BEA eLink Data Integration Option

User Guide

BEA eLink Data Integration Option 1.3
Document Edition 1.3
January 2000

Copyright

Restricted Rights Legend

Trademarks or Service Marks

BEA eLink Data Integration Option User Guide

Document Edition	Part Number	Date	Software Version
1.0	870-001001-001	May 1999	BEA eLink Data Integration Option 1.0
1.1	870-001001-001	July 1999	BEA eLink Data Integration Option 1.1
1.2	870-001001-002	October 1999	BEA eLink Data Integration Option 1.2
1.3	870-001001-003	January 2000	BEA eLink Data Integration Option 1.3

Contents

3. Installing BEA eLink DIO

4. Configuring the Environment for eLink DIO

5. Data Mapping Conversion

About This Document

BEA eLink Data Integration Option (hereafter referenced as eLink DIO) runs with the BEA eLink Platform product as well as with other BEA products. This document covers the following topics:

- *Understanding eLink and the Data Integration Option* provides an overview of the eLink DIO product and data mapping process and introduces the eLink DIO components.
- *Sample Scenarios for the Data Integration Option* explains how eLink DIO fits into the BEA eLink Platform environment.
- *Installing BEA eLink DIO* explains how to install the eLink DIO product.
- *Configuring the Environment for eLink DIO* provides information for configuring BEA eLink Platform and eLink DIO to handle service requests for data mapping.
- *Data Mapping Conversion* provides information about using the eLink DIO component to define the rules for data mapping. It includes information for developing the rules for data mapping and executing a TUXEDO service request for data mapping.
- *Error and Informational Messages* describes error and informational messages as well as actions to resolve the errors.
- *Data Mapping Worksheet* is a planning worksheet for the data mapping installations, configurations and tasks.

What You Need to Know

This document is intended for system administrators who will configure and administer eLink DIO as well as programmers who will map FML buffers to non-FML buffer types. This guide assumes knowledge of BEA eLink Platform and TSI Mercator products. This guide also assumes knowledge of the C programming language. In addition, programmers will find useful pointers for developing client programs and service routines that send data through the eLink DIO.

e-docs Web Site

BEA product documentation is available on the BEA corporate Web site. From the BEA Home page, click on Product Documentation or go directly to the "e-docs" Product Documentation page at http://e-docs.beasys.com.

How to Print the Document

You can print a copy of this document from a Web browser, one file at a time, by using the File—>Print option on your Web browser.

A PDF version of this document is available on the eLink Data Integration Option documentation Home page on the e-docs Web site (and also on the documentation CD). You can open the PDF in Adobe Acrobat Reader and print the entire document (or a portion of it) in book format. To access the PDFs, open the eLink Data Integration Option documentation Home page, click the PDF files button and select the document you want to print.

If you do not have the Adobe Acrobat Reader, you can get it for free from the Adobe Web site at http://www.adobe.com/.

Related Information

The following BEA publications are also available:

- *TUXEDO System 6 Reference Manual*
- *TUXEDO System 6 Programmer's Guide, Volumes 1 and 2*

For more information about data mapping, refer to the Mercator documentation set. In particular, refer to the following documents when using the eLink DIO component.

- *Mercator Type Editor Reference Guide*
- *Mercator Map Editor Reference Guide*
- *Mercator Functions & Expressions Reference Guide*

Contact Us!

Your feedback on the BEA eLink Data Integration Option documentation is important to us. Send us e-mail at **docsupport@beasys.com** if you have questions or comments. Your comments will be reviewed directly by the BEA professionals who create and update the eLink Data Integration Option documentation.

In your e-mail message, please indicate that you are using the documentation for the BEA eLink Data Integration Option 1.3 release.

If you have any questions about this version of BEA eLink Data Integration Option, or if you have problems installing and running BEA eLink Data Integration Option, contact BEA Customer Support through BEA WebSupport at **www.beasys.com**. You can also contact Customer Support by using the contact information provided on the Customer Support Card, which is included in the product package.

When contacting Customer Support, be prepared to provide the following information:

- Your name, e-mail address, phone number, and fax number
- Your company name and company address

- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

Documentation Conventions

The following documentation conventions are used throughout this document:

Item	Examples
Variable names	Variable names represent information you must supply or output information that can change; they are intended to be replaced by actual names. Variable names are displayed in italics and can include hyphens but not underscores. The following are examples of variable names in text: *error-file-name* The *when-return* value...

User input and screen output	For screen displays and other examples of input and output, user input appears as in the first of the following lines; system output appears as in the second through fourth lines:
	`dir c:\accounting\data`
	`Volume in drive C is WIN_NT_1`
	`Volume Serial Number is 1234-5678`
	`Directory of C:\BEADIR\DATA`

Item	Examples
Syntax	Code samples can include the following elements: ■ Variable names can include hyphens but not underscores (e.g., *error-file-name)* ■ Optional items are enclosed in square brackets: [ ]. If you include an optional item, do not code the square brackets. ■ A required element for which alternatives exist is enclosed in braces {}. The alternatives are separated by the pipe (vertical bar) character: |. You must include only one of the alternatives for that element. Do not code the braces or pipe character. ■ An ellipsis (...) indicates that the preceding element can be repeated as necessary.
Omitted code	An ellipsis (...) is used in examples to indicate that code that is not pertinent to the discussion is omitted. The ellipsis can be horizontal or vertical.
Environment variables	Environment variables are formatted in an uppercase font. ENVFILE=${APPDIR}
Key names	Key names are presented in boldface type. Press **Enter** to continue.
Literals	Literals are formatted in a monospace font. `class extendSample`
Window items	Window items are presented in boldface type. Window items can be window titles, button labels, text edit box names or other parts of the window. Type your password in the **Logon window**. Select **Export** to make the service available to the client.

CHAPTER

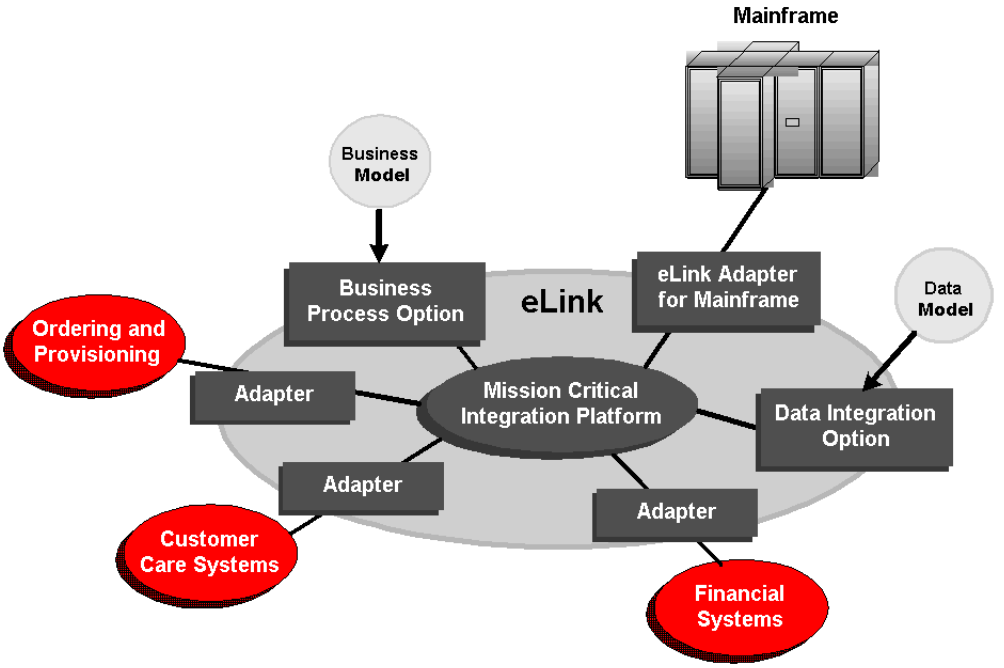
1 Understanding eLink and the Data Integration Option

BEA eLink provides an open Enterprise Application Integration (EAI) solution that allows applications throughout organizations to communicate seamlessly. Using EAI, you gain the long-term flexibility and investment protection you need to keep up with today's ever-changing business environment.

Typically, companies use packaged applications to automate internal operations, such as financial, manufacturing, human resources, etc. While they successfully address the needs of these specific areas, these proprietary platforms often do not work together. To compete today, you need a much greater exchange of information. Systems need to communicate at a process level within your own organization, as well as with customer's and supplier's systems. BEA eLink Platform is the underlying basis of BEA eLink, a family of off-the-shelf enterprise application integration (EAI) products that leverage BEA's transaction platform to integrate existing legacy applications with customer-focused and business-to-business e-commerce initiatives.

BEA eLink Platform provides a proven infrastructure for integrating applications within the enterprise and across the Web. BEA eLink Platform ensures high-performance, secure transactions and transparent access to mission-critical applications and information throughout the enterprise and across the Web. Figure 1-1 illustrates the eLink logical architecture and shows where the eLink Adapters fit into the process.

Figure 1-1 BEA eLink Solution Illustration



The entire BEA eLink family (including all options and adapters) is highly scalable. Multiple instances of BEA eLink components can collaborate so that work is divided between eLink domains. BEA eLink includes SNMP integration for enterprise management.

The current BEA eLink Platform leverages the BEA TUXEDO infrastructure because it is based on a service-oriented architecture. Both BEA TUXEDO and BEA eLink communicate directly with each other and with other applications through the use of **services**. Multiple services are grouped into “application servers” or “servers”. The terms, TUXEDO services/ servers and eLink services/servers can be used interchangeably.

The BEA eLink Data Integration Option uses the eLink Platform product for processing service requests for data mapping and data conversion. The following products are part of the eLink DIO data conversion solution:

- BEA eLink Data Integration Option Design Client

The eLink DIO Design Client is used for the development phase, which includes defining the data mappings between the FML and non-FML buffers. The Design Client is supported on Windows NT only. The Design Client consists of the following components:

- TSI Mercator Authoring System
- BEA eLink FML Importer
- TSI Mercator Online Documentation
- BEA eLink Data Integration Option Online Documentation

- BEA eLink Data Integration Option Server

  The eLink DIO Server is the component used for execution of the data translation and will run on Windows NT and supported UNIX platforms (AIX, HP-UX, and Solaris.) The eLink DIO Server consists of the following components:

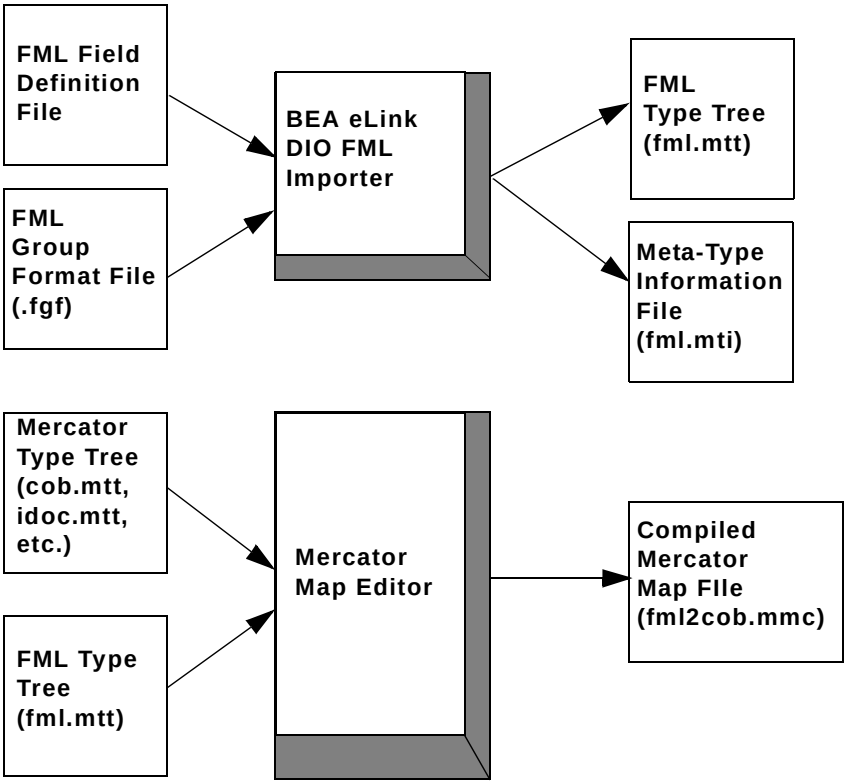
  - BEA eLink Executable Server (ELINKDIO)
  - BEA eLink Data Integration Option Online Documentation

Overview of the eLink DIO Data Mapping Process

The data mapping process includes two phases: the development phase and the execution phase. The development phase uses the eLink DIO Design Client and sets up the environment and mapping rules for data conversion.

Figure 1-2 describes the data mapping development process flow. Simply put, the process flow begins with an input file or an FML Field Definition File and an FML Group Format file (`.fgf`) and creates a compiled Mercator map file (`.mmc`).

Figure 1-2 Data Map Development



Development Phase

The development phase uses the eLink DIO Design Client and sets up the environment and mapping rules for data conversion. This phase is separated into two parts, the creation of Mercator type tree files and the creation of the Mercator map file. You must have a source type tree and a destination type tree. Use a supported Mercator Importer and the eLink DIO FML Importer to create the type tree files. Before creating the type tree files, you must describe your input data as well as the format of your output data. The eLink DIO FML Importer in the eLink DIO Design Client also creates a Meta-Type Information (`.mti`) file which the TUXEDO `ELINKDIO` server uses during

execution. After both the non-FML and the FML type trees are created, a Mercator map file (`.mmc`) is created and compiled using the type tree files. The eLink DIO Server during the execution phase issues the TUXEDO service request for data mapping. It is during execution that the actual mapping occurs.

Using Importers to Create Mercator Type Tree Files

Use a supported Mercator Importer to input a type of file and create a type tree file as output. For more information on how to use the Mercator Importers, refer to the Mercator documentation. Use the eLink DIO FML Importer to create an FML type tree.

You *must* input into the eLink DIO FML Importer an FML Field Definition file and an FML Group Format (`.fgf`) file. To define these groups, refer to "Data Mapping Conversion." After defining these groups in the FML Group Format file, use the eLink DIO FML Importer to create a Meta-Type Information file (`.mti`) and create an output FML type tree file (`.mtt`).

The eLink DIO reads the FML data to convert it to the proprietary format. The Meta-Type Information (`.mti`) file determines which FML fields will be retrieved and in what order.

Compiling Type Tree Files into Mercator Map Files

Mercator map files define adapters, source and destination type trees, and transformation rules. One map must be created for each message that is to be mapped. One map is needed to convert the FML buffer to a non-FML buffer type, and another map is needed for the conversion from the non-FML buffer type back to FML for the response.

Execution Phase

Prior to a TUXEDO service request executing a Mercator map file, be sure that you have specified the ELINKDIO server and the eLink DIO configuration file in the UBBCONFIG file. Also define the services for data mapping in the eLink DIO configuration file.

The execution phase requires that certain files exist in the execution environment. After moving the files to this environment, ELINKDIO server can process a service request for data mapping.

Setting Up the Execution Environment

Certain files must be transferred from the development environment on Windows NT to the execution environment, which is also where APPDIR is pointing. If your execution environment is HP-UX, SUN Solaris, or AIX, use the Mercator **Map>Port** functionality to convert the Mercator map files (`*.mmc`) to a format that is suitable for these UNIX platforms.

Note: If your execution environment is Windows NT, you do not need to perform the **Map>Port** step.

After performing the Port process for the Mercator map files, transfer the required files to the execution environment. When transferring these files from the development environment on Windows NT to the execution environment, such as HP-UX, AIX, or SUN Solaris, specify the appropriate mode (ASCII or binary) to use during the transfer.

- *`fml2nonf`*`.mti` (ASCII file)

  Note: This file is a copy of the *`sample`*`.mti` renamed to match the name of the *`fml2cob`*`.mmc` map file.

- *`nonf2fml`*`.mti` (ASCII file)

  Note: This file is a copy of the *`sample`*`.mti` renamed to match the name of the *`cob2fml`*`.mmc` map file.

- *`sample`*`.fml` (ASCII file)
- *`fml2nonf`*`.mmc` (binary file)
- *`nonf2fml`*`.mmc` (binary file)

Initiating a Service Request for Data Mapping

After the execution environment is set up, you can issue a TUXEDO service request that will implement the data mapping rules defined in the map files. Before issuing the TUXEDO service request, be sure the following tasks are complete.

1. Edit the TUXEDO UBBCONFIG file to define the ELINKDIO server.
2. Transfer the required files to the execution environment.
3. Add `sample.fml` to FIELDTBLS and FIELDTBL32. Also add the path for `sample.fml` to FLDTBLDIR and FLDTBLDIR32.
4. Ensure that TUXDIR and APPDIR are appropriately set.
5. Initiate the eLink Platform software.
6. Issue the eLink service request.

Note: For information on the eLink Platform software, refer to the eLink Platform documentation.

Requirements for Data Mapping

The following software must be used to map data between an FML buffer and a non-FML data format.

- eLink DIO Design Client
- eLink DIO Server

eLink DIO Design Client

The eLink DIO Design Client provides a data mapping development environment that runs on Windows NT. For more information about the development environment for data mapping and translations, refer to the "Overview of the eLink DIO Data Mapping Process."

TSI Mercator Authoring System Components

The following sections describe the different Mercator Authoring System components necessary for defining the data mapping rules in the development environment.

Importers

Use the Mercator Importers to create type tree files. For example, to convert an FML buffer to a COBOL Copybook, use the COBOL Copybook Importer to create a COBOL type tree. For information about using Importers, refer to the Mercator documentation. BEA eLink DIO supports the following Mercator Importers:

- COBOL Copybook Importer

  Note: The COBOL Copybook Importer is supported with BEA eLink for Mainframe products only.

- SAP R/3 ALE IDOC Importer

Type Editor

Use the Mercator Type Editor to make any modifications to the type tree files, such as adding the Control information to the header. The Type Editor also provides an Export feature that will produce a type tree file in a suitable format for different platforms. For information about using the Type Editor, refer to the *Mercator Type Editor Reference Guide.*

Map Editor

Mercator uses user-defined files called maps to perform data translations. An eLink DIO service performs the data translation. The eLink DIO configuration file defines the services and the maps that eLink DIO uses for the data translations.

Each map has at least one input card and one output card. The service passes the input data to input card 1 and returns data back to the service-requester via output card 1. There may be any number of additional input or output cards (for example, to read from a file or create a file). Output card 1 has a header with control information set in the map. This card determines the destination and format of the output buffer.

Note: The header information is *required* in the output card. For information about defining Control information with /Q, refer to the "Control Information for TUXEDO /Q."

Use the Mercator Map Editor to define the mapping rules and compile a map file. For information about creating a Mercator map file, refer to the *Mercator Map Editor Reference Guide*.

Type Tree Maker

Use the Mercator Type Tree Maker to build your own Importer. Creating your own Importer provides conversion to user-defined formats. For information about using the Type Tree Maker, refer to the *Mercator Type Tree Maker Reference Guide*.

Unsupported Mercator Components

The following Mercator software components are not supported with BEA eLink DIO:

- Type Tree Collections (ANSI, X12, EDIFACT, HL7)
- Tree Installer
- Application and Database adapters
- System Editor

BEA eLink DIO FML Importer

The eLink DIO FML Importer reads the FML Field Definition file and an FML Group Format file to create a type tree file. This component reads the FML Field Definition file and creates a Type Tree Maker Command file (`.mts`) which is then run through the type tree maker. The eLink DIO FML Importer also creates a Meta-Type Information (`.mti`) file which the TUXEDO `ELINKDIO` server uses during execution.

eLink DIO Server

The eLink DIO Server is the component used for execution of the data translation and will run on Windows NT and supported UNIX platforms (AIX, HP-UX, and Solaris.) For more information about executing data translations, refer to the "Overview of the eLink DIO Data Mapping Process."

ELINKDIO Server

The ELINKDIO server is an executable program that runs on HP-UX, SUN Solaris, AIX or Windows NT. It runs in an eLink Platform environment performing the following functions:

- Specifies the eLink DIO configuration file by defining the -c option in the CLOPT parameter in the SERVERS section of the UBBCONFIG file. The eLink DIO configuration file advertises the maps which are advertised as TUXEDO services.
- Maintains other command line information for execution of the Mercator Mapping Library.
- Executes TuxRunMap library which calls the Mercator Mapping Library to perform data mapping.
- Receives the FML buffer and converts it into an internal format for use by Mercator.

2 Sample Scenarios for the Data Integration Option

The BEA eLink Data Integration Option uses the eLink Platform product for processing service requests for data mapping and data conversion.

A TUXEDO translation service takes an input buffer, performs a data translation on the buffer, and returns the translated data in an output buffer, exposing the Mercator wrapper for TUXEDO as a service. It also can write output data to queues or files.

For example, to call a map called TOEDI.MMC that produces EDI data in an FML format, the service name would be TOEDI.

Each map has at least one input card and one output card. The service passes the input data to input card 1 and returns data back to the service-requester via output card 1. There may be any number of additional input or output cards (to read from a file or create a file, for example). Output card 1 has a "header" with control information set in the map. This card determines the destination and format of the output buffer. *This header is required*. See the section entitled "Control Information for TUXEDO /Q" in this document for more details.

This document will describe the following topics:

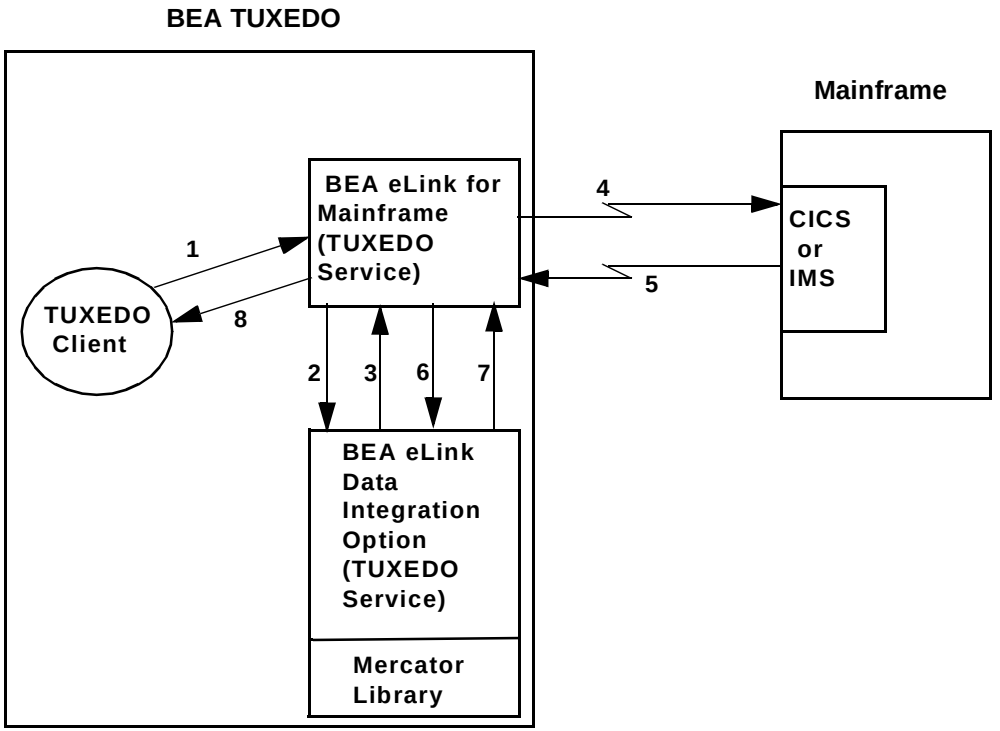
- Using Data Mapping with a Mainframe Environment
- Using Data Mapping with Multiple Destinations
- Using Data Mapping with Other eLink Adapters

Using Data Mapping with a Mainframe Environment

Figure 2-1 shows how the products and components work together to convert buffers between an FML and a non-FML format and transfer the buffers between a UNIX or Windows NT environment and a mainframe environment.

The BEA eLink for Mainframe gateway asynchronously calls external TUXEDO services, a feature which will be used to call the ELINKDIO server. The ELINKDIO server is a standard TUXEDO MSSQ server that receives a TUXEDO buffer as input, invokes the Mercator data mapping engine, and returns a TUXEDO buffer as output.

Figure 2-1 Data Mapping Scenario of FML to COBOL Copybook to FML



The following description explains the process flow in Figure 2-1 of a TUXEDO client making a request to send an FML buffer from a UNIX environment to a CICS region on a mainframe and return an FML buffer to the TUXEDO client.

1. A TUXEDO client issues a request to send an FML buffer to the mainframe.
2. The request passes the FML buffer to the BEA eLink for Mainframe gateway. The service name in the request causes the gateway to pass the FML buffer to the eLink DIO component to convert the FML buffer to a COBOL copybook format.
3. The eLink DIO component converts the FML buffer to a COBOL copybook and passes it back to the BEA eLink for Mainframe gateway.

Note: For conversions to COBOL copybook, the COBOL type tree does not contain the eLink DIO header, so you must edit the type tree using the Type Editor. You must add the eLink DIO control header from the FML type tree to output COBOL copybooks.

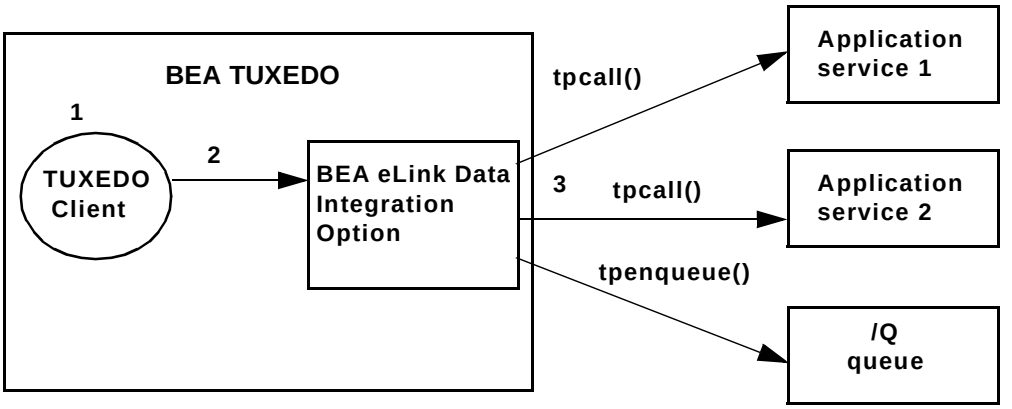
4. The BEA eLink for Mainframe gateway passes the converted COBOL copybook to the CICS or IMS region on the mainframe.
5. After the requested transaction is complete, the mainframe returns a response by sending the COBOL copybook back to the BEA eLink for Mainframe gateway.
6. The BEA eLink for Mainframe gateway passes the copybook to the eLink DIO for conversion back to an FML buffer.
7. After the copybook is converted to an FML buffer, it is passed back to the BEA eLink for Mainframe gateway.
8. The BEA eLink for Mainframe gateway then passes the FML buffer response to the TUXEDO client.

Using Data Mapping with Multiple Destinations

The data mapping process uses a TUXEDO environment that includes the eLink Platform and the BEA eLink Data Integration Option products. Figure 2-2 shows how the products work together to convert buffers between an FML and a non-FML format.

The ELINKDIO server is a standard TUXEDO MSSQ server that receives a TUXEDO buffer as input, invokes the Mercator data mapping engine, converts the buffer to another format and then passes it to multiple destinations.

Figure 2-2 Converting FML to New Format and Passing to Multiple Destinations



The following description explains the process flow in Figure 2-2 of a TUXEDO client making a request to send an FML buffer to multiple applications.

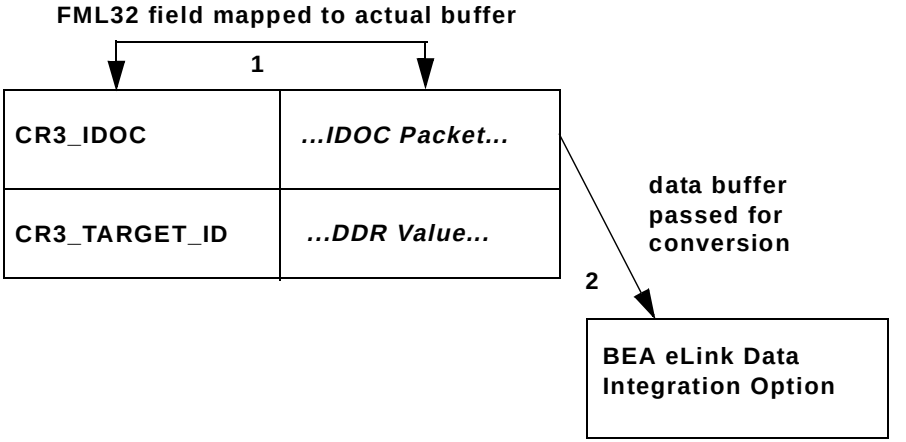
1. A TUXEDO client issues a request to send an FML buffer to multiple applications.
2. The request passes the FML buffer to the eLink Data Integration Option to convert the FML buffer to another buffer format.
3. The eLink Data Integration Option converts the FML buffer to the new format and passes it to the appropriate applications.

Note: The mapping to multiple destinations is done through Mercator input and output cards. For information on defining input and output cards, refer to the Mercator documentation.

FML32 Value Buffers

A particular use of the eLink Data Integration Option component allows for processing of FML32 Value buffers. Figure 2-3 shows how to map a Mercator tag to an FML32 field that then points to the actual data buffer. Once the actual data buffer is located, only it is passed to the eLink Data Integration Option application for conversion.

Figure 2-3 Passing an FML32 Value Buffer



1. The FML32 name contains the actual data buffer to be passed to the eLink DIO component.

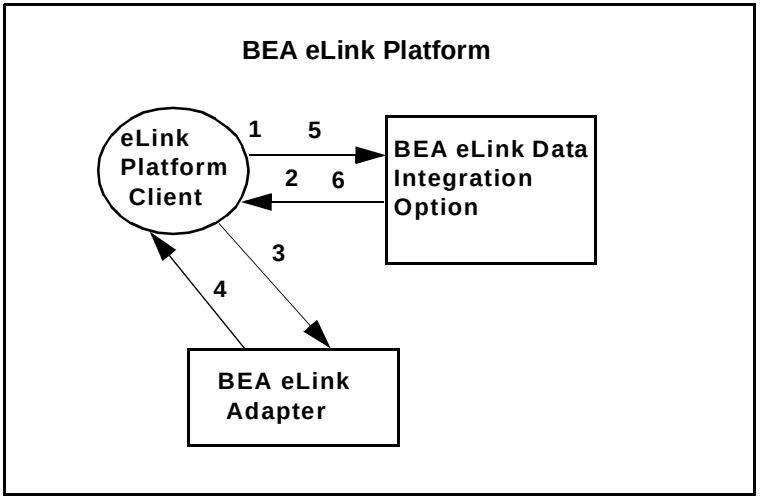
   Note: The FML32_VAL_FLD specifies the name of the FML32 field in the input buffer which contains data for processing. This parameter replaces the MERCATOR_FV_IN field used in previous releases of eLink DIO.

2. The data is extracted from the FML32 field and sent directly to Mercator for mapping.

Using Data Mapping with Other eLink Adapters

The data mapping process uses an eLink Platform environment that includes the eLink Platform and the BEA eLink Data Integration Option products as well as other eLink Adapters. For example, Figure 2-4 shows how the products and components work together to convert buffers between an FML and a non-FML format and then pass a request to an eLink Adapter.

Figure 2-4 Data Mapping Scenario of FML with eLink DIOs



The following description explains the process flow for invoking eLink DIO directly from the client to transform data prior to sending it to an eLink Adapter. The same process could be used after receiving data from an eLink Adapter to transform it back to an eLink Platform format.

1. An eLink Platform client calls eLink DIO to transform data to a format usable by an eLink Adapter.
2. The eLink DIO software sends the transformed data back to eLink Platform.
3. BEA eLink Platform calls the eLink Adapter to process the data.

4. After the eLink Adapter processes the data, it passes the response back to eLink Platform.
5. The resulting data from the eLink Adapter is passed to the eLink DIO to be transformed into an eLink Platform-compatible format, such as FML.
6. The eLink DIO transforms the data to an eLink Platform-compatible format, such as FML, and passes it back to eLink Platform.

3 Installing BEA eLink DIO

This chapter consists of the following topics:

Installation Prerequisites

Refer to the *BEA eLink Data Integration Option Release Notes* for information on prerequisite software that must be installed and operational prior to installing the BEA eLink DIO software. Also, for planning purposes, refer to the "Data Mapping Worksheet" section.

The following products must be installed and operational during the development and execution of a data mapping service request.

BEA eLink DIO Design Client

Prior to installing the eLink DIO Design Client, install the following prerequisite software in the following order.

1. TSI Mercator Open Edition Client 1.4.2

   Note: Use the SERIAL number from the eLink DIO license key when prompted for a Serial number during the Mercator Open Edition Client installation.

   The TSI Mercator Open Edition Client is available on the eLink DIO Design Client CD-ROM in the `client/tsi_Clnt` directory. Select the `Setup.exe` program to install the software.

2. TSI Mercator Open Edition Client 1.4.2 Service Pack 8

   You may apply service pack 8 to the TSI Mercator Open Edition Client. This service pack is available on the eLink DIO Design Client CD-ROM in the `client/tsiSpck8` directory. Select the `Setup.exe` program to install the service pack.

Unsupported Mercator Features

The following Mercator software components are not supported with BEA eLink DIO:

- Type Tree Collections (ANSI, X12, EDIFACT, HL7)
- Tree Installer
- Application and Database adapters
- System Editor

BEA eLink DIO Server

Prior to installing the eLink DIO Server, install eLink Platform 1.1.

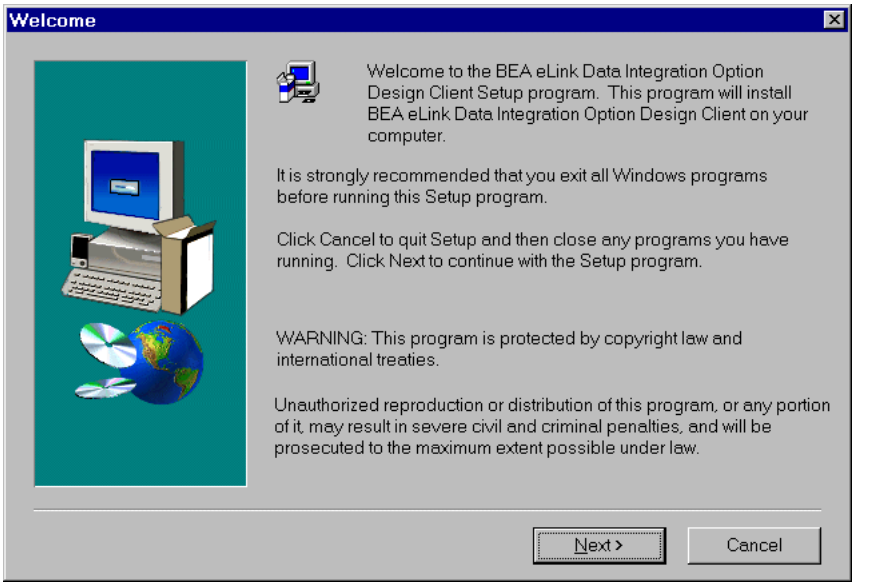
Installing the eLink DIO Design Client (Windows NT only)

Perform the following steps to install the eLink DIO Design Client software on a Windows NT system.

Note: TSI Mercator Authoring System must be installed prior to installing the eLink DIO Design Client software for your development environment.

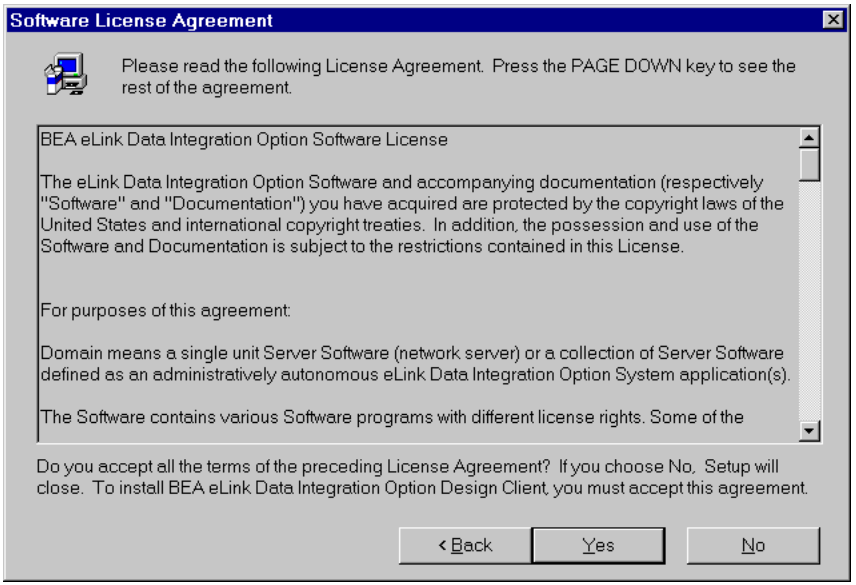
1. Insert the product CD-ROM and click on the **Run** option from the **Start menu**. The **Run** window displays. Click on the **Browse** button to select the CD-ROM drive. Select the `client/eLnkClnt` directory and select the `Setup.exe` program. Click **OK** to run the executable and begin the installation. The following **Welcome** screen displays. Click **Next** to continue with the installation.

Figure 3-1 Welcome



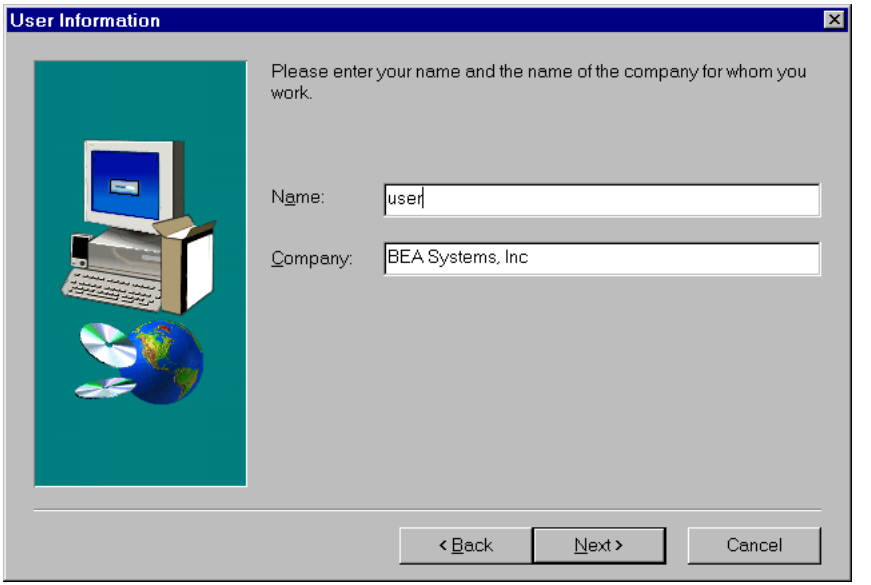
2. The **Software License Agreement** screen displays after the **Welcome** screen. Click **Yes** to accept the terms of the agreement and continue with the installation. Click **No** to abort the installation process.

Figure 3-2 User Information



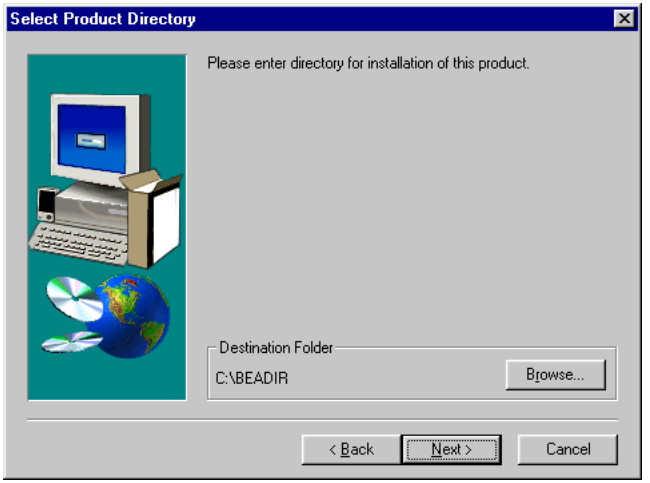
3. The **User Information** screen displays after the **Software License Agreement** screen. Enter your name in the **Name** field. Enter the name of your company in the **Company** field. Click **Next** to continue with the installation.

Figure 3-3 User Information



4. The **Select Product Directory** screen displays after the **User Information** screen. Click the **Browse** button and select a new destination folder if you do not want to install to the default destination folder. Click **Next** to continue the installation.

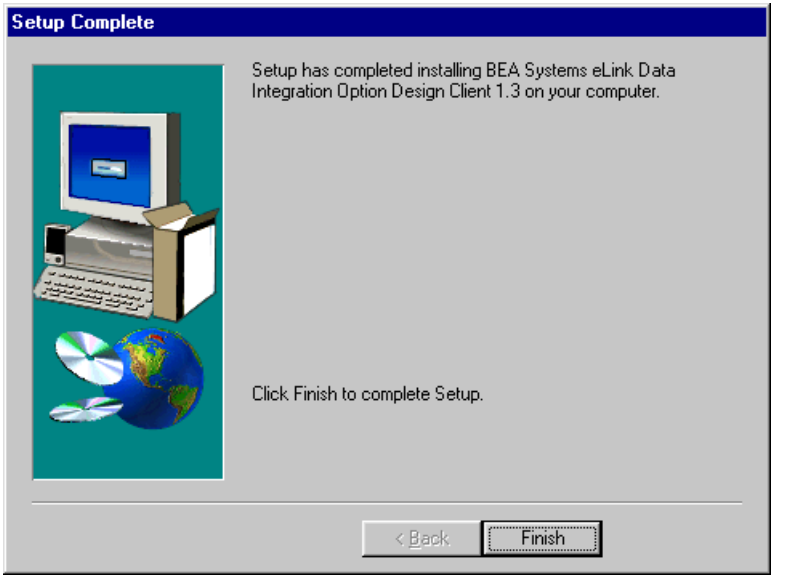
Figure 3-4 Select Product Directory



The installation begins and a progress bar displays the status. The eLink DIO Design Client components install into the specified directory.

5. The **Setup Complete** screen notifies you that the eLink DIO Design Client software is installed on your system. Click **Finish** to complete the setup process.

Figure 3-5 Setup Complete



Distribution Libraries and Executables for eLink DIO Design Client

The eLink DIO Design Client CD-ROM contains the following libraries and executable programs. After installing the eLink DIO Design Client software, verify that these libraries and programs are installed on your system.

Windows NT 4.0

Verify that the following files are installed by the eLink DIO software:

Table 3-1 Windows NT Installed Files for Design Client

Directory	Files
/elink	elinkimprt.exe
	mti2fgf.exe
	libfml32.dll
	libgp.dll

Installing the eLink DIO Server on the Execution Platform

This section explains how to install the eLink DIO Server software on the following execution platforms:

- Microsoft Windows NT
- HP-UX
- SUN Solaris
- IBM AIX

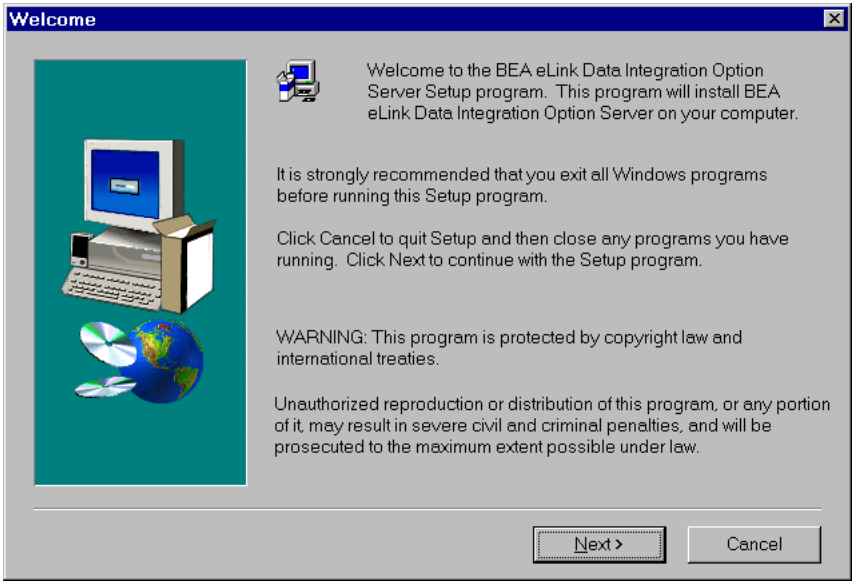
Note: BEA eLink Platform must be installed prior to installing the eLink DIO Server software for your execution environment. You must install the eLink DIO Server execution components within the TUXEDO directory.

Installing eLink DIO Server on Windows NT

Perform the following steps to install the eLink DIO Server software on a Windows NT system.

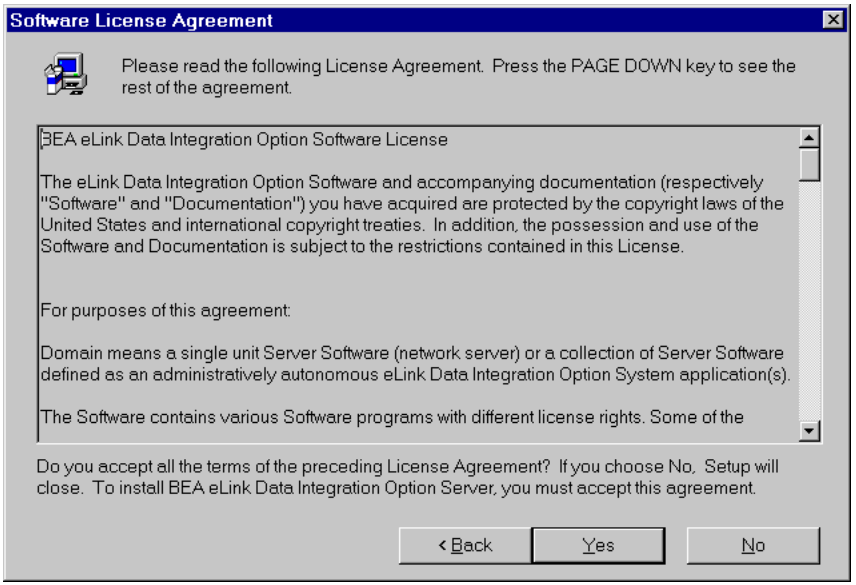
1. Insert the product CD-ROM and click on the **Run** option from the **Start menu**. The **Run** window displays. Click on the **Browse** button to select the CD-ROM drive. Select the `server/winnt` directory and select the `Setup.exe` program. Click **OK** to run the executable and begin the installation. The following **Welcome** screen displays. Click **Next** to continue with the installation.

Figure 3-6 Welcome



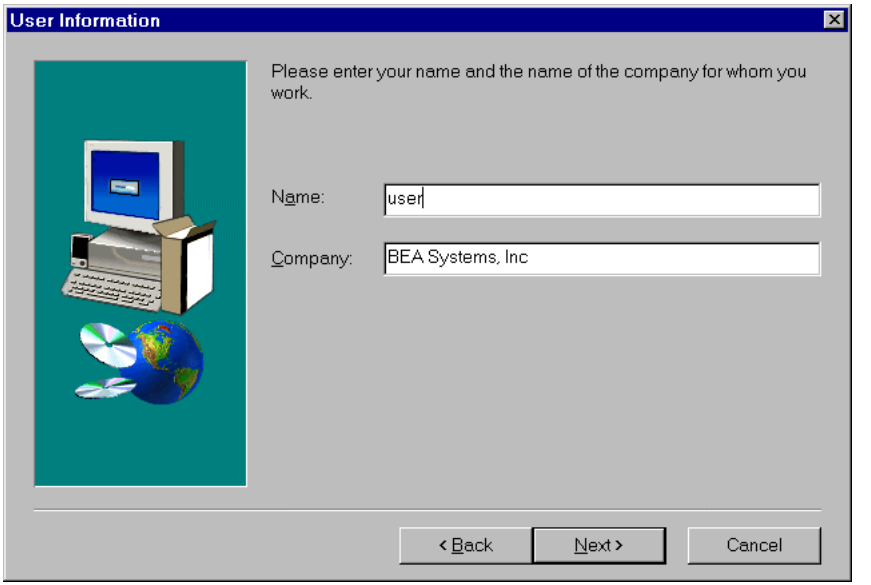
2. The **Software License Agreement** screen displays after the **Welcome** screen. Click **Yes** to accept the terms of the agreement and continue with the installation. Click **No** to abort the installation process.

Figure 3-7 Software License Agreement

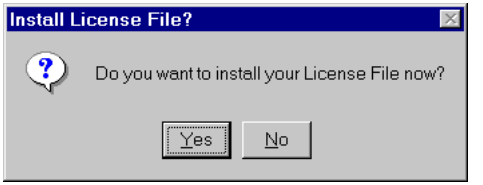


3. The **User Information** screen displays after the **Software License Agreement** screen. Enter your name in the **Name** field. Enter the name of your company in the **Company** field. Click **Next** to continue with the installation.

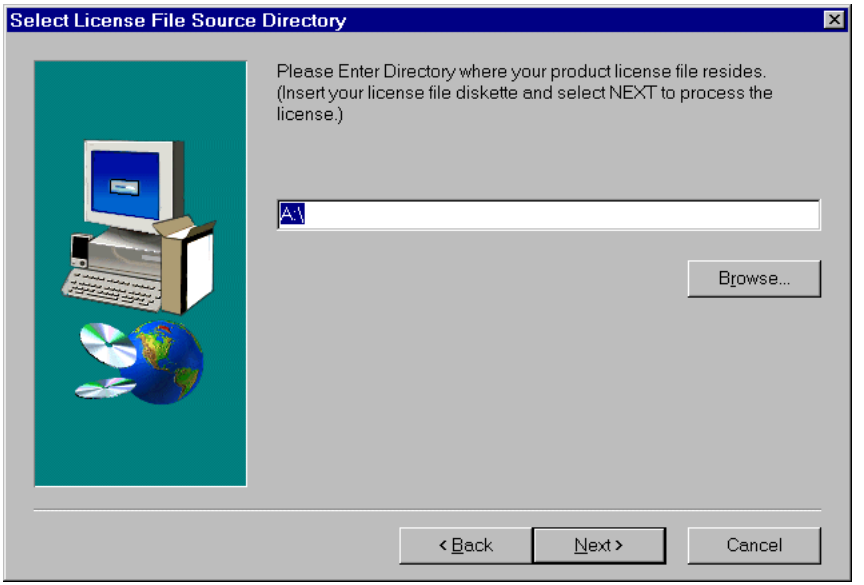
Figure 3-8 User Information



4. The **Install License File?** screen displays after the **User Information** screen. For additional license key information, refer to the *BEA eLink Data Integration Option Release Notes*.



   a. Click **Yes** to install the license file now. The following screen displays prompting you for the location of the license file. Browse to select the directory location of the license file. Click **Next** to continue with the installation process.



b. Click **No** to bypass the installation of the license file now. Be sure to install the license file prior to initializing the software.

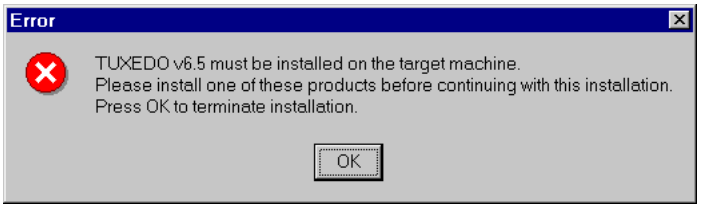
5. If BEA TUXEDO is installed and detected on your system, the installation begins and a progress bar displays the status. The eLink DIO Server components install into the TUXEDO directory. You may abort the installation process anytime prior to completion by clicking the **Cancel** button.

   When the installation completes, the **Setup Complete** screen shown in Step 6 notifies you that the eLink DIO Server software is installed on your system.

   If BEA TUXEDO is not installed on your system, the following **Error** screen displays.

   Warning: If Windows NT is your execution environment, BEA TUXEDO should be installed first and the eLink DIO Server should be installed within the same directory. If you install the eLink DIO Server outside of the TUXEDO directory, you will need to copy the files into the TUXEDO directory for processing of data mapping service requests.

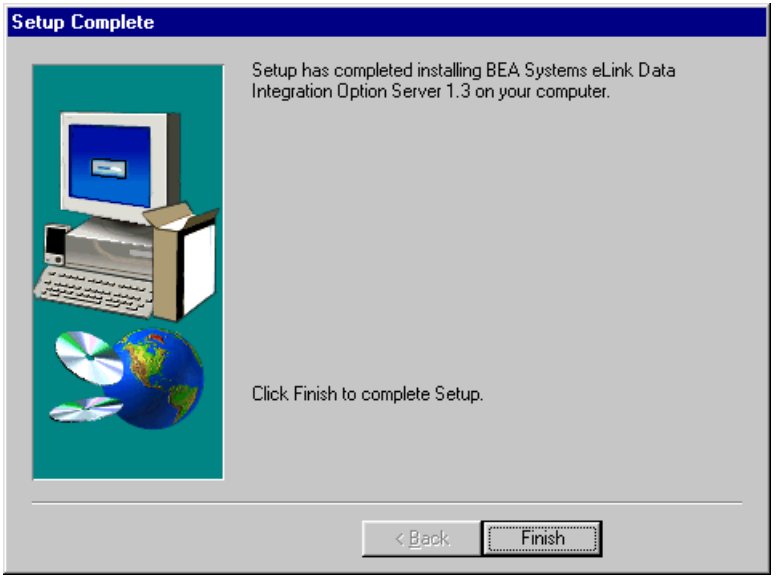
Figure 3-9 Error



Click **OK** to terminate the installation process.

6.The **Setup Complete** screen notifies you that the eLink DIO Server software is installed on your system. Click **Finish** to complete the setup process.

Figure 3-10 Setup Complete



Installing eLink DIO Server on a UNIX Platform (HP-UX, AIX, or SUN Solaris)

To install the eLink DIO Server software, you run the `install.sh` script. This script installs all the necessary software components.

Note: The eLink DIO Server should be installed into the BEA TUXEDO directory. Prior to initiating the installation script, determine the directory location of BEA TUXEDO.

Perform the following steps to install the eLink DIO Server software on a supported Unix platform:

1. Log on as root to install the eLink DIO software.

```
$ su -
Password:
```

2. Access the CD-ROM device.

```
# ls -1 /dev/cdrom

total 0

brw-rw-rw-     1 root     sys     22,   0 May 17 10:55 c1b0t010
```

3. Mount the CD-ROM.

```
# mount -r -F cdfs /dev/cdrom/c1b0t0l0 /mnt
```

4. Change the directory to your CD-ROM device.

```
# cd /mnt
```

5. List the CD-ROM contents.

```
# ls

install.sh  hp sun5x
```

6. Execute the installation script.

```
# sh ./install.sh
```

7. The installation script runs and prompts you for responses.

Listing 3-1 Install.sh Example

```
01) hp/hpux1020      02) hp/hpux11        03) ibm/aix43
04) sun5x/sol26      05) sun5x/sol7

Install which platform's files? [01-5, q to quit, l for list]: 1

** You have chosen to install from hp/hpux1020 **

BEA eLink Data Integration Option Release 1.3

This directory contains the BEA eLink Data Integration Option System
for
HP-UX 10.20 on 9000/800 series.


Is this correct? [y,n,q]: y

To terminate the installation at any time
press the interrupt key,
typically <del>, <break>, or <ctrl+c>.

The following packages are available:

  1     dio             BEA eLink Data Integration Option

Select the package(s) you wish to install (or 'all' to install
all packages) (default: all) [?,??,q]: 1

BEA eLink Data Integration Option
(9000) Release 1.3
Copyright (c) 2000 BEA Systems, Inc.
All Rights Reserved.
Distributed under license by BEA Systems, Inc.
BEA eLink is a trademark of BEA Systems, Inc.


Directory where Data Integration Option files are to be installed
(Enter your Tuxedo directory path) [?,q]: /work/cmadm/tux65

Using /work/cmadm/tux65 as the Data Integration Option base
directory

Determining if sufficient space is available ...
1856 blocks are required

6723480 blocks are available to /work/cmadm/tux65
```

```
Unloading /cmhome/dist/blue-2/server/hp/hpux1020/dio/DIOT65.Z ...
bin/ELINKDIO
bin/lic.sh
lib/libadk.sl
lib/libdbutil.sl
lib/libm4elink.sl
1780 blocks
... finished

Changing file permissions...
... finished

If your license file is accessible, you may install it now.
Install license file? [y/n]: n

Please don't forget to use lic.sh located in your product bin
directory
to install the license file from the enclosed floppy.
Refer to your product Release Notes for details on how to do this.


Installation of BEA eLink Data Integration Option was successful

Please don't forget to fill out and send in your registration card
```

---

8. Change the directory to your root directory.

```
# cd /
```

9. Unmount the CD-ROM device.

```
# unmount /mnt
```

Distribution Libraries and Executables for eLink DIO Server

The eLink DIO Server CD-ROM contains the following libraries and executable programs. After installing the eLink DIO Server software, verify that these libraries and programs are installed on your system.

HP-UX

Verify that the following files are installed by the eLink DIO software:

Table 3-2 HP-UX Installed Files

Directory	Files
/bin	ELINKDIO
/lib	libadk.sl libm4elink.sl libdbutil.sl

SUN Solaris

Verify that the following files are installed by the eLink DIO software:

Table 3-3 SUN Solaris Installed Files

Directory	Files
/bin	ELINKDIO
/lib	libadk.so libm4elink.so libdbutil.so

IBM AIX

Verify that the following files are installed by the eLink DIO software:

Table 3-4 IBM AIX Installed Files

Directory	Files
/bin	ELINKDIO

Directory	Files
/lib	libadk.so
	libm4elink.so
	libdbutil.so

Windows NT 4.0

Verify that the following files are installed by the eLink DIO software:

Table 3-5 Windows NT Installed Files

Directory	Files
/bin	ELINKDIO.exe
	m4elink.dll
	libadk.dll
	dbutil.dll
	dbutil32.dll
	runmer32.dll
	mercma32.dll
	mercad32.dll
	mercrm32.dll
	mercadpt.dll

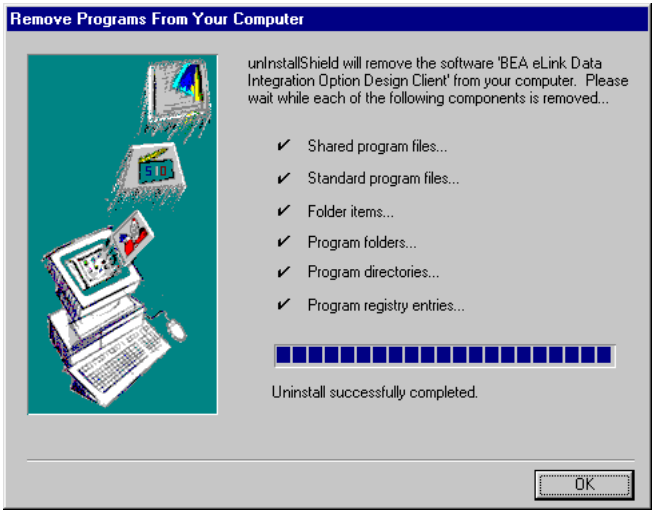
Uninstalling eLink DIO on Windows NT

Perform the following steps to uninstall the eLink DIO software on a Windows NT system:

1. Access the **Control Panel** window from the **Start>Settings>Control Panel** menu option.

2. Double-click on the **Add/Remove Programs** option from the Control panel listings to access the **Add/Remove Programs Properties** window.
3. In the **Add/Remove Programs Properties** window, select **BEA eLink Data Integration Option 1.3** from the program list and click the **Add/Remove** button.
4. The uninstall process for BEA eLink Data Integration Option begins. The **Remove Programs From Your System** screen displays. Click **OK** to complete the uninstall process.

Figure 3-11 Removing Programs From Your Computer



4 Configuring the Environment for eLink DIO

Before running the eLink DIO, you must configure the eLink Platform torecognize the ELINKDIO server and the eLink DIO configuration file.

Configuring the environment for eLink DIO consists of the following basic tasks:

- Configuring the ELINKDIO Server on eLink Platform
- Configuring eLink DIO

Configuring the ELINKDIO Server on eLink Platform

Before running the eLink DIO product, you must edit the UBBCONFIG file to recognize the ELINKDIO server that will be used for the data mapping. For UBBCONFIG parameter syntax and definitions, refer to the *BEA TUXEDO Reference Manual* on the eLink Platform online documentation CDROM.

Define the ELINKDIO Server on eLink Platform

Define the ELINKDIO server as the server responsible for the data mapping operation. To define this server, add the ELINKDIO information in the *SERVERS section of the UBBCONFIG file. The following parameters are required for defining the ELINKDIO server.

Note: You must specify `ELINKDIO` as the file to execute for the data mapping process. Within this server definition, you must also specify the `-C` option in the CLOPT parameter to pass the `dio.cfg` file.

Listing 4-1 Syntax for ELINKDIO Server Definition in the UBBCONFIG File

```
*SERVERS

ELINKDIO
        SRVGRP=groupname
        SRVID=n
        CLOPT="-A -- -C dio.cfg -WUD -TIO -AE"
```

For information about the SRVGRP, SRVID, and CLOPT parameter syntax and definitions, refer to the *BEA TUXEDO Reference Manual* on the eLink Platform online documentation CDROM. The definition of the `-C` option in the CLOPT parameter follows and is required for configuration process.

CLOPT="-A -- -C *configuration_file*"
: specifies the name of the eLink DIO configuration file.

  The options specified to the right of the double dashes (`-- -WUD -TIO -AE`) in Listing 4-1 are optional and write trace files and audit logs. These Mercator options may slow performance; however, may assist in troubleshooting. For an explanation of these options, refer to the *Execution Command Reference Guide* in the Mercator online documentation.

Sample UBBCONFIG File for Data Mapping

Listing 4-2 is a sample UBBCONFIG file for Windows NT. In this sample, the ELINKDIO server is defined in the *SERVERS section with the required CLOPT -C option specified.

Listing 4-2 Sample UBBCONFIG File for Data Mapping

```
*RESOURCES

IPCKEY        123791
DOMAINID      simpapp
MASTER        simple

*MACHINES

DALNT6
       LMID          = simple
       TUXDIR        = "\tuxedo"
       TUXCONFIG     = "\myappdir\tuxconfig"
       APPDIR        = "\myappdir"
       FIELDTBL      = "sample.fml"
       FIELDTBL32    = "sample.fml"
       FLDTBLDIR     = "\myappdir"
       FLDTBLDIR32   = "\myappdir"
       ULOGPFX       = "\myappdir\ULOG"
                              # LD_LIBRARY_PATH=\mercator
                              # SHLIB_PATH=\mercator
                              PATH=\mercator


*GROUPS

eLINK
       LMID=simple   GRPNO=1

*SERVERS

DEFAULT:
       CLOPT="-A"

ELINKDIO
       SRVGRP=eLINK         SRVID=10
       REPLYQ=N
       CLOPT="-A -- -C dio.cfg -WUD -TIO -AE"
```

```
*ROUTING
```

Configuring eLink DIO

In addition to the UBBCONFIG file defining the ELINKDIO server, the eLink DIO product must define the configuration parameters for data mapping and data translation. The eLink DIO product requires a configuration file, such as `dio.cfg`, separate from the UBBCONFIG file.

The eLink DIO configuration file requires the following sections:

- SERVER

  The SERVER section defines the eLink DIO server for data mapping.

- SERVICE

  The SERVICE sections define the various services the specified server supports.

Define the eLink DIO SERVER Section

Define the eLink DIO configuration parameters for the data mapping operation. The following parameters are required for defining a server in the eLink DIO configuration file.

Listing 4-3 Syntax for the *SERVER Section in the eLink DIO Configuration File

```
*SERVER

MAP-DIR=map_file_dir
MTI-DIR=mti_file_dir
NOCACHE=Y
SERVICE_LIST=A,B,C,...
```

*SERVER
denotes the start of server parameters.

MAP_DIR=*map_file_dir*
specifies the directory path where the Mercator map files reside. This location can be a directory other than APPDIR.

The default is APPDIR.

MTI_DIR=*mti_file_dir*
specifies the directory path where the MTI files reside. This location can be a directory other than APPDIR.

The default is APPDIR.

NOCACHE=Y
specifies whether to disable the MTI and Mercator map caching feature.

Y causes eLink DIO to access the MTI files and Mercator map files from disk each time a service executes. You can load changes to the MTI files or Mercator map files without recycling eLink Platform.

The default behavior is for eLink DIO Server to cache map and MTI files and access them from the cache.

SERVICE_LIST=*servicenameA, servicenameB, servicenameC,...*
specifies the list of services that this server will support.

Define the eLink DIO SERVICE Section

After defining the eLink DIO server in the SERVER section, you must specify the services that the server supports. The following parameters are required for defining a service in the eLink DIO configuration file.

Advertise the service for data mapping by defining it in the SERVICE section of the eLink DIO configuration file.

Listing 4-4 Syntax for Advertising the Mapping Service

```
*SERVICE=servicename

MAP_FILE=mapname
[MTI_FILE=mtiname]
BUFFER=FML32 | FML32_VAL | STRING | CARRAY
FML32_VAL_FLD=fieldname
```

*SERVICE=*servicename*

specifies a 1-15 character name of the service for data mapping or transformation process. The eLink DIO product advertises this servicename during server initialization.

MAP_FILE=*mapname.mmc*

specifies the name of the Mercator map file to use when executing this service. Multiple services can use the same map file. This is a required parameter.

MTI_FILE=*mtiname.mti*

specifies the name of the MTI file to use when executing this service. Multiple services can use the same MTI file.

Note: This parameter is only required for BUFFER=FML32.

BUFFER=FML32 | FML32_VAL | STRING | CARRAY

specifies the type of input buffer the eLink DIO service receives when called. Specifying the input buffer type allows eLink DIO to validate the input buffer to ensure the service is invoked correctly.

FML32 specifies that eLink DIO should receive an FML32 input buffer. This buffer type uses MTI processing and requires the MTI_FILE parameter.

STRING specifies that eLink DIO should receive a string input buffer.

CARRAY specifies that eLink DIO should receive a character array buffer as input.

FML32_VAL specifies that eLink DIO should receive an FML32 input buffer that contains one large field with all the data necessary for processing. The FML32_VAL-FLD parameter is required to process this type of buffer.

FML32_VAL_FLD=*fieldname*

specifies the name of the FML32 field in the input buffer which contains data for processing. This parameter replaces the MERCATOR_FV_IN field used in previous releases of eLink DIO.

Note: This parameter is only required when specifying BUFFER=FML32_VAL.

5 Data Mapping Conversion

The eLink DIO product provides a method for data mapping and conversion for the eLink Platform environment. The eLink DIO product can transform data from one format to another, split a single input into multiple outputs (message explosion), or perform intelligent data dependant routing. Several applications work together to accomplish this task. The following information will outline these tasks.

Creating Files for Conversion

You must create source and destination files prior to the conversion process. For converting non-FML to FML buffers or FML buffers to non-FML, you must have the following files:

- non-FML source file that meets the Mercator conversion standards

  For information about the source file requirements, refer to the Mercator Help file.

- FML Field Definition file

For information about creating an FML Field Definition file, refer to the *BEA TUXEDO Programmer's Guide* and the *BEA TUXEDO Reference Manual* on the BEA eLink Platform Online Documentation CD-ROM.

- FML Group Format file

  For information on the format and definition of the FML Group Format file, refer to the "Creating an FML Group Format File" section.

Creating an FML Group Format File

An FML32 Group Format file (`*.fgf`) is used with an FML32 Field Definition Table to apply structure to FML32 fields received by the eLink DIO in an FML input buffer. Data structures applied to input FML32 buffers can range from simple to complex. A simple structure may be an FML Group Format that supplies the order of individual FML32 fields received by a service request. An example of a complex structure could have nested groups of data items with multiple occurrences of some or all of the groups or data items.

Listing 5-1 is the syntax for the FML Group Format file.

Listing 5-1 Syntax for FML Group Format File

```
GROUP <groupname> [OCCURS n | s]
      field-type FML-field-name [OCCURS n | s]
END
```

GROUP

specifies a collection of data items that should be treated as an individual unit. The GROUP keyword marks the beginning of a group. Groups are terminated by the END keyword. GROUPS can be nested. The outermost group (or root) contains all of the fields or groups used in the FML32 buffer.

Note: The top GROUP (or root) cannot have an OCCURS clause.

<groupname>

specifies the name for the group. You can use the name of the FML Field Definition file as the `groupname` for the top GROUP.

<field-type> <FML-field-name>

specifies fields within an FML Group Format file. Fields are identified using the field type and the FML field name. Valid field types are string, long, short, char, double, float, or carray. The FML field name must match the name specified in the FML Field Definition file or an error will occur.

Note: Group names and field names *must* be unique within an FML Group Format file.

OCCURS *n* | S

specifies the number of occurrences of a group or data item within a group.

specifying a number (*n*) denotes the number of times the specific group or data item occurs in an input FML32 buffer.

specifying the keyword `S` indicates that a group or data item has an indefinite number of occurrences. The number of groups or data items processed may vary from one service call to the next. The eLink DIO product determines how many occurrences of a group or data item exist in the input FML32 buffer and process the data accordingly. Only use the keyword S at the top group level under the outermost group.

The OCCURS parameter is optional. The default is 1 occurrence.

FML Group Format File Examples

Listing 5-2 is a simple FML Group Format file. This sample file supplies the order in which individual FML Fields are formatted for input to the data mapping process. This FML Group Format file specifies the structure to build using the FML32 data items in the input buffer with names that match the FML Group Format field names.

Listing 5-2 Simple FML Group Format File Example

```
GROUP NAME
      string FIRST_NAME
      char MIDDLE_INIT
      string LAST_NAME
      int AGE
END
```

Listing 5-3 is a sample FML Group Format file with nested groups and occurrences. The GAA_ARRAY is the outermost group and will not support an OCCURS S clause. The valid top level groups in the following example are GAA_UPPER and GAA_SECOND. The OCCURS S definition in GAA_UPPER indicates that this top level group has an indefinite number of occurrences. GAA_UPPER contains another group, GAA_LOWER. Groups nested within groups other than the outermost group must have a fixed number of occurrences. GAA_LOWER occurs five times within each occurrence of GAA_UPPER.

Listing 5-3 Sample FML Group Format File

```
GROUP GAA_ARRAY OCCURS
      GROUP GAA_UPPER OCCURS S
            short EMA_S_SHORT
            GROUP GAA_LOWER OCCURS 5
                  string  EMA_S_STR OCCURS 2
                  long EMA_L_LONG OCCURS 4
            END
            string EMA_S_STR2
```

```
      END
      GROUP GAA_SECOND OCCURS 3
            short EMA_S_SHORT
            short EMA_S_CHAR
      END
END
```

Using the mti2fgf Utility

The BEA eLink DIO Design Client installs the mti2fgf utility. This utility takes a file containing Meta-Type Information and converts it back to FML Group Format language. This utility is helpful if you have the Meta-Type Information file, but no longer have the source FML Group Format file. After using the utility, the resulting `*.fgf` file is sent to stdout and can be used in the eLink DIO FML Importer.

To use the `mti2fgf` utility, issue the following command.

Listing 5-4 mti2fgf Utility

```
$ mti2fgf filename.mti
```

Developing Rules for Conversions Between FML Buffers and Other Applications

After you transfer the necessary files to the development environment, you can begin the process for building a Mercator map file. The map file will contain the specific mapping rules used to transform input data to desired output data formats. The following tasks must be done in the development environment prior to the execution of a service request.

1. Set System Properties for the Mercator Open Edition Client

2. Create an FML Type Tree File Using the eLink DIO FML Importer
3. Create a Mercator Map File

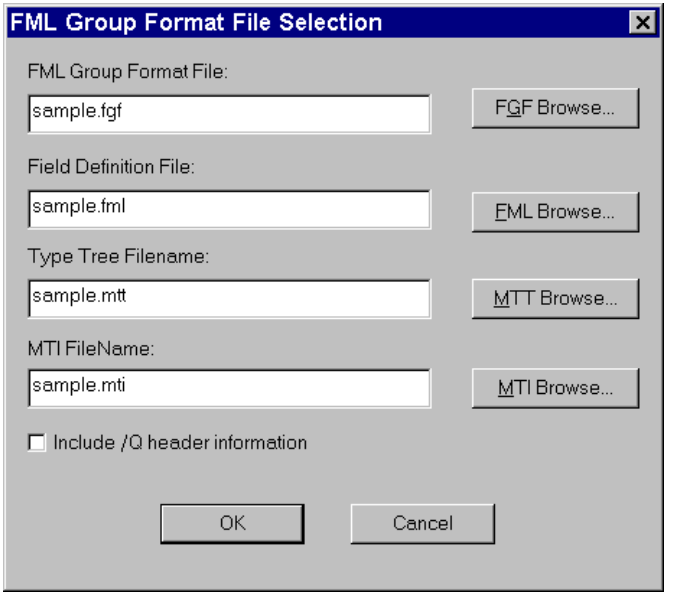
Set System Properties for the Mercator Open Edition Client

Before developing your data maps, you must put the Mercator Open Edition Client in your system path. Select **Program>Settings>Control Panel>System Properties** to access properties page. Select the **Environment** tab and enter the path for Mercator Open Edition Client under **User Variables**. The default installation directory is `C:\mercator`.

Create an FML Type Tree File Using the eLink DIO FML Importer

To create an FML type tree file, complete the following tasks.

1. Obtain the desired FML Field Definition file from the execution environment.
2. Create the necessary FML Group file (`.fgf`). For information on the FML Group Format file, refer to the "Creating an FML Group Format File" section. For information on creating FML Field Definitions file, refer to the *BEA TUXEDO Programmer's Guide* and the *BEA TUXEDO Reference Manual* on the eLink Platform Online Documentation CDROM.
3. Access the **eLink DIO FML Importer** dialog box by invoking `elinkimprt.exe`. The **FML Group Format File Selection** dialog box displays.



Browse to specify the following files on the FML Group Format File Selection.

FML Group Format File (`.fgf`) specifies the FML Group Format file to use in the data mapping conversion. This file is used in conjunction with an FML Field Definition file for the conversion.

Note: After selecting an FML Group Format file, the eLink DIO FML Importer will default the Type Tree and MTI file names to the same file name as the FML Group Format. The extensions will be unique, `*.mtt` and `*.mti` respectively.

FML Field Definition File (`.fml`) specifies the FML buffer to use in the data mapping conversion.

Type Tree File Name is the directory location and name of the FML type tree file to use for creating the Mercator Map file. If you reference only a file name and no directory path, the file will be created in the current directory.

MTI File Name is the directory path and file name for the metadata associated with the FML type tree file. If you reference only a file name and no directory path, the file will be created in the current directory.

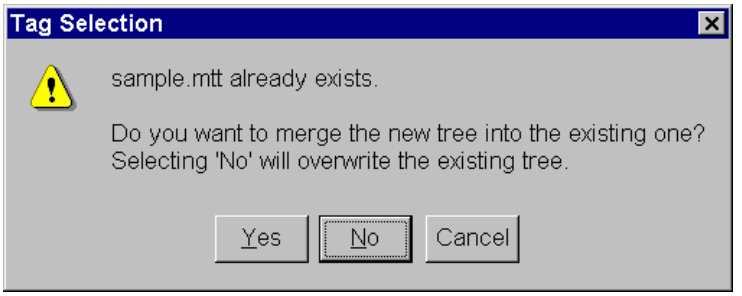
Note: Check the Include /Q header information box when using eLink Data Integration Option with the TUXEDO /Q feature. Do *not* check the Include

/Q header information box when using the BEA eLink Data Integration Option component with eLink for Mainframe.

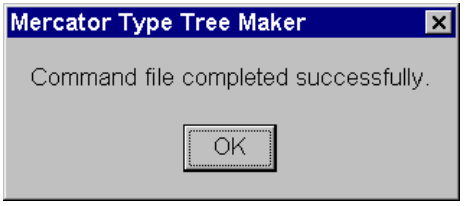
Click **OK** to validate the `.fgf` and `.fml` files and create the `.mtt` and `.mti` files.

If the following screen displays, click No to overwrite the file. No is the default.

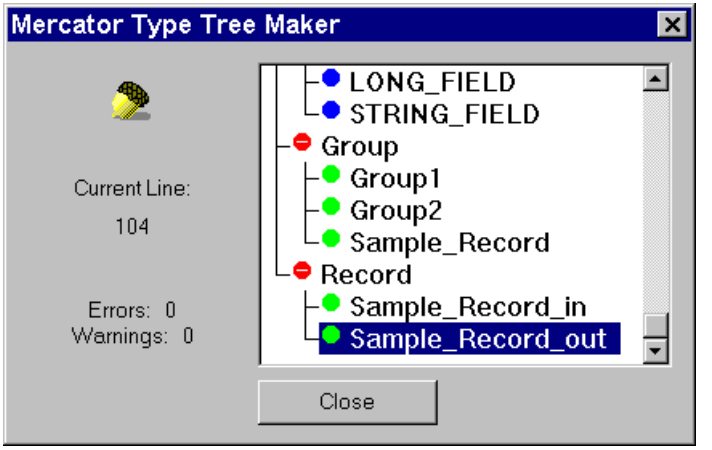
Note: Do *not* click Yes. It will merge the existing tree with the new tree.



4. If the FML type tree processes without error, the following pop up will display. Click **OK** to continue with the process.



5. The Mercator Type Tree Maker displays the newly created FML type tree file. Click **Close** to save the type tree file.



FML Type Tree Fields

The FML type tree specifies the type definitions of the input and output files. There are four branches for the type tree file; Control, Field, Message, and SET. The SET line is not used.

- Control contains general information about the field, which is defined in the Header file.
- Field contains individual field definitions
- Message
  - Input contains one or more sets
  - Output contains header information plus one or more sets
- SETS contain multiple occurrences of zero or more specific FML fields. There may be multiple sets.

Create a non-FML Type Tree File Using Importers

From your source file, create a type tree file (`*.mtt`).

1. Using the Mercator product, load your source file such as a COBOL copybook into the appropriate Importer and create a type tree file. For specific information about how to create a type tree using the Importer, refer to the Mercator Help file.
2. Using a text editor or the Mercator Type Editor, open the type tree file (*.mtt). If the type tree file does not contain the Control information (from the FML type tree file), add it into the header of the type tree file (*.mtt). This can be done by dragging and dropping the Control information from the FML type tree file into the other type tree file. Listing 5-5 is a sample of the header information that must be added to the COBOL type tree file before creating the Mercator map file (`.mmc`).

   Note: If you are using the TUXEDO /Q feature, refer to the "Control Information for TUXEDO /Q" section.

Listing 5-5 Sample COBOL Type Tree Header Information

```
CONTROL
    FIELD
        FieldName
        Format
        Service
```

3. Within the non-FML type tree file, create a new category for the input and output cards. For example, create a new category named TuxMessage.
4. Select the new category that was created, for example TuxMessage. Under this category, create two new groups. One group for input messages and another group for output messages.
5. After creating the two new groups, double click on the input message group to open the component screen. Drag and drop the appropriate group under the record category into the component screen of the input message.
6. Double click on the output message group to open the component screen. Drag and drop the Header group under the Control category into the component screen of the output message. Then drag and drop the appropriate group under the Record category into the component screen of the output message.

7. Analyze the tree by selecting `Analyze` from the **Tree menu**. Select the **Results** button to view the analysis results. If there are no errors, save the file. If there are errors, refer to the *Mercator Type Editor Reference Guide* for information on resolving these errors. Be sure to save the file when finished.

Control Information for TUXEDO /Q

Control information is used to return /Q message attributes to specify data formats of /Q message attributes in output cards. Specifically,

- In the first output card of maps used with the generic translation server, specify the data format and optional service name for the purposes of forwarding. The header information must be present in maps called by the generic translation server because it provides the only means of specifying the format of the output data.
- In output cards calling TUXEDO services and /Q, use the control information to specify the data format, service name and /Q message attributes.
- In input cards calling /Q, the control information is included in the data if the -HDR option is specified. This data contains attributes of the received message.

The control information consists of the following elements.

Table 5-1 Control Information

	Usage		
Element	**Service / /Q**	**When Present**	**Description**
Format	Both	Always	Specifies the desired format of the output buffer. Possible values are “CARRAY”, “STRING”, “FML”, and “FMLV” (FML values). Abbreviations of these may also be used. These are respectively: “C”, “S”, “F”, and “FV”.
View	Both	Always	Specifies the name of the view if the Format type is VIEW. **Note:** This is not a valid format.

Table 5-1 Control Information

FieldName	Both	Always	Specifies the name of the FML field to be specified for the output data. This is used if the Format type is FMLV (FML values).
Priority	/Q	/Q only	On output, specifies the priority of the message to be sent. This overrides the -P option.
CorrID	/Q	/Q only	On output, specifies the correlation ID of the message to be sent. This overrides the -CID option.
ReplyQ	/Q	/Q only	On output, specifies the reply queue of the message to be sent. This overrides the -RQ option.
FailureQ	/Q	/Q only	On output, specifies the failure queue of the message to be sent. This overrides the -FQ option.
URCode	/Q	/Q only	On output, specifies the user return code of the message to be sent. This overrides the -UR option.

Create a Mercator Map File

An input card and an output card must be created prior to compiling a Mercator map file (`*.mmc`). Input cards contain a type tree and an adapter. Use both type tree files for the buffer types (for example FML and COBOL copybook) that are being mapped. For information on how to build a Mercator map file, refer to the *Mercator Map Editor Reference Guide* and the *Mercator Functions & Expressions Reference Guide.*

1. Access the Mercator Map Editor window.
2. Set the map file name by selecting the **Rename** option from the **Map menu**. Enter a map name that is 8 characters or less, for example `FML2COB`.
3. After naming the map file, save the file by selecting **File>Save As**. Browse to select the directory and file name, for example `FML2COB`.

4. Activate the **From** side of the **Mercator Map Editor** window. To add an input card, select **Add** from the **Card menu** to display the **Add Input Card** dialog box.

   Enter a unique name for the new input card in the **Card Name** field.

   In the **Type** section, click **File** to browse and select the input card type tree file (`*.mtt`) for the File field.

   In the **Name** field of the **Type** section, click the **Browse** button to display the contents of the type tree file (`.mtt`). From the display, select the input record.

   Leave the default value, **File**, in the **Data Source** field.

   In the **File** section, you can use the **Name** field as a comment box. Enter a description of the input card as your comment. Text must be entered in this field.

   Note: The value in this field is ignored and has no effect on the data mapping process; therefore, you can use this field as a comment field.

   Click **OK** to create the input card.

5. Activate the **To** side of the **Mercator Map Editor** window. To add an output card, select **Add** from the **Card menu** to display the **Add Output Card** dialog box.

   Enter a unique name for the new output card in the **Card Name** field.

   In the **Type** section, click **File** to browse and select the output card type tree file (`*.mtt`) for the **File** field.

   In the **Name** field of the **Type** section, click the **Browse** button to display the contents of the type tree file (`.mtt`). From the display, select the output record.

   Leave the default value, **File**, in the **Data Source** field.

   In the **File** section, you can use the **Name** field as a comment box. Enter a description of the output card as your comment. Text must be entered in this field.

   Note: The value in this field is ignored and has no effect on the data mapping process; therefore, you can use this field as a comment field.

   Click **OK** to create the output card.

6. On the **To** side, formulate your mapping rules. You must define the **Format** field in the Header Control Group with a mapping rule. If the output record is a COBOL copybook, enter `="C"` as the rule in the **Format** field. If the output record is an FML buffer, enter `="FML"` as the rule in the **Format** field.

   Map the remaining fields. For information on mapping rules and defining functional maps, refer to the *Mercator Functions & Expressions Reference Guide*.

7. Compile your map file (`*.mmc`) by selecting the **Build** option from the **Map menu**. If errors occur during the build process, access the building results for an explanation of the error. Resolve the errors in the mappings and execute the build process again until no errors occur.

8. Save the compiled map file by selecting **File>Save**.

9. If your execution platform is HP-UX, AIX, or SUN Solaris, you must port the maps to these platforms. To port the map, select **Map>Port** and then select the platform from a list. When the port is complete, the files will be renamed with a platform-specific extension, such as `*.hp`.

Executing a Service Request with Data Mapping

To set up the execution environment, complete the following tasks:

1. Transfer Files to the Execution Environment
2. Set Environment Variables
3. Start BEA TUXEDO and Other Remote Applications
4. Write a Client Program to Call eLink DIO Services
5. Issue a Service Request for Data Mapping

   The client program requests the mainframe service advertised by the eLink for Mainframe gateway.

Transfer Files to the Execution Environment

After completing the development process, the following files must exist and be transferred to the execution environment.

- FML destination file (`*.fml` - ASCII file)

  Transfer this file to the execution environment. Define the FLDTBLDIR32 to point to the file's directory structure. Also, define the FML Field Definition file with other FML32 tables in the FIELDTBL32 environment variable.

- Meta-Type Information file (`filename.mti` - ASCII files)

  Transfer the Meta-Type Information file to the directory listed as MTI_DIR in the SERVER section of the eLink DIO configuration file. If the eLink DIO configuration does not specify an MTI_DIR, then load these files to APPDIR.

- Mercator map files (`filename.mmc` - Binary file)

  Note: If you are executing on HP-UX, AIX, or SUN Solaris, the map files will have a platform-specific extension after the porting function. For example, after porting the `filename.mmc` file to Sun Solaris, the file is named `filename.sun`.

  Transfer the Mercator map files to the directory listed as MAP_DIR in the SERVER section of the eLink DIO configuration file. If the eLink DIO configuration does not specify a MAP_DIR, then load these files to APPDIR.

- Client programs that create input buffers for eLink DIO and call eLink DIO services.

Set Environment Variables

Be sure to set the following environment variables.

- TUXDIR identifies the location of the eLink Platform installation.
- APPDIR identifies the location of user-developed eLink Platform application files.
- FLDTBLDIR identifies directories that contain FML Definition Tables.

- FLDTBLDIR32 identifies directories that contain FML Definition Tables.
- FIELDTBL identifies individual FML Definition Tables used by eLink Platform and eLink DIO.
- FIELDTBL32 identifies individual FML Definition Tables used by eLink Platform and eLink DIO.

The following environment variables are platform-specific. Be sure to set the appropriate one for your execution environment. These parameters identify to the system the location of shared libraries needed to execute eLink Platform and eLink DIO.

- LD_LIBRARY_PATH (for SUN Solaris)
- SHLIB_PATH (for HP-UX)
- LIBPATH (for AIX)
- PATH (for Windows NT)

Start BEA TUXEDO and Other Remote Applications

After transferring the necessary files to the execution environment, be sure that the BEA TUXEDO and other products such as BEA eLink for Mainframe are running. For information on how to start these products, refer to the related BEA TUXEDO documentation and the BEA eLink for Mainframe online documentation CD.

Write a Client Program to Call eLink DIO Services

The client program should be a standard TUXEDO program. This program should create and populate an FML32, STRING, or CARRAY buffer. The client calls a service advertised by the eLink DIO Server using this buffer as input. The eLink DIO Server performs data mapping and returns any required output to the client program.

Issue a Service Request for Data Mapping

Execute client programs previously created to perform service calls to services supported by the eLink DIO Server.

After starting these products, you can issue a service request for data mapping. For information on setting up TUXEDO service requests, refer to the BEA TUXEDO documentation.

Note: Leave logging on during execution of your map files to assist in diagnostics if errors occur. Logging will produce a large amount of output. The log file (.log) can be accessed directly in the directory where the compiled map file (*.mmc) is located.

A Error and Informational Messages

This document contains the following descriptions of error, informational, and warning messages that can be encountered while using the BEA eLink DIO component.

201	**ERROR: Cannot open '<*filename*>' (errno=<*os-errno-value*> - '<*os-errno-text*>'**	
	Description:	The specified Meta-Type Information file cannot be opened.
	Action:	A Meta-Type Information file must exist and be located in the application directory identified by the APPDIR environment variable defined in the ubbconfig file (or the ENV file referenced in the ubbconfig file). Verify its presence and that 'USER' has read permissions.
	See Also:	None.
202	**ERROR: Cannot restore type description from '<*buffer-type*>' (rc=<*response-code*>)'**	
	Description:	The buffer's data type cannot be determined because its definition is not available. Refer to the supplemental code for details.
	Action:	Examine the supplemental code and correct the problem.

	See Also:	None.
203	**ERROR: Cannot translate input FML data '(rc=<*response-code*>)'**	
	Description:	FML data could not be translated into an internal format and made available to the Mercator mapping engine. Refer to the supplemental code for details.
	Action:	Examine the supplemental code and correct the problem.
	See Also:	None.
204	**ERROR: Cannot open '<*filename*>' (APPDIR not set)**	
	Description:	The specified Meta-Type Information file <*filename*> cannot be opened because the APPDIR environment variable is either not available or incorrectly initialized.
	Action:	A Meta-Type Information file must exist and be located in the application directory identified by the APPDIR environment variable defined in the ubbconfig file (or the ENV file referenced in the ubbconfig file). Verify the definition of the APPDIR environment variable.
	See Also:	None.
205	**ERROR: Illegal buffer type '<*buffer-type*>' (tperrno=<*tuxedo-errno-value*> - '<*tuxedo-errno-text*>')**	
	Description:	The buffer received by the eLink DIO service handler is not a TUXEDO buffer type.
	Action:	Examine the TUXEDO error response code (`tperror`) along with its brief explanation text to determine the nature of the programming error. More detailed information may be available in the TUXEDO documentation.
	See Also:	None.
206	**ERROR: INTERNAL ERROR: Invalid parameters**	
	Description:	The eLink DIO has reached an unexpected state and may possibly become unstable.
	Action:	Contact Technical Support.
	See Also:	None.

207	**ERROR: Failure to map input buffer type '<*buffer-type*>' (rc=<*response-code*>)**	
	Description:	The eLink DIO is unable to convert the input buffer specified as <*buffer-type*> into an internal byte stream. Refer to the supplemental code for details.
	Action:	Examine the supplemental code and correct the problem.
	See Also:	None
208	**ERROR: Failure to map named (<*field-name*>) input buffer type '<*buffer-type*>' (rc=<*response-code*>)**	
	Description:	The eLink DIO is unable to convert the input buffer whose <*buffer-type*> is specified by the named FML field <*field-name*> into an internal byte stream. Refer to the supplemental code for details.
	Action:	Examine the supplemental code and correct the problem. <*buffer-type*> is displayed "<*invalid-buffer-type*>" if <*field-name*> is not found or out of range.
	See Also:	None.
209	**ERROR: Data mapping failure (rc=<*merc-code*> - '<*merc-message-text*>')**	
	Description:	This is a general data mapping response diagnostic. <*merc-code*> and <*merc-message-text*> are detailed in the Mercator documentation.
	Action:	Examine <*merc-code*> and <*merc-message-text*> for the cause of the failure and correct the problem.
	See Also:	None.
210	**ERROR: Invalid or missing control information (rc=<*response-code*>)**	
	Description:	The control information in output record data is missing or contains one or more invalid values. It cannot be processed by the eLink DIO. Refer to the supplemental code for the cause.

	Action:	The common cause for this failure is the value assigned to the FORMAT field of the Control information. Refer to this diagnostic code for details for correct assignment values.
	See Also:	None.
212	**ERROR: Failure to map output buffer, type not specified**	
	Description:	The eLink DIO could not process the result received from the Mercator mapping engine because no output buffer is specified.
	Action:	Specify an output buffer, either by using `tpalloc(3)` and sending this together with the request in the `tpcall(3)`, or by defining it in the control information header in the output record type tree.
	See Also:	None.
213	**ERROR: Failure to map output byte stream to FML (rc=<*response-code*>)**	
	Description:	The eLink DIO is unable to convert the output buffer from the internal byte stream received from the Mercator mapping engine.
	Action:	Examine the supplemental code and correct the problem.
	See Also:	None.
214	**ERROR: Failure to map output byte stream to '<*buffer-type*>' (rc=<*response-code*>)**	
	Description:	The eLink DIO is unable to convert the output buffer from the internal byte stream received from the Mercator mapping engine to the specified <*buffer-type*>.
	Action:	Examine the supplemental code and correct the problem.
	See Also:	None.
215	**ERROR: Illegal buffer type '<*buffer-type*>'**	
	Description:	The output buffer type is not one of the supported types and cannot be used to complete the mapping operation.
	Action:	Refer to diagnostic 225 for a list of supported buffer types.

	See Also:	None.
216	**ERROR: Memory allocation error**	
	Description:	A general memory allocation failure has occurred.
	Action:	Call Technical Support.
	See Also:	None.
217	**WARN: Cannot load shared library '<*filename*>'**	
	Description:	The eLink DIO cannot load its shared library. The eLink DIO uses a method in this library to obtain map execution status from the Mercator engine.
	Action:	If this condition is not corrected, the eLink DIO reports successful engine status regardless of the actual operation status. This library is expected to be installed in the TUXEDO `lib` directory (UNIX installations) or `bin` directory (Windows NT installations).
	See Also:	None.
218	**WARN: Failed to find function in TUXEDO adapter**	
	Description:	The eLink DIO cannot obtain its status reporting method from its shared library. The eLink DIO uses this method to obtain map execution status from the Mercator engine.
	Action:	Contact Technical Support.
	See Also:	None.
220	**INFO: Translation server started**	
	Description:	This diagnostic reports the ELINKDIO TUXEDO server is ready to service requests.
	Action:	None.
	See Also:	None.
222	**ERROR: Missing '*filename*'**	
	Description:	The Meta-Type Information disassembler requires the name of an existing Meta-Type Information file.

	Action:	Repeat the command using the name of an existing Meta-Type Information file.
	See Also:	None.
223	**ERROR: Cannot restore type from '<*filename*>' (rc=<*response-code*>)**	
	Description:	The Meta-Type Information disassembler is unable to disassemble <*filename*>. <*response-code*> indicates the nature of the failure.
	Action:	Refer to the diagnostic <*response-code*> for a detailed reason.
	See Also:	None.
224	**INTERNAL ERROR: TUXEDO API failure (tperrno=<*tuxedo-code*> - '<*tuxedo-message*>')**	
	Description:	The eLink DIO encountered a failure response from TUXEDO when performing the requested operation. The <*tuxedo-code*> is the numeric value of the diagnostic code. <*tuxedo-message*> is a brief description of the diagnostic response.
	Action:	None.
	See Also:	None.
225	**ERROR: Illegal FORMAT value '<*value*>' specified in control header**	
	Description:	The value of the FORMAT field of the control information in the output buffer is missing or out of range.
	Action:	The possible values are "CARRAY" (or "C"), "STRING" (or "S"), "FML" (or "F") and "FMLV" (or "FV"). Correctly assign this field in the control information of the output buffer.
	See Also:	None.
226	**WARN: Cannot locate '<*filename*>' (TUXDIR not set)**	
	Description:	The Adapter cannot load its shared library. The eLink DIO uses a method in this library to obtain map execution status from the Mercator engine.

	Action:	If this condition is not corrected, the eLink DIO reports successful engine status regardless of the actual operation status. This library is expected to be installed in the TUXEDO lib directory (UNIX Installation) or bin directory (Windows NT installations).
	See Also:	None.
227	**INFO: Service *servicename* advertised**	
	Description:	Service *servicename* has been advertised by TUXEDO.
	Action:	None.
	See Also:	None.
228	**ERROR: Service *servicename* tpadvertise failed**	
	Description:	An attempt to advertise Service *servicename* has failed.
	Action:	Determine the cause of the failure and restart eLink DIO.
	See Also:	None.
229	**ERROR: Configuration file is required for ELINKDIO**	
	Description:	The configuration file parameter was not specified or was specified incorrectly in the command line arguments for ELINKDIO server in the TUXEDO UBBCONFIG file.
	Action:	Correct the UBBCONFIG file command line arguments for the ELINKDIO server and restart the ELINKDIO server.
	See Also:	None.
230	**ERROR: NOMTI option not allowed with the configuration file**	
	Description:	The NOMTI parameter was specified in addition to an eLink DIO configuration file.
	Action:	Remove the NOMTI parameter or the eLink DIO configuration file specification and restart the ELINKDIO server.
	See Also:	None.

231	**ERROR: Unable to open configuration file *filename***	
	Description:	An attempt to open configuration file *filename* failed.
	Action:	Verify that the *filename* exists and that the file permissions allow for read access.
	See Also:	None.
232	**ERROR: Unable to locate SERVICE_LIST for server**	
	Description:	The SERVICE_LIST parameter is not present in the SERVER section of the eLink DIO configuration file.
	Action:	Add the SERVICE_LIST parameter to the SERVER section of the eLink DIO configuration file. Verify configuration file syntax.
	See Also:	None.
233	**ERROR: Service *servicename* not found**	
	Description:	The SERVICE section of the eLink DIO configuration file for *servicename* is not found. The ELINKDIO server has tried to find a SERVICE section for one of the services listed in the SERVICE_LIST of the SERVER section of the eLink DIO configuration file and is unable to find it.
	Action:	Verify that each service in the SERVICE_LIST in the SERVER section of the eLink DIO configuration file has a corresponding SERVICE section.
	See Also:	None.
234	**ERROR: Unable to open map file *filename* for service *servicename***	
	Description:	The ELINKDIO server has failed in an attempt to open the map file *filename* listed in the SERVICE section of the eLink DIO configuration file for *servicename*.
	Action:	Verify the map file *filename* exists and that the *filename* permissions allow for read access.
	See Also:	None.

235	**ERROR: Unable to open mti file *filename* for *servicename***	
	Description:	The ELINKDIO server has failed in an attempt to open the MTI file *filename* listed in the SERVICE section of the eLink DIO configuration file for *servicename*.
	Action:	Verify that the MTI file *filename* exists and that the *filename* permissions allow for read access.
	See Also:	None.
236	**ERROR: No FML32_VAL_FLD found for service *servicename***	
	Description:	The FML32_VAL_FLD parameter has not been specified for service *servicename* and a buffer type of FML32_VAL has been specified for *servicename*.
	Action:	Verify that FML32_VAL is the correct buffer type. If so, add the FML32_VAL_FLD parameter to the SERVICE configuration to specify the field name of the FML32 field containing the input data.
	See Also:	None.
237	**ERROR: No MAP_FILE found for service *servicename***	
	Description:	A MAP_FILE parameter has not been found for service *servicename* in the eLink DIO configuration file.
	Action:	Add the MAP_FILE parameter to the SERVICE section of the eLink DIO configuration file for service *servicename*.
	See Also:	None.
238	**ERROR: No BUFFER parameter specified for service *servicename***	
	Description:	A BUFFER parameter has not been found for service *servicename* in the eLink DIO configuration file.
	Action:	Add the BUFFER parameter in the SERVICE section of the eLink DIO configuration file for service *servicename*.
	See Also:	None.

239	**ERROR: Configuration Load Failed rc=*rcvalue***	
	Description:	An attempt to load the configuration file failed, with return code *rcvalue*.
	Action:	Determine the cause of the failure and restart eLink DIO.
	See Also:	None.
240	**ERROR: No MTI_FILE found for service *servicename***	
	Description:	A service *servicename* using the FML32 buffer type did not have an MTI_FILE specified in the configuration.
	Action:	Correct the configuration file for this service.
	See Also:	None.
241	**ERROR: Invalid buffer type *buffertype* for service *servicename***	
	Description:	An invalid buffer type *buffertype* was specified for service *servicename* in the eLink DIO configuration file.
	Action:	Verify that the buffer type for *servicename* is either FML32, FML32_VAL, CARRAY, or STRING.
	See Also:	None.
242	**ERROR: Unable to locate data for server**	
	Description:	Unable to locate the SERVER section in the eLink DIO configuration file.
	Action:	Verify the configuration file syntax and restart eLink DIO.
	See Also:	None.
401	**ERROR: Fieldname <*name*> not found in FML file**	
	Description:	An FML Field name referenced in this FML Group Format file is not defined in the FML Field Definition File.
	Action:	Provide field names in the FML Field Definition file.
	See Also:	None.
402	**ERROR: FieldName <*name*> has a different type from the FML file**	

	Description:	The data type specified for the FML field is not the same as the data type in the FML Field Definition file.
	Action:	Modify the information so that the field name and data type in the FML Field Definition file match.
	See Also:	None.
403	**ERROR: Fieldname <*name*> occurs multiple times in the FGF file**	
	Description:	This field name occurs multiple times in the FGF file, it should occur only once.
	Action:	Edit the FGF file so that the field name occurs only once.
	See Also:	None.
404	**ERROR: Cannot create MTI file (rc=<*response-code*>)**	
	Description:	This error occurs when the Meta-Type Information file cannot be saved.
	Action:	Check directory write permissions or disk space. If the error persists, contact Technical Support.
	See Also:	None.
410	**ERROR: Outermost GROUP in the FGF file must have exactly one occurrence**	
	Description:	The outermost GROUP in the FGF file cannot have an OCCURS clause with any number other than 1.
	Action:	Remove the OCCURS clause in the outermost group or set OCCURS=1.
	See Also:	None.
411	**ERROR: Found duplicate Group name <*name*> in FGF file**	
	Description:	Group Names within FGF file must be unique.
	Action:	Modify the group Names so that they are unique.
	See Also:	None.
420	**ERROR: Syntax Error at line <*line number*>**	

	Description:	There is a syntax error in the FML group format file.
	Action:	Correct the syntax and recompile. Some guidelines for correcting the syntax follow: ■ Verify that each GROUP statement has a corresponding END statement. ■ Verify that each GROUP statement has a name. ■ Verify that each GROUP has at least one statement. ■ Verify that your FML type is a string, char, carray, short, long, float, or double. ■ Verify that each OCCURS clause is supplied with a number.
	See Also:	Section “Creating an FML Group Format File” on page 5-2
3001	**INTERNAL ERROR: Program argument value null**	
	Description:	An eLink DIO contract has been violated; the eLink DIO may be in an unreliable state.
	Action:	Contact Technical Support.
	See Also:	None.
3002	**INTERNAL ERROR: Type descriptor unreliable**	
	Description:	Type descriptor reference unreliable.
	Action:	Contact Technical Support.
	See Also:	None.
3003	**INTERNAL ERROR: Filename argument out of range**	
	Description:	An eLink DIO contract has been violated; the eLink DIO may be in an unreliable state.
	Action:	Contact Technical Support.
	See Also:	None.
3004	**ERROR: Cannot open file for read**	
	Description:	The eLink DIO is unable to open a Meta-Type Information file for read access.

	Action:	Verify this filename prefix matches the prefix of the corresponding Mercator map file. For example, if the name of the Mercator map file is `ToCOBOL.mmc`, then the filename for the corresponding Meta-Type Information file must be `ToCOBOL.mti`. This file must be located in the application's directory (APPDIR environment variable) and have appropriate read permissions.
	See Also:	None.
3005	**ERROR: Cannot open file for writing**	
	Description:	The eLink DIO is unable to open a Meta-Type Information file for write access.
	Action:	Verify the file access permissions of the application directory (APPDIR environment variable). Appropriate write permissions must be enabled.
	See Also:	None.
3006	**INTERNAL ERROR: Null persistence file pointer**	
	Description:	An eLink DIO contract has been violated; the eLink DIO may be in an unreliable state.
	Action:	Contact Technical Support.
	See Also:	None.
3007	**INTERNAL ERROR: Null member name argument**	
	Description:	An eLink DIO contract has been violated; the eLink DIO may be in an unreliable state.
	Action:	Contact Technical Support.
	See Also:	None.
3008	**ERROR: Field '<*field-name*>' not found**	
	Description:	The FML buffer does not contain the field specified by <*field-name*>.

	Action:	Verify the buffer data, Meta-Type Information and Mercator map files are correct for the requested operation. This will likely occur if either one of these files or the buffer are incorrect.
	See Also:	None.
3009	**ERROR: Field '<*field-name*>' missing matching value**	
	Description:	The FML buffer does not contain the value for the field specified by <*field-name*>.
	Action:	Correct the buffer initialization step and proceed.
	See Also:	None.
3010	**ERROR: Field '<*field-name*>' value not found**	
	Description:	The FML buffer does not contain the value for the field specified by <*field-name*>.
	Action:	Correct the buffer initialization step and proceed.
	See Also:	None.
3011	**INTERNAL ERROR: Unsafe copy**	
	Description:	An eLink DIO contract has been violated; the eLink DIO may be in an unreliable state.
	Action:	Contact Technical Support.
	See Also:	None.
3012	**ERROR: Failure reading Meta-Type Information**	
	Description:	The Meta-Type Information file could not be read into process memory.
	Action:	The Meta-Type Information file may have become corrupted, or possibly it is not a Meta-Type Information file. Verify and regenerate this file.
	See Also:	None.

3013	**ERROR: Cannot grow persistence file**	
	Description:	The Meta-Type Information cannot be loaded into memory due to resource exhaustion.
	Action:	Contact Technical Support.
	See Also:	None.
3014	**ERROR: Cannot allocate heap space for object persistence**	
	Description:	This is a general memory allocation failure due to resource exhaustion.
	Action:	Contact Technical Support.
	See Also:	None.
3015	**ERROR: Not a Meta-Type Information file**	
	Description:	The eLink DIO does not recognize this file as a Meta-Type Information file.
	Action:	The Meta-Type Information file may have become corrupted, or possibly it is not a Meta-Type Information file. Verify or regenerate this file.
	See Also:	None.
3016	**ERROR: Meta-Type Information file contents unreliable**	
	Description:	The Meta-Type Information file contents could be read but not processed.
	Action:	The Meta-Type Information file may have become corrupted, or possibly it is not a Meta-Type Information file at all. Verify or regenerate this file.
	See Also:	None.
3017	**INTERNAL ERROR: Program argument out of range**	
	Description:	An eLink DIO contract has been violated; the eLink DIO may be in an unreliable state.
	Action:	Contact Technical Support.

See Also: None.

B Data Mapping Worksheet

This worksheet will assist you in setting up your data mapping environment properly.

Step 1: Install Products

The following products must be installed and operational during the execution of a data mapping service request. Install these products in the following order.

1. BEA eLink Platform 1.1

   Directory location: _____

2. TSI Mercator Open Edition Client 1.4.2

   Note: Use the SERIAL number from the eLink DIO license key when prompted for a Serial number during the Mercator Open Edition Client installation.

   Directory location: _____

3. TSI Mercator Open Edition Client 1.4.2 Service Pack 8

   Directory location: _____

4. BEA eLink Data Integration Option 1.3

   Directory location: _____

Step 2: Set eLink Platform Environment Variables

Set the following environment variables:

- TUXDIR
- APPDIR
- FLDTBLDIR
- FLDTBLDIR32
- FIELDTBL
- FIELDTBL32

The following environment variables are platform-specific. Be sure to set the appropriate one for you execution environment.

- LD_LIBRARY_PATH (for SUN Solaris)
- SHLIB_PATH (for HP-UX)
- LIBPATH (for IBM AIX)
- PATH (for Windows NT)

Step 3: Put Mercator in the System Path

Before developing your data maps on Windows NT, you must put the Mercator Open Edition Client in your system path. Select **Program>Settings>Control Panel>System Properties** to access properties page. Select the **Environment** tab and enter the path for Mercator Open Edition Client under **User Variables**. The default installation directory is `C:\mercator`.

Step 4: Transfer Source Files to Windows NT

1. Transfer the FML Field Definition files and non-FML source files to the development environment on the Windows NT system. Use the following table to track the location and names of these files.
2. Create the FML Group Format file to use with the FML Field Definition file in the development environment on the Windows NT system.

Use the following table to track the location and names of these files.

FML Field Definition Files	FML Group Format Files	non-FML Source Files

Step 5: UBBCONFIG File Definitions

The following configurations must be modified for data mapping service requests to process correctly.

1. Define a GROUP for data mapping. Use the following table to plan for the group information that needs to be specified.

GROUP Parameters	Arguments
groupname	*groupname* =
LMID=*logical-machine-ID*	*logical-machine-ID* =
GRPNO=*n*	*n* =

2. Define the ELINKDIO server in the TUXEDO UBBCONFIG file.

   Define the ELINKDIO server in the SERVERS section of the UBBCONFIG file. The ELINKDIO server is responsible for the data mapping operation. Use the following table to plan for the server information that needs to be specified.

SERVERS Parameters	Arguments
server identifier	ELINKDIO
SRVGRP=*groupname*	*groupname* =
SRVID=*ID*	*ID* =
CLOPT="-A -- -C *dio.cfg*"	*dio.cfg* =

Step 6: Create an FML Type Tree File

For information about how to create an FML type tree file, refer to "Create an FML Type Tree File Using the eLink DIO FML Importer" on page 5-6.

Step 7: Create a non-FML Type Tree File

For information about how to create a non-FML type tree file, refer to "Create a non-FML Type Tree File Using Importers" on page 5-9.

Step 8: Create a Mercator Map File

For information about how to create a Mercator map file, refer to "Create a Mercator Map File" on page 5-12.

Step 9: Port Map Files for Execution on UNIX Platforms

If your execution platform is HP-UX, AIX, or SUN Solaris, you must port the Mercator map files to these platforms. To port the map, access the Mercator Map Editor and select **Map>Port** and then select the platform from a list. When the port is complete, the files will be renamed with a platform-specific extension, such as `*.hp`.

Note: After transferring these ported files to the execution environment you must rename these files with the `*.mmc` extension again. The execution environment will not recognize the ported files with the extensions such as `*.hp`.

Step 10: Renaming Meta-Type Information File (Optional)

You must have a copy of the Meta-Type Information (`sample.mti`) file saved as the Mercator map name (`FML2COB.mti` and `COB2FML.mti`).

To obtain this file, copy the FML Meta-Type Information (`.mti`) file and rename it using the file name of both map files. For example, the FML type tree file is `sample.mti` and the Mercator map file is `FML2COB.mmc`. Copy `sample.mti` and rename it `FML2COB.mti`. The other map file is `COB2FML.mmc`. Then copy the `sample.mti` file and rename it to the `COB2FML.mti`.

Step 11: Transfer Files

For information about transferring files, refer to "Transfer Files to the Execution Environment" on page 5-15.

Step 12: Start eLink Platform and Remote Applications

For information about how to start eLink Platform and other remote application products, refer to the specific product online documentation.

Step 13: Issue a Service Request for Data Mapping

For information about setting up service requests, refer to the eLink Platform online documentation.