data Element panel (TP10) when you map the data elements. For **send**, they will now be automatically mapped with their corresponding special literal when the HL loop is qualified. If different mappings for these elements are desired, the elements must be remapped.

- HL element 628 will be mapped with &HLID
- HL element 734 will be mapped with &HLPID
- HL element 735 will be mapped with &HCODE
- HL element 736 will be mapped with &HCHILD



Notices

This publication was developed for products and services offered in the United States. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Trademarks and service marks

The following terms in this publication are trademarks or service marks of the International Business Machines Corporation in the United States and other countries:

AIX
DataInterchange
DataInterchange for MVS
DataInterchange for CICS
DataInterchange Client
DB2
DB2 Connect
Expedite
IBM
IBM Global Services
MQSeries
MVS
OS/2
OS/390

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.



Glossary

A

AAR. Association of American Railroads. Represents the railroad industry in areas such as standards, public relations, and advertising.

acknowledgment. See *functional acknowledgment*, *network acknowledgment*.

Activity Log. A DataInterchange Client equivalent of the ACTLOGS profile on DataInterchange Host.

ADE. See *data format*.

ANSI. American National Standards Institute.

ANSI ASC X12. ANSI Accredited Standards Committee X12, which develops and maintains generic standards for business transactions for EDI.

application. A program that processes business information. An application that requests services from DataInterchange is an enabled application.

application data. The actual data in an application data file.

application data format. See *data format*.

application default profile. Identifies business applications, such as purchasing and accounts receivable, to DataInterchange and sets specific DataInterchange processing defaults for an application.

B

base structure. The data structure that contains all the data structures and data fields that define the application data for a single transaction.

binary format (BIN). Representation of a decimal value in which each field must be 2 or 4 bytes long. The sign (+ or -) is in the far left bit of the field, and the number value is in the remaining bits of the field. Positive numbers have a 0 in the sign bit. Negative numbers have a 1 in the sign bit and are in two's complement form.

C

CICS. Customer Information Control System.

CD-ROM. Compact Disk-Read Only Memory; a storage medium for large amounts of data needed external to the personal computer.

client-server. A computing environment in which two or more machines work together to achieve a common task.

code list. A table, supplied by DataInterchange or defined by the user, that contains all acceptable values for a single data field.

composite data element. In EDI standards, a group of related subelements, such as the elements that make up a name and address.

compound element. An item in the source or target document that contains child items. Examples are EDI segments and composite data elements, data format records and structures, and XML elements.

Config. The DataInterchange Client database that stores the information necessary for running DataInterchange Client, including messages, queries, reports, and preferences.

control number. Numbers (or masks used to create numbers) that are used to identify an interchange, group, or EDI transaction.

control string. An object compiled from a map, data format, and EDI standard transaction; it contains the instructions used by the translator to translate a document from one format to another.

control structure. The beginning and ending segments (header and trailer) of standard enveloped transmissions.

conversion. The DataInterchange Client process of transforming Host Standards, ADFs, and Trading Partner Transactions (TPTs) into DataInterchange Client format Standards, Data Formats, and Maps.

Crystal Reports. A product used by DataInterchange Client to format reports.

Customer Information Control System (CICS). An IBM licensed program that enables transactions entered at remote terminals to be processed concurrently by user-written application programs.

customize. To alter to suit the needs of a company, such as removing from an EDI standard the segments and data elements that the company does not use.

Customization data. Data not used directly by the translator, such as data formats, EDI standards, and maps.

D

data dictionary. A file containing the definitions of all the data elements of an EDI standard.

data element. A single item of data in an EDI standard, such as a purchase order number. Corresponds to a data field in a data format.

data element delimiter. A character, such as an asterisk (*), that follows the segment identifier and separates each data element in a segment. See also *element separator* and *segment ID separator*.

data field. A single item of data in a data format, such as a purchase order number. Corresponds to a data element in an EDI standard.

data format. A description of the application data for a particular transaction. A data format is composed of loops, records, data structures and fields.

data format dictionary. A file that contains data format components.

data format record. A group of logically related fields set up as a record in a data format.

data format structure. A group of related data fields in a data format, such as the fields making up the line item of an invoice. Corresponds to a composite data element in an EDI standard.

DataInterchange. The DataInterchange product; a translator of data from one document format to another; the pieces of this product include a TSO parameter entry mechanism, a CICS parameter entry mechanism, a Windows-based parameter entry mechanism (DataInterchange Client), and a translator.

DataInterchange Client. A Windows-based product for entry of parameters needed by the DataInterchange translator.

DataInterchange/MVS. The DataInterchange product used on the host; pieces include a TSO parameter entry mechanism and a translator.

DataInterchange/MVS-CICS. The CICS-based DataInterchange product.

data structure. A group of related data fields in a data format, such as the fields making up the line item of an invoice. Corresponds to a segment in a standard.

data transformation map. One of three supported map types. A data transformation map is a set of mapping instructions that describes how to translate data from a source document into a target document. Both the source and target documents can be one of several support document types.

DB2. Database 2, an IBM relational database management system.

ddname. Data definition name.

decimal notation. The character that represents a decimal point in the data.

delimiter. A character that terminates a string of characters, such as the value contained in a data element.

DI Client. DataInterchange Client; the Windows-based, client/server interface for DataInterchange.

dictionary. See *data dictionary*.

DLL. Dynamic Link Library; an executable module that is linked into the main DataInterchange executable module.

DLL/VBX. Dynamic Link Library for Visual Basic; a DLL which adheres to the conventions of the Visual Basic programming language.

document. A business document that is exchanged between two enterprises as part of a business process, such as a purchase order or invoice. A document within DataInterchange is singular. For example, it cannot contain multiple purchase orders. A document can also be represented in any syntax. For example, an XML purchase order and an EDI purchase order are both documents.

Document Type Definition (DTD). A list of all components included in the XML document and their relationship to each other. This defines the structure of an XML document.

domain. The data structure or group of data structures in a data format to and from which you should restrict the mapping of EDI repeating segments and loops.

drivers. See *DLL* and *DLL/VBX*.

DTD. See *Document Type Definition*.

E

EDI. Electronic data interchange.

EDIA. Electronic Data Interchange Association.

EDI administrator. The person responsible for setting up and maintaining DataInterchange.

EDI message. See *message*.

EDI standard. The industry-supplied, national, or international formats to which information is converted, allowing different computer systems and applications to interchange information.

EDI transaction. A single business document, such as an invoice.

EDI transaction set. A group of logically related data that make up an electronic business document, such as an invoice or purchase order.

EDIFACT. See *UN/EDIFACT*.

electronic data interchange (EDI). A method of transmitting business information over a network, between business associates who agree to follow approved national or industry standards in translating and exchanging information.

electronic transmission. The means by which information is transferred between parties, such as over a public network.

element. See *data element*.

element separator. A character that separates the data elements in a segment. See also *data element delimiter*.

encryption. The encoding and scrambling of data. Data is encrypted by the sender and decrypted by the receiver using a predetermined program and unique electronic key.

event. An occurrence that is important to a user's computer tasks, such as a software error, sending a transaction, or acknowledging a message.

Extensible Markup Language (XML). A standard metalanguage for defining markup languages that was derived from, and is a subset of SGML. It is used to represent structured documents and data.

F

field. See *data field*.

floating segment. A segment of an EDI standard that may exist in many positions relative to other segments.

forward translation table. A user-defined table that translates data values that differ between trading partners. For example, if a manufacturer and supplier have different part numbers for the same item, each company can use its own part number and have it converted to the other company's part number during translation. Forward translation tables translate local values to standard values.

functional acknowledgment. An electronic acknowledgment returned to the sender to indicate acceptance or rejection of EDI transactions.

functional group. One or more transaction sets of a similar type transmitted from the same location, enclosed by functional group header and trailer segments.

G

global variable. A variable that is shared among all instances of all documents within a translation session.

H

header. A control structure that indicates the start of an electronic transmission.

hierarchical loop. A technique for describing the relationship of data entities which are related in a parent/child manner, like a corporate organization chart. Used in mapping to group related data elements and segments such as trading partner address.

HL. See *hierarchical loop*.

I

ICS. International Control Segments.

import. The process of taking DataInterchange objects exported on another DataInterchange system and incorporating them into the receiving system.

Information Exchange. A commerce engine of IBM Interchange Services for e-business that permits users to send and receive information electronically.

interchange. The exchange of information between trading partners.

J

JCL. Job Control Language.

K

key. In a profile member, the field that identifies the member. For example, the key for members of the trading partner profile is the trading partner nickname.

L

literal. In mapping, a value that is constant for each occurrence of the translation. If you provide the literal value during mapping, the translator does not have to refer repeatedly to the source to obtain the value.

local variable. A variable that is specific to the instance of the document in which it is being used.

log file. A file in which events are recorded.

logging. The recording of events in time sequence.

loop. A repeating group of related segments in a transaction set or a repeating group of related records and loops in a data format.

loop ID. A unique code identifying a loop and the number of times the group can be repeated.

loop repeat. A number indicating the maximum number of times a loop can be used in a transaction set.

M

mailbox. If you use a mail type protocol to exchange messages with your trading partners, you will have one or more registered mailboxes. The mailbox profile is used in DataInterchange to define your mailboxes and any associated preferences.

map. A set of instructions that indicate to DataInterchange how to translate data from one format to another.

map rule. An association between a data transformation map and a trading partner.

maximum use. A number indicating the maximum number of times a segment can be used in a transaction set or the maximum number of times that a data format loop or record can repeat.

message. A free-form, usually short, communication to a trading partner. In UN/EDIFACT standards, a group of logically related data that make up an electronic business document, such as an invoice. A message is equivalent to a document.

message log. The file in which DataInterchange Client logs messages about errors that occur within the client.

MQSeries. A product of the IBM company; used to implement messaging and queueing of data groups.

MQSeries Queue profile. Represents a relationship between a logical name and a physical MQSeries queue name.

multiple-occurrence mapping. A form of mapping in which all occurrences of a loop or repeating segment are mapped to the same repeating structure in the data format.

MVS. Multiple virtual storage.

N

network acknowledgment. A response from the network indicating the status of an interchange envelope, such as sent or received.

network commands. The commands that you want DataInterchange to pass to your network, defined in the network commands profile. In the host product, this file is named NETOP.

Network Profile. The DataInterchange Client terminology for NETPROF members on DataInterchange Host.

O

ODBC. Open Data Base Connectivity. ODBC is an industry standard for making connections between a variety of software products and databases on different hardware platforms.

ODETTE. Organization for Data Exchange through Teletransmission in Europe.

Open Data Base Connectivity. See *ODBC*.

P

parse. To break down into component parts.

path qualified mapping. A form of mapping in which all occurrences of a repeating compound or simple data element are mapped to a repeating compound or simple data element in another document.

PDS. Partitioned data set.

PDS members. Groups of related information stored in partitioned data sets.

profile. Descriptive information about trading partners, network connections, and so on. Each profile can contain one or more objects or *members*. For example, the trading partner profile contains members for your trading partners (one member for trading partner address).

program directory. A document shipped with each release of a product that describes the detailed content of the product.

Q

qualifier. A data element which gives a generic segment or data element a specific meaning. Qualifiers are used in mapping single or multiple occurrences.

R

receive map. One of three supported map types. A receive map is a set of mapping instructions that describe how to translate an EDI standard transaction into a proprietary application data document.

receive usage. An association between a receive map and a trading partner.

record. A logical grouping of related data structures and fields.

release character. The character that indicates that a separator or delimiter is to be used as text data instead of as a separator or delimiter. The release character must immediately precede the delimiter.

repository data. A group of data definitions, formats, and rules/usages, that DataInterchange uses to process your data.

requestor. See *mailbox*.

reverse translation table. A user-defined table that translates data values that differ between trading partners. For example, if a manufacturer and supplier have different part numbers for the same item, each company can use its own part number and have it converted to the other company's part number during translation. Reverse translation tables translate standard values to local values.

rule. See *map rule*.

runtime data. Data used by the DataInterchange translator, such as control strings, code lists, translation tables and profiles.

S

security administrator. The person who controls access to business data and program functions.

segment. A group of related data elements. A segment is a single line in a transaction set, beginning with a function identifier and ending with a segment terminator delimiter. The data elements in the segment are separated by data element delimiters.

segment directory. A file containing the format of all segments in an EDI standard.

segment identifier. A unique identifier at the beginning of each segment consisting of two or three alphanumeric characters.

segment ID separator. The character that separates the segment identifier from the data elements in the segment.

segment terminator. The character that marks the end of a segment.

send map. One of three supported map types. A send map is a set of mapping instructions that describe how to translate a proprietary application data document into an EDI standard transaction.

send usage. An association between a send map and a trading partner.

simple element. An item in the source or target document that does not contain child items, only data. Examples are EDI data elements, data format fields, XML attributes, and PCData values.

single-occurrence mapping. A form of mapping in which each occurrence of a loop or repeating compound or simple data element in a document is mapped to a different compound or simple data element in another document.

source document definition. A description of the document layout that will be used to identify the format of the input document for a translation.

special literal. The send and receive Mapping Data Element Editors include the Literal or Mapping Command field. Literals are constant values you enter in this field, such as 123. Special literals are values you enter in this field that begin with an ampersand (&) and are command to DataInterchange, rather than constant values. For example, to use today's date, you enter &DATE.

standards. See *EDI standard*.

structure. See *data structure* or *data format structure*.

subelement. In UN/EDIFACT standards, a data element that is part of a composite data element. For example, a data element and its qualifier are subelements of a composite data element.

subelement separator. A character that separates the subelements in a composite data element.

T

tag. In UN/EDIFACT standards, the segment identifier. In export/import, a code identifies each field in the export record. Such export/import files are known as "tagged" files.

target document definition. A description of the document layout that will be used to create an output document from a translation.

TD queue. See *transient data queue*.

TDCC. Transportation Data Coordinating Committee.

TDQ. Transient data queue.

temporary storage queue (TS). Storage locations reserved for immediate results in CICS. They are deleted after the task that created them is complete and they are no longer necessary.

TPT. Trading partner transaction. See *map*.

trading partner profile. The profile that defines your trading partners, including information about network account numbers, user IDs, who pays for network charges, etc.

trading partners. Business associates, such as a manufacturer and a supplier, who agree to exchange information using electronic data interchange.

trading partner transaction. See *map*.

trailer. A control structure that indicates the end of an electronic transmission.

transaction. A single business document, such as an invoice. See also *EDI transaction*.

transaction set. A group of standard data segments, in a predefined sequence, needed to provide all of the data required to define a complete transaction, such as an invoice or purchase order. See also *EDI transaction set*.

Transaction Store. The file that contains the results of translations and a history of translation activity.

transform. The process of converting a document from one format to another.

transient data queue (TD). A sequential data set used by the Folder Application Facility in MVS/CICS to log system messages.

translation. The process of converting a document from one format to another.

translation table. A user-defined table that translates data values that differ between trading partners. For example, if a manufacturer and supplier have different part numbers for the same item, each company can use its own part number and have it converted to the other company's part number during translation.

TSQ. See *temporary storage queue*.

U

UCS. Uniform Communication Standard.

unary operator. An operator that changes the sign of a numeric value.

UN/EDIFACT. United Nations Electronic Data Interchange for Administration Commerce and Transport.

Uniform Communication Standard (UCS). The EDI standard used in the grocery industry.

UN/TDI. United Nations Trade Data Interchange.

Usage. An association between a send or receive map and a trading partner.

V

validation table. A table, supplied by DataInterchange or defined by the user, which contains all acceptable values for a single data field.

variable. The entity in which a value may be stored based on data received; as opposed to a constant value.

W

WINS. Warehouse Information Network Standard.

Windows. Microsoft's graphical operating system under which DataInterchange Client runs.

X

X12. A common EDI standard approved by the American National Standards Institute.

XML. See Extensible Markup Language.



Index

Numerics

841 transaction set 309

A

Accumulators 262
 actions of 264
 adding to maps 263
 types of 263
Activity Log profiles 71
 creating 73
 purpose 71
 setup overview 72
Application Control key, setting 253
application data
 components 148
 fields 149
 loops 150
 records 149
 structures 149
 obtaining 146
 raw data records 147
 structuring 147
Application Defaults Profile dialog box 59
Application Defaults profiles 75
 creating 77
 setup overview 76
assignment 328
associated types 58

B

Between operator 291

BIN segment ID 309
binary segment receive processing 316

C

cascading windows 40
CICS Performance profiles 85
 creating 87
 purpose 85
 setup overview 86
client-server mode. *See* configuring DataInterchange Client
code lists 191
 creating 202
commands 329
 error 329
 mapto 329
 qualify / default 331
 setproperty 332
conditional commands 328
config database 8
configuration
 recommended 14
 multi-user, multi-server 14
configuration alternatives
 multi-user, client/server 11
 multi-user, stand alone 10
 single user, stand alone 9
configuring DataInterchange Client
 in client-server mode 6
 in stand-alone mode 7
configuring systems 11

- Contacts 141
 - adding 142
- Continuous Receive Profile dialog box 60
- Continuous Receive profiles 80
 - creating 82
 - setup overview 80
- control and data format 147
- control strings
 - compiling 235, 278
 - viewing compiled 236, 279
- Control Strings dialog box 68
- copying an item 35
- cryptographic services. *See* security services
- customization time data 9

D

- data elements 191
 - creating 199
- Data Format dialog box 64
- Data Format Dictionary dialog box 64
- Data Format Dictionary Editor
 - creating 156
 - importing a COBOL Copybook 157
- Data Format editors 154
 - accessing 154
 - Data Formats Editor 161
 - Dictionary Editor 156
 - Fields Editor 171
 - Loops Editor 164
 - navigating 173
 - Data Formats Editor paths 174
 - Dictionary Editor paths 173
 - Fields Editor paths 176
 - Loops Editor paths 175
 - Records Editor paths 175
 - Structures Editor paths 176
 - Record Editor 166
 - Record ID Information Editor 159
 - Structures Editor 169
- data format worksheets 151
- data formats 145
 - creating 146, 162
 - data types for 177
 - A (Alphabetic) 177
 - AC (Application control) 177
 - AN (Alphanumeric) 177
 - BN (Binary–unsigned) 178
 - Bn (Binary–unsigned) 177
 - CH (Character) 178
 - DT (Date) 178
 - FN (File name) 178
 - Hn (Hexadecimal) 178
 - HX (Hexadecimal) 178
 - ID (Identifier) 178
 - In (Integer–signed) 179
 - IT (Integer–signed) 179
 - Ln (Decimal–leading sign) 179
 - N (Numeric) 179
 - Nn (Numeric) 179
 - PD (Packed decimal) 179
 - Pn (Packed decimal) 180
 - R (Real) 180
 - Rn (Real) 180
 - TM (Time) 180
 - ZD (Zoned decimal) 180
 - Zn (Zoned decimal) 181
 - reusing components 176
- data transformation
 - Map Editor
 - starting 212
 - using 213
 - data transformation Map Editor 211
 - Map Command window pane 223
- data transformation maps
 - advanced techniques 228
 - creating 213
 - Details tab 219
 - editing 221
 - map rules
 - copying 234
 - creating 232
 - editing 234
 - specifying 231
 - viewing 232
 - qualification
 - editing an element qualification 228
 - specifying 224
 - types of 224
 - qualifying
 - repeating simple and compound elements 224
 - by expression 227
 - by multi-occurrence 226
 - by occurrence 225
 - by value 226
- data transformation translation tables 229
 - creating 230
- data types
 - supported when mapping elements 324
- DataInterchange Client
 - moving to 17
- DataInterchange Host, accessing 6
- deleting an item 37

Development system 12
 DI Client Release Migration 19
 DI variables
 DIAPPPFILE 266
 DIAPPTYPE 266
 DIAUTOCC 266
 DICCCTRL 266
 DICUSERDATA 267
 DIERRFILTER 267
 DIEXPTRACE 267
 DIMAPCHAIN 267
 DIMAPSWITCH 267
 DIPROLOG 267
 DISAPSEQ 268
 DIVALLEVEL 268
 DIVALTYPE 268
 DIVARTRACE 268
 dictionary 190
 Document Type Definition (DTD) 183
 DTD Editor 186

E

EDI standard dictionary
 creating a 193
 EDI Standard Dictionary dialog box 65
 EDI standard transaction
 creating a 195
 EDI Standard Transaction dialog box 65
 EDI standards 189
 download 16
 installing 16
 EDI standards editors 191
 accessing 192
 Code List Editor 202
 Data Elements Editor 199
 Dictionary Editor 193
 Envelope Standard Editor 203
 Segments Editor 197
 Transactions Editor 195
 EDI systems 12
 editing an item 36
 editor window tool bar 33
 editor windows 27
 grids 38
 electronic format identification (EFI) segment 310
 envelope control string
 compile 206
 Envelope Control String list window 206
 Envelope profile
 ICS (I) 92
 UCS (U) 93

UN/EDIFACT (E) 92
 UN/TDI (T) 93
 X12 (X) 93
 Envelope profiles 89
 creating 91
 setup overview 89
 envelope standard
 editing 203
 Envelope Standard dialog box 65
 envelopes 190
 event logs 307
 viewing 307
 examples
 PC executable files 314
 sending to limited systems 314
 variable length PC files 314
 export/import 51
 file specifications 52
 exporting 52
 to a file 52
 to other systems 54
 expressions 325, 372
 Extensible Markup Language (XML) 183

F

fields 149
 creating 172
 file management 35
 fixed-to-fixed
 map, create 280
 translation 280
 format specifications 312
 functions 332
 char 332
 concat 332
 created 333
 date 334
 datecnv 334
 find 334
 found 335
 getproperty 335
 hexdecode 336
 hexencode 336
 isempty 336
 left 338
 length 338
 lower 338
 number 338
 numformat 340
 occurrence 340
 overlay 341

- right 342
- round 342
- strcomp 343
- strcompl 343
- strcompn 344
- strcompni 344
- substring 345
- time 346
- translate 346
- trimleft 347
- trimright 348
- truncate 348
- upper 349
- validate 349

G

- generic receive usages 278
 - defining 278
- generic rules (usages) 135
- generic send usages 277
 - defining 277
- global accumulator 263
- global variables 236

H

- help 44
 - accessing 45
 - using buttons 45
 - using F1 key 45
 - using menu 45
 - links 46
 - screen buttons 46
 - using 45
- help menu commands 44
- hierarchical loops 279
 - mapping HL segment 279
 - preparing 400
- HL segment 399
 - literal keywords for 404
- host functions
 - unavailable in DI Client Interface 21

I

- importing 56
 - a DTD file 57
- Install Wizard 3
- installing DataInterchange Client 3
 - hardware requirements 4
 - procedure 4

Setup Types

- Compact 5
- Custom 5
- Typical 5
- installing standards 16
- Is Empty operator 291
- Is Not Empty operator 291

K

- keywords 323

L

- Language profiles 95
 - setup overview 96
- Like operator 291
- list window tool bar 32
- list windows 24
 - modifying information in 24
- literal keywords 359
 - mapping techniques 382
- literals 264, 322
 - accumulator 359
 - adding to a map 265
 - and data types 265
 - conditional processing 359
 - control data 392
 - using for receive mapping 358
- loops 150
 - creating 164

M

- Mailbox dialog box 58
- Mailbox profiles 99
 - creating 101
 - setup overview 99
- map rules
 - copying 234
 - creating 232
 - editing 234
 - specifying 231
 - viewing 232
- map variables 321
- mapping
 - advanced techniques 228, 266
 - data transformation maps vs. send and receive maps 209
 - generic service segment fields (receive) 393
 - getting started 207
 - service segment fields (send) 395

- specific service segment fields (receive) 392
- mapping commands 264
 - adding to a map 265
- Mapping Data Element Editor 249
 - applying advanced mapping capabilities
 - with 223, 250
 - Repeat button 252
 - Special Handling button 252
- Mapping dialog box 66
- maps
 - creating 207
 - migration 281
- Message Log 47
 - viewing messages in 47
- middleware 6
- Migrating a Map 281
- migrating a map 237
 - Client Migration Option 281
- Migrating Data Between Versions and Releases 19
- minimal trading partners 133, 231, 272
- moving items to DataInterchange Client
 - control strings 18
 - in client-server mode 18
 - in stand-alone mode 18
 - setup profiles and trading partner profiles 17
 - in client-server mode 17
 - in stand-alone mode 18
- MQSeries Queue profiles 103
 - creating 105
 - setup overview 104
- MQSeries, using with Continuous Receive 82
- multiple systems 39
- multi-user database 13

N

- naming convention changes 21
- navigator bar 31
- Network Commands profiles 113
 - creating 115
 - setup overview 114
- Network Profile dialog box 59
- Network profiles 108
 - creating 110
 - setup overview 109
- Network Security profiles 117
 - creating 120
 - setup overview 118
- nickname 274

O

- ODBC link 6

- operators
 - arithmetic 327
 - comparison 326
 - logical 326
 - unary 327
- order of precedence 327

P

- printing an item 37
- processes and rules 132
- properties
 - EDI standard envelope generic 350
 - EDI standard envelope specific 351
 - target document 350

Q

- qualification
 - editing a data element 261
 - editing a loop or segment 258
 - editing an element qualification 228
 - specifying 224, 255
 - types of 224, 255
- qualifying
 - data elements 259
 - by value 259
 - loops and segments 255
 - by occurrence 255
 - by path 257
 - by value 257
 - repeating simple and compound elements 224
 - by expression 227
 - by multi-occurrence 226
 - by occurrence 225
 - by value 226
- qualifying a repeating data element or composite data element
 - by path 262
- qualifying a repeating data element or composite data element by occurrence 261
- queries 285
 - changing default 24
 - copying 292
 - creating 287
 - deleting 292
 - editing 291
 - purpose of 285
 - running 292
 - setup overview 286
 - using filters in 289

R

- receive
 - processing for binary segment 316
- Record ID Information profile
 - creating 159
- records 149
 - creating 167
- referenced types 58
- references
 - forward and reverse 324
- relation operators, selected 291
- Release Migration 19
- renaming an item 36
- reports 295
 - creating 296
 - deleting 298
 - editing 297
 - previewing 298
 - printing 298
 - setup overview 295
- requirements 4
- running DI Client 16
- runtime data 9

S

- Security Profile dialog box 62
- security services
 - authentication 117
 - compression 118
 - encryption 117
 - filtering 118
- segment
 - creating a 198
- segment ID
 - BIN 309
 - BIN length 310
- segments 190
 - binary 311
 - receive processing 316
 - send processing 311
- selecting commands 28
 - using menus 28
 - using tool bars 31
- send and receive maps
 - advanced techniques 266
 - control strings
 - compiling 278
 - viewing compiled 279
 - creating 241
 - editing 248
 - Map Editor 239

- starting 240
 - using 241
- qualification
 - editing a data element 261
 - editing a loop or segment 258
 - specifying 255
 - types of 255
- qualifying
 - data elements 259
 - by value 259
 - loops and segments 255
 - by occurrence 255
 - by path 257
 - by value 257
- qualifying a repeating data element or composite data element
 - by path 262
- qualifying a repeating data element or composite data element by occurrence 261
- translation tables 269
 - creating 270, 271
- sending
 - PC executable files 314
 - PC variable length files 314
 - to systems with file limitations 314
- setting preferences 41
 - customizing field tags 43
 - message log 41
 - selecting tree view 42
 - system color 42
 - viewing control and status bars 43
 - windows 41
- shortcuts, keyboard and mouse 34
- specifying a path 323
- stand-alone mode. *See* configuring DataInterchange Client
- structures 149
 - creating 170
- system database 9
- systems
 - multiple 39
 - selecting 39

T

- tiling windows 40
- Trading Partner Profile dialog box 62
- Trading Partner profiles 127
 - creating 130
 - purpose 127
 - setup overview 128

- trading partners 127
 - copying usages of 276
 - creating usages for 137, 273
 - data transformation map rule 140
 - editing usages of 276
 - minimal concept 273
 - receive map usage 139
 - send map usage 138
 - specifying usages and rules 136
 - specifying usages of 272
 - viewing usages of 136, 232, 273
- transaction accumulator 263
- transaction sets 190
- Transaction Store 301
 - queries 303
 - reports 306
 - default Interchange 306
 - setup overview 302
- transactions 190
 - BIN segment length 310
 - electronic format identification (EFI) 310
 - parts of
 - code lists 191
 - data elements 191
 - dictionary 190
 - segments 190
 - transaction sets 190
 - set 841 309
- translating data
 - setup 49
- translation tables 229, 269
 - creating 230, 270, 271
 - forward 270
 - reverse 270
- translator hierarchy 136
- tree windows 26

U

- User Exits profiles 121
 - creating 123
 - setup overview 122
- using
 - 841 transaction set 309

V

- validation 396
- variable
 - loop 322

- variables
 - global 322
 - local 321
 - naming 322
 - special 322
- viewing an item 35

W

- window title bars 40
- window types 24
- windows
 - cascading 40
 - closing 27
 - tiling 40

X

- XML Dictionary
 - creating 185
- XML Dictionary Editor 185
- XML editors
 - accessing 184

