IBM WebSphere Transformation Extender

Pack for EDI

Version 2.7

Note
Before using this information, be sure to read the general information in "Notices" on page 97.

30 June 2006

This edition of this document applies to IBM WebSphere Transformation Extender Pack for EDI Version 2.7; and to all subsequent releases and modifications until otherwise indicated in new editions.

To send us your comments about this document, email DTX_doc_feedback@us.ibm.com . We look forward to hearing from you.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Chapter 1. Introduction

Electronic Data Interchange (EDI) is the electronic exchange of routine business information using an agreed-upon file structure. The file structures are defined with EDI standards for the electronic documents. These uniform electronic formats are referred to as standards. Standards, which are periodically updated, are referenced with versions, which specify releases. Version 4010 is version 004, release 010. Version 3050 is version 003, release 050.

Overview

The Packs for EDI are collections of X12, EDIFACT, TRADACOMS, ODETTE, and EANCOM type trees that represent different versions of an individual EDI standard.

The Packs for EDI that are available to download from the IBM ESD site (www-306.ibm.com/software) are:

- Pack for X12
- Pack for EDIFACT
- Pack for TRADACOMS
- Pack for ODETTE
- Pack for EANCOM

  You will need your **User ID** and **Password** to login.

Example files

EDIFACT and X12 example files are installed in the following directory when you install the Packs for EDI for X12 or EDIFACT:

*install_dir***packs\EDI_v***n.n**

where *install_dir* represents the installation directory, and *n.n* indicates the current Pack for EDI version.

For details on these example files, see the EDIFACT Type Trees documentation and the Pack for X12 documentation.

Chapter 2. What is an EDI type tree?

An EDI type tree includes definitions of objects used in EDI. Using these type trees, you can map directly from a data source or to a data target containing EDI data. You do not need to learn yet another format to map to and from EDI, as you do with many other translator products.

The largest object in an EDI type tree is a transmission. A transmission may include many interchanges from or to many trading partners.

ANSI X12 data

The basic business document in ANSI X12 data is called a transaction set. Transaction sets are enclosed in an envelope that separates one transaction set from another (**ST-SE** envelope).

Groups of transaction sets that are functionally related are enclosed in a functional group envelope (**GS-GE** envelope). The functional group envelope separates one functional group from another. Functional groups that come from the same trading partner are grouped together in an interchange envelope (**ISA-IEA** envelope). A series of interchanges from a variety of trading partners forms a transmission.

Envelopes

Each envelope in EDI data begins with a particular segment and ends with a particular segment. For example, the transaction set envelope begins with an **ST** segment and ends with an **SE** segment.

Transaction sets

Transaction sets are made up of segments and loops. A segment contains a unit of information, for example, a line item or a patient record. Each segment begins with a unique initiator that distinguishes it from all other segments. A segment also has a terminator that tells when the segment ends.

A loop is a repeating pattern of segments and other loops.

Segments

Segments are made up of data elements and composites. A data element is the basic unit of an EDI transaction. Data elements are the items of **EDISegments**, which are delimited and a delimiter separates their components. The value of this delimiter appears in the interchange envelope header, the **ISA** segment.

A composite is a group of related data elements.

For further information on X12 type trees, see the Pack for X12 Type Trees documentation.

EDIFACT data

EDIFACT data is similar to ANSI X12 data. The difference is that the business document in EDIFACT data is called a message and a loop in EDIFACT data is called a group. The EDIFACT standard includes many composites, whereas their use in ANSI X12 is less extensive.

For further information on EDIFACT type trees, see the EDIFACT Type Trees documentation.

TRADACOMS data

The following is an example of TRADACOMS data:

```
STX=ANA:1+5013546009111:AB TRADING COMPANY   LIMITED+5013546002222:SMITHS
LIMITED+920106:175200+1+GANDALF+INVTES+B'
MHD=1+INVFIL:6'
TYP=0700+INVOICES'
SDT=5013546009227:9397706+AB TRADING COMPANY    LTD+HILLSBOROUGH
WORKS:LANGSETT ROAD:SHEFFIELD:SOUTH YORKSHIRE:S6 2LW+172482067'
CDT=5013546002693+SMITHS LTD+(9397706):INTWOOD ROAD:CRINGLEFORD ,
NORWICH:NORFOLK.:NR4 6XB'
FIL=1+1+920106'
FDT=920106+920106'
MTR=7'
MHD=2+INVOIC:8'
CLO=5013546002693::533589/651+SMITHS LTD+SMITHS LTD:
127 CRAIGHALL RD:PORT DUNDAS:GLASGOW'
IRF=28138501+920106+920106'
PYT=2.50% - 30 DAYS MONTHLY ACCOUNT    '
ODD=1+6516923:::911210+03431901:920106'
ILD=1+1+:240100752+++1::M2+51:45900:M+81338:M2+3733400+S+17500
+++COOLAG STANDARD ROOFBOARD:*75mm x 750mm x 1200mm+++120500+1797559+32500'
STL=1++S+17500+1+37334+++++37334+933+36401+6370+43704+42771'
TLR=1+37334+++++37334+933+36401+6370+43704+42771'
MTR=9'
MHD=3+INVOIC:8'
CLO=5013546002693::533589/141+SMITHS LTD+SMITHS LTD:COPPS ROSD:
FLEETWOOD:LANCASHIRE:NR4 6XB'
IRF=06138502+920106+920106'
PYT=1+2.50% - 30 DAYS MONTHLY ACCOUNT    '
ODD=1+1413315:::920102+03803601:920103'
ILD=1+1+:320200026+++1::M+28:28000:M+114545:M+3207300+S+17500
+++MASTERBOARD DOOR FACING:6mm x 2135mm x 915mm+++147800+931140+22500'
STL=1+S+17500+1+32073+++++32073+802+31271+5472+37545+36743'
TLR=1+32073+++++32073+802+31271+5472+37545+36743'
MTR=9'
MHD=4+INVOIC:8'
CLO=5013546002693::533589/698+SMITHS LTD+COLLECTION'
IRF=04138503+920106+920106'
PYT=2.50% - 30 DAYS MONTHLY ACCOUNT    '
ODD=1+6986217:::920106+03852501:920106'
ILD=1+1+5012061000641:100153265+++1::M2+12:104880:ZZ+14000:M2+
1468300+S+17500+++FIBREGLASS DRITHERM:75mm x 455mm x
1200mm (16 pp)+++14000+0+0'
STL=1+S+17500+1+14683+++++14683+367+14316+2505+17188+16821'
TLR=1+14683+++++14683+367+14316+2505+17188+16821'
MTR=9'
MHD=5+VATTLR:6'
VRS=1+S+17500+84090+81988+14347+98437+96335'
MTR=3'
```

```
MHD=6+INVTLR:5'
TOT=84090+81988+14347+98437+96335+3'
MTR=3'
END=6'
```

The TRADACOM standard is primarily used in England and trading partners throughout Europe. TRADACOMS messages are used for domestic trade within the UK and they cover a range of commercial transactions plus reports and master files. There are twenty-six published files; these messages are updated as required to meet changing business needs. The current version of the standard is 9.

EDI version release type trees

The data for each EDI version is defined in a separate tree. For example, the **ansi3070.mtt** tree defines ANSI X12 data version 3070. Each EDI type tree is arranged in a similar way, so you can easily define your specific EDI trading relationships.

Types in EDI trees

The type names in EDI trees are consistent with the terminology used in the EDI standards. The root of each tree is called **EDI**. Each tree has an **ANSI** or **EDIFACT** category, an **Interchange** category and a **Transmission** category.

You can combine versions and standards by selecting one tree and copying or merging other versions or standards to it. You can create a tree, which includes separate definitions for each of your trading partners by customizing a tree for one partner and then copying or merging types with definitions of other partners.

An EDI version type tree includes the entire data dictionary for that particular version and the transmission and interchange objects common to all versions.

The type names of data elements are abbreviated versions of the element descriptions. For example, the name of the type that defines the data element Account Type Code is **Acc'tTypeCd**. If you find these or other names inappropriate for your use, simply rename the type - every reference to that type is renamed for you automatically. For a list of abbreviations used in EDI trees and their corresponding full name, see "**EDI type tree abbreviations**."

The data objects of a particular standard are located under a category with the version name, for example, **V3050** in the **ansi3050.mtt** type tree. The names of particular elements, segments, transaction sets, and messages do not change from tree to tree unless the standard, itself, has changed. This naming convention makes it easier to migrate from one version to the next, if the standard has not changed.

Modifying EDI trees

Make your own copy of each EDI tree you are likely to use. For example, if you are using ANSI 3050 data, open the **ansi3050.mtt** type tree and choose **Save As** from the **File** menu. Rename this type tree something new, for example, **my3050.mtt**. You can then modify your own version tree. If you inadvertently delete something you need, or later decide to add more, open the original tree and copy the missing information to your own tree.

You should customize your EDI tree to suit your trading needs. For information on modifying your tree to include only those types you need, see ″**Creating industry subsets**″.

Chapter 3. Analysis of EDI trees

Before an EDI version tree is released, the tree is analyzed using the Type Tree Analyzer. Some trees produce analysis errors, because of the way data objects are defined in EDI. Most of the analysis errors were resolved by making changes to the tree. These changes eliminate the analysis errors, but keep the EDI definitions intact.

A small number of errors cannot be resolved because resolving them requires knowledge of the user's specific data structure.

This chapter explains changes that were made to the EDI trees and the analysis errors that occur when you analyze each EDI tree. It also explains what to do about these errors.

Summary of errors found

In some cases, the size of an element code, which is defined as an item restriction, was greater than the maximum allowable size for that item. In these cases, the restriction was deleted:

ANSI Tree	Element Type
ansi2003.mtt	AmendmentTypeCd Element
ansi2003.mtt	CommodityGeoLogicalConnectorCd Element
ansi2040.mtt	AmendmentTypeCd Element
ansi3030.mtt	ActionCd306 Element

Note: The type trees listed in the table above are installed in the directory:

*install_dir***packs\EDI_v***n.n****x12\trees**

Analysis uncovered a few transaction sets whose definitions are ambiguous. These transaction sets were not removed from the type tree. However, if you use any of these transaction sets, they must be modified in accordance with how you use it.

In cases where it was possible to change the definition of a transaction set to make it unambiguous, we changed it. Please verify the change before using that transaction set. If a solution was not possible, without information on the user's data structure, the transaction set was left in its ambiguous state.

Note: If you do not use a transaction set that causes an analysis error, delete it from the tree. You can always copy it again from the original EDI tree if you begin to use it.

Here is a list of the transaction sets that cause analysis errors:

Tree	Transaction Set	Analysis Error
ansi2003.mtt	**#830**	optional segments not distinguishable
ansi2040.mtt	**#830**	optional segments not distinguishable

Tree	Transaction Set	Analysis Error
ansi3010.mtt	**#830**	optional segments not distinguishable
ansi3020.mtt	**#110**	optional segments not distinguishable
	#838	data objects of same component not distinguishable
	#426	optional segments not distinguishable
ansi3030.mtt	**#838**	data objects of same component not distinguishable
	#861	optional loops not distinguishable
	#304	optional segments not distinguishable
	#404*	blocked loops not distinguishable
	#417*	blocked loops not distinguishable
ansi3060.mtt	**#304**	optional segments not distinguishable
ansi3070.mtt	**#304**	optional segments not distinguishable
ansi4010.mtt	**#304**	optional segments not distinguishable
ansi4020.mtt	**#304**	optional segments not distinguishable

* The type trees were changed so these errors do not occur. Explanations of how the trees were changed are found later in this chapter.

X12 transaction sets that did not pass analysis

When the definition of a transaction set is in error, you will get one error for the inbound transaction and one for the outbound one. For example, there are two errors in the **ansi2003.mtt** tree concerning the **LoopLIN** of the **#830** transaction set, one for inbound and one for outbound.

The inbound and outbound errors are the same. In our discussion, only the inbound errors are addressed.

Delete X12 transaction sets in error if not used

If you do not use a transaction that causes an analysis error, delete both occurrences of it, that way, when you analyze your tree, you will not get the errors, and the analysis will be faster. You can delete the entire transaction set category under the **Inbound** category and the entire transaction set category under the **Outbound** category. In addition, delete the **Funct'lGroups**, under **Inbound** and **Outbound**, that contain the deleted transaction. Then, analyze the tree again.

For example, the **#838** transaction set in the **ansi3030.mtt** type tree causes an analysis error. If you are using ANSI version 3030, and you are not using the **#838**, you can delete the entire **#838** category for the transaction set, under **Inbound** and under **Outbound**.

Also, delete the **Funct'lGroup** type **#838**, under **Inbound** and under **Outbound**.

For information on customizing your own EDI type tree, see ″**Creating industry subsets**″.

If you *do* use a transaction set that causes an analysis error, please read the explanation of the errors below. Then determine how you want to fix it.

ANSI2003 - the #830 transaction set

After you analyze the **ansi2003.mtt** type tree, the following analyzer message appears: L199 - COMPONENT 4 is not distinguishable from COMPONENT 7 that may follow in TYPE 'LoopLIN #830 Inbound Partner Set V2003 ANSI EDI' (error).

In the **#830** transaction set type, there is a **LoopLIN** type. Within the **LoopLIN** type, there is a **LoopSLN** type. **LoopSLN** ends with an optional **PID Segment**. However, there is also a **PID Segment** later in the component list of the **LoopLIN**. In between, the **PO3 Segment** and the **CTP Segment** are optional. This means that if a **PID Segment** appears in the data, it may be the one in the **LoopSLN** or the one in the **LoopLIN**.

Possible solutions for making the transaction set unambiguous include the following:

- Make at least one occurrence of the **PO3 Segment** or **CTP Segment** required, make its component range minimum at least 1.
- Remove one of the **PID Segment** components.

ANSI2040 - the #830 transaction

The same error that occurred in the **#830** transaction set in ANSI version 2003 also occurred in ANSI version 2040. See the explanation for ANSI2003.

ANSI3010 - the #830 transaction

The same error that occurred in the **#830** transaction set in ANSI version 2003 also occurred in ANSI version 3010. See the explanation for ANSI2003.

ANSI3020 - the #110 transaction

The **#110** transaction set causes two errors.

The first analyze error defines indistinguishable components with optional components in between the **SL1 Segments** in the **ansi3020.mtt** type tree and may read as follows: L199 - COMPONENT 14 is not distinguishable from COMPONENT 17 that may follow in TYPE 'Transaction #110 Inbound Partner Set V3020 ANSI EDI' (error).

The 14th and 17th components of the **#110** transaction set are the **LoopLX** and the **SL1 Segment**.

The problem is an ambiguous **SL1 Segment**; there is one in the actual transaction set. It could be confused with the one nested in the **LoopLX** within the **LoopL5**.

You will have to modify the definition according to your own specifications. Here is a possible solution for making the transaction set unambiguous:

- Remove one of the **S1 Segment** components.
- Make either the **L4 Segment** or **L3 Segment** components required by changing the component range minimum to at least 1.

The second analyze error for the **ansi3020.mtt** type tree **#110** transaction set also identifies indistinguishable components:

The 14th and 18th components of the **#110** transaction set are the **LoopLX** and the **L10 Segment**.

This problem is similar to the first error for this transaction. The **L10 Segment** in the actual transaction set and the one nested within the **LoopLX** are indistinguishable because optional components fall in between.

Modify the definition according to your own specifications. Here is a possible solution for making the transaction set unambiguous:

Remove one of the **L10 Segment** components.

- Make either the **L3**, **L4**, or **SL1** components outside of **LoopL5** and **LoopL1** required.

ANSI3020 - the #838 transaction

The **ansi3020.mtt** analyze error **L201** is different from the **L199** error. For example, the following **L201**message indicates that in the 4th component you cannot distinguish between one occurrence and the next occurrence: *L201 - Different data objects of COMPONENT 4 are not distinguishable in TYPE 'LoopPLA #838 Inbound Partner Set V3020 ANSI EDI' (error). In this case, in the 4th component you cannot distinguish between one occurrence and the next occurrence.* In this case,

The **LoopN11** in the **LoopPLA** begins with an **N1 Segment**. Nested within it is another **N1 Segment**. The **Nested N1 Segment** is optional and all of the components that follow it are optional. If a second **N1 Segment** appears in the data, is it the beginning of another **LoopN11**? Or, is it the **N1 Segment** of the **LoopTUD**?

Note: If you use this transaction set, the best way to fix it is to call your trading partner and ask how it is to be interpreted.

ANSI3060, ANSI3070, ANSI4010, and ANSI4020 - the #304 transaction

The L199 analyzer error occurs when the **ansi3060.mtt**, **ansi3070.mtt**, **ansi4010.mtt**, and the **ansi4020.mtt** are analyzed. The **#304** transaction set in ANSI version 3060 also occurs in ANSI version 3070, 4010, and 4020. The **LoopPO42** contains an **N9 Segment**, which is indistinguishable from the **NP** following the **LoopPO42**.

Modify the definition according to your own specifications. Remove one of the **N9 Segment** components or make one of the optional components in between the indistinguishable **N9 Segment**s required.

Chapter 4. Creating industry subsets

This chapter provides instructions for modifying one of the EDI version trees to create an industry subset. An industry subset is a type tree that contains only those functional groups that are of interest to your specific industry or company. Functional group definitions contain the transaction set definitions that can appear in random order in an **Interchange**. In addition to enveloping **Segments**, the version-specific partitioned functional group types are the only components of the **Interchange**.

If you are going to map directly from an industry subset, you probably have one of the following EDI environments:

- All of your partners use the same subset.
- You plan to use a partner profile table to lookup appropriate partner information.
- You plan to use Trading Partner PC to lookup the appropriate partner information through DDE.
- You plan to use Trading Partner EC to route appropriate partner information.

Benefits

Using a subset tree, particularly for EDI input, improves mapping run-time performance, as there are fewer functional group partitions to test during validation. In addition, the size (disk space required) of the type tree source file can be significantly reduced (minimized).

Data validation of a type tree starts at the top, and moves down the list of **F####** subtypes, attempting validation of each subtype against the data until a match is found.

Other sections in this guide provide examples of mapping to and from EDI using an industry subset.

Use the following procedures to ″prune″ a version tree to meet your needs.

Making a copy of the standard EDI type tree

Copying the standard EDI type trees allows you to change the copy without modifying the original type trees.

To copy the standard EDI type tree

1. Open the EDI type tree from which you want to create an industry subset, such as **ansi3050.mtt**.
2. From the **File** menu, choose **Save As**.
3. For the **File name** field, enter a new name for the type tree.

   For example, **my3050.mtt.**
4. Click **Save**.

Removing unnecessary partner Funct'lGroups

Note: In this procedure, only the required functional group types are retained. In addition, when you map using the EDI tree, only the necessary functional group types appear in the Map Designer. In the example below, the assumed industry is healthcare and only the functional groups containing healthcare transaction sets are kept in the type tree.

To remove unnecessary Inbound and Outbound Partner Funct'lGroups

1. In the newly-created type tree, select the functional group **F** type that is a subtype of **Inbound Partner Funct'lGroup**.

   For example, select the type **F3050** in the **ansi3050.mtt** tree.
2. Right-click the type and choose **Expand All Subtypes** from the context menu.
3. Right-click on the functional group type **F** and choose **Select All Subtypes**.

   All functional group subtypes are selected (highlighted).
4. For each functional group to be retained in the subset tree, locate the subtype for that functional group.

   For example, locate the type **#276** under **F3050** for functional group **HR** that contains 276 transaction sets.
5. Hold down the CTRL key, and then click each subtype you want to retain.

   All the items to be deleted remain selected.
6. After all functional group subtypes that should remain in the subset tree are no longer selected, press the **Delete** key (or choose **Delete** from the **Type** menu).
7. Repeat Steps 1 through 4 for the **F** type under **Outbound Partner Group**, delete the functional groups you do not want.

Creating the target type tree

The type tree created in this procedure is the target type tree that will contain the industry subsets.

To create a new type tree, and name the root type EDI

1. From the **File** menu, choose **New**.
2. A new type tree is created in a new window with the **Root** type.
3. From the **Type** menu, choose **Properties** to display the type properties.
4. In the **Name** field, change the name of the root type from **Root** to **EDI**.
5. From the **File** menu, choose **Save**.
6. Enter a name for the type tree, for example, **hc3050.mtt** (**hc** for healthcare), and click **OK**.

Merging the transmission type

The types retained in the type tree are merged into the target type tree to create the industry subsets.

To merge the Transmission type to the new type tree

1. Right-click the **Transmission** type in the tree you modified in ″**To remove unnecessary Inbound and Outbound Partner Funct'lGroups**″ , then select **Merge** from the context menu.

2. When the Merge Type dialog appears, enable the **Merge Sub-Tree** check box (make sure it is checked).
3. Click anywhere in the new type tree that you created in ″**To create a new type tree, and name the root type EDI**″.
4. When the name of the new tree appears in the **To Tree** box, click **Merge**.
5. Click **Close**.

   Note: Merging may take a few moments.

When you merge **Transmission** to the new tree, all of the types referenced by **Transmission** and types in its sub-tree are copied to the new tree. Only the necessary functional groups, transaction sets, segments, composites and elements are copied to the new tree.

Analyzing the new industry subset type tree

This procedure ensures that your new type tree is valid.

To analyze and save your new industry subset type tree

1. From the **Tree** menu, choose **Analyze → Logic Only**.

   The Analyze Tree dialog box displays the Analysis **Complete** and the **Task Completion** information.
2. If there are analysis errors, correct the errors, and save the type tree.
3. Analyze and correct the type tree until there are no errors.
4. From the **File** menu, choose **Save**.

Adding functional groups to an industry subset tree

After creating an industry subset tree, it may later be desirable to add additional functional groups to that tree. Use the following procedure to do this:

To merge additional functional groups to industry subset type tree

1. Open your target industry subset tree you want to add functional group(s) to, such as **hc3050.mtt**.
2. Open the standard EDI type tree for the version that the industry subset tree was created from, such as **ansi3050.mtt**.
3. In the standard EDI type tree (such as **ansi3050.mtt**), right-click the desired functional group type under the functional group **F** type for the appropriate direction (**Inbound** or **Outbound**) under **Partner Funct'lGroup**.

   For example, the type **#277 F3050 Inbound** in the **ansi3050.mtt** type tree.
4. Select **Merge** from the context menu.
5. The Merge Type dialog box appears with the selected functional group type in the **From** field.
6. Click anywhere in your target industry subset type tree (such as **hc3050.mtt**) to define where the functional group type is being added.
7. The subset type tree name appears in the **To** tree field of the Merge Type dialog box.
8. Click **Merge**.
9. Click **Close**.

10. Repeat Steps 3 through 6 for each additional functional group to be added to the industry subset tree.
11. Close the EDI standard type tree.

Analyze, correct errors if necessary, and save your updated type tree.

Chapter 5. Making a multi-version tree

In order to support different trading partners, it is often necessary to create a type tree that represents multiple EDI version releases. For example, one partner is sending you **#850** purchase orders from ANSI version 3040 and another partner is sending you **#850** purchase orders from ANSI version 3050. This chapter provides instructions for modifying one of the EDI type trees to create a multi-version type tree.

Creating a multi-version tree

In the example that follows, two industry subset trees are used. They are both for the healthcare industry, for ANSI versions 3040 (**hc3040.mtt**) and 3050 (**hc3050.mtt)**.Create an industry subset tree for each ANSI version you use. See ″**Creating industry subsets**″. for instructions on creating industry subset trees. Save this multi-version type tree as a new file. This new type tree is the target type tree that will contain your multi-version data definitions.

Saving a multi-version tree

To save your multi-version tree as a new type tree file

1. Open the industry subset tree containing the latest EDI version.

   For example, if your industry subset trees are **hc3050.mtt** and **hc3040.mtt**, open the **hc3050.mtt** type tree.
2. From the **File** menu, choose **Save As**.
3. Enter a new name for the multi-version type tree.

   For example, **multiver.mtt**, and click **Save**.

Merging the functional group types to a multi-version tree

To merge the functional group types to the multi-version tree

In this procedure, all types referenced by the functional group types are copied to the multi-version tree.

1. Open another industry subset tree that contains the version information you want to use.

   For example, **hc3040.mtt**.
2. In this industry subset tree, right-click the **F####** functional group type (**Funct'lGroup**) under **Inbound**.

   For example, right-click **F3040**.
3. From the context menu, choose **Merge**.

   The Merge Type dialog box appears with the subset tree type to be merged in the **From** field.
4. Enable the **Merge sub-tree** check box.
5. Click anywhere in the multi-version tree to select it as the target.

   The name of the multi-version tree appears in the **To tree** field of the **Merge Type** dialog box.
6. Click **Merge**.

7. Click **Close**.

   The additional inbound functional group types, and all the types referenced by the inbound functional groups are merged into the multi-version tree.
8. Repeat steps 2 through 6, but select the functional group **F** type under **Outbound**.

   The multi-version tree now includes the necessary types from the industry subset tree, the functional group types, elements, composites, segments and transaction sets.

If you want to include more versions in the multi-version tree, repeat steps 1 through 7 using another industry subset tree.

Always analyze and save your type trees after modification.

Chapter 6. Creating a multi-partner tree

You may have the need to create a multi-partner type tree. This allows you to specify particular partners and/or group partners by business relationships, such as customer, distributor or supplier. This chapter provides step-by-step instructions for creating a multi-partner type tree.

Creating an individual partner

In the example that follows, a healthcare industry subset tree, **hc3050.mtt**, is used to create the multi-partner tree.

To create a base multi-partner tree from an industry subset or multi-version tree

1. Open an industry subset tree or multi-version type tree to use as the basis for your multi-partner tree.

   For example, open the **hc3050.mtt** tree.
2. From the **File** menu, choose **Save As**.
3. Enter a new name for the multi-partner type tree file.

   For example, **multip.mtt**.
4. Click **Save**.

This new type tree (**multip.mtt**) can now be modified for your trading partner.

Modifying the multi-partner tree

To modify the multi-partner tree for the first partner

1. From the **Edit** menu, choose **Replace**.

   The Replace dialog box appears.
2. In the **Find what** field, enter `Partner`.
3. In the **Replace with** field, enter the name of a partner you are trading with.

   For example, if you are trading with **CompanyA**. You will rename each **Partner** type with the name **CompanyA**.
4. Enable the **Match whole word only** check box.

   This option ensures that only the types named **Partner** are changed, and not any other type where the word **Partner** is part of a different type name (for example, **ISAPartnerInfo**).
5. Click **Replace All**.
6. After the type names are replaced, click **Close**.

   Each occurrence of the type **Partner** is replaced by the new type name, **CompanyA** in this example.
7. Enter appropriate inbound ISA control information for each trading partner

   Note: When creating a multi-partner tree, you will need to specify each partner's unique sender/receiver identification information in a component rule.

You can enter this information using one of the following methods:

- Assign the values in the component rule
- Use the DDEQUERY function to request the information from Trading Partner PC
- Use the RUN or EXIT function in a component rule to get the information

The identification-related ISA control information is needed in order to distinguish between multiple partners in the same type tree.

Entering inbound ISA control information

To enter appropriate inbound ISA control information for the trading partner

1. Open, by double-clicking the group window for the partner-specific inbound ISA segment.
2. Enter a component rule on **ISAPartnerInfo**, which specifies (at least), the values of the **Sender Interchange IDQual'r Element** and the **InterchangeSenderID Element**.

   For example, if **CompanyA**'s ID qualifier is 00 and the **InterchangeSenderID** is 313488, enter the component rule:
3. Save the type tree.

To enter appropriate outbound ISA control information for the trading partner

This procedure is similar to the procedure titled ″**To enter appropriate inbound ISA control information for the trading partner**″ . Follow these instructions but use the **Outbound ISA Segment**.

Adding another partner

The multi-partner tree is your target tree to add other **Partner Transmission** types.

To merge the Partner types to the multi-partner tree

1. Open the multi-partner tree and the industry subset tree on which you originally based the multi-partner tree.

   For example, open the **multip.mtt** type tree and the **hc3050.mtt** type tree.
2. Merge each of the **Partner Transmission** types from the industry subset tree to the multi-partner tree.

   For example, select **Partner X12 Outbound Transmission** in the **hc3050.mtt** tree. Then choose **Merge** from the **Type** menu, and click on the **multip.mtt**. All of the types that the **Transmission** references are copied.
3. Analyze the multi-partner type tree to make sure all components are connected properly.
4. Save the type tree.
5. Follow the procedures for creating an individual partner section for the new partner (if you want to add another partner). Then, do the steps 1 through 3, above, for the new partner.

Creating business-related partners

Suppose you want to create a multi-partner tree. You have multiple customers and multiple suppliers. You divide your partner definitions into **Customer** and **Supplier**. Then define each partner under the appropriate type, **Customer** or **Supplier**.

Creating a multi-partner tree

Create a multi-partner tree that defines your partners by business category. However, do not enter ISA information.

For example, you create a multi-partner tree with the definitions of a **Customer** partner and a **Supplier** partner. Name the tree **business.mtt**.

To partition each ISA type

1. Right-click an **Inbound ISA Segment** subtype, and choose **Properties** from the context menu.
2. For the **Partitioned** value, select **Yes**. The icon for the selected type changes to a diamond u, indicating that the type is now partitioned.
3. Repeat steps 1 and 2 for the remaining subtypes of **Inbound ISA Envelope Control** and **Outbound ISA Envelope Control**.

To add individual partner subtypes

To add individual partner subtypes, refer to the following steps. As appropriate, add individual partner subtypes under the partitioned subtypes of Inbound ISA and Outbound ISA.

1. Select a subtype of **Inbound ISA Segment** or **Outbound ISA Segment**.
2. From the **Type** menu, choose **New**.
3. Enter the name for the new partner subtype.

   For example, suppose you have two customer partners from whom you receive EDI data, ShopMart and BusyBee. You create the types **ShopMart** and **BusyBee** under **Customer Inbound ISA Segment**.
4. Repeat steps 1 through 3 for each partner-specific type.

After you create the partner-specific ISA types, your ISA types might look something like this:

Add ISA information to the individual partners. For instructions on adding ISA information, see the procedures in the earlier section, ″**Creating an individual partner**″.

Analyze, correct errors if necessary, and save your updated type tree.

Chapter 7. Making a multi-standard tree

In order to support requirements for different partners, it is often necessary to create a type tree that represents multiple standards. This would be the case, for instance, if some partners use ANSI X12 and others use EDIFACT. To meet this need, this chapter provides instructions for modifying one of the standards trees to create a multi-standard tree (for instance, a type tree containing both ANSI X12 and EDIFACT).

Creating a multi-standard tree

For the purposes of this illustration, we will create a multi-standard tree containing ANSI X12 version 3050 and EDIFACT version 91.1.

To create a base multi-standard tree from an industry subset or multi-version tree

1. Open an industry subset or multi-version tree.

   For example, open the **hc3050.mtt** tree.
2. From the **File** menu, choose **Save As**.
3. Enter a new name for the multi-standard tree.

   For example, **multistd.mtt**.
4. Click **Save**.

To copy the new standard-specific types into the multi-standard tree

1. Open the type tree representing the other standard you want in your multi-standard tree.

   For example, open the **edif91_1.mtt** tree.
2. Drag the standard category to the root of the multi-standard tree.

   For example, drag the **EDIFACT** category from the **edif91_1.mtt** tree to the root of the **multistd.mtt** tree. This copies all of the EDIFACT version 91.1 data objects to the multi-standard tree.
3. Drag each standard type under the **Interchange** and **Transmission** categories to the multi-standard tree.
4. Save the multi-standard tree.
5. Follow steps 1 through 4 for each standard you want to add to the multi-standard tree.

Always analyze and save your type trees after modification.

Chapter 8. Mapping inbound EDI data

If you intend to map inbound EDI data, you will need:

- A type tree for your output data, typically the definitions of data to be processed by your application.
- An EDI type tree for your input data.
- Specifications for producing the output you want to generate.
- Some EDI test data to test your map.

Overview

An EDI map is a map that has at least one source or target that contains EDI data. If an input contains EDI data, the map is typically called an inbound map. If an output contains EDI data, the map is typically called an outbound map. You may have both an EDI input and an EDI output, this is often called a turn-around map.

EDI data is usually contained in a file going to or coming from a particular communications system. If you use a direct communication line, you will send or receive a transmission containing interchanges from just one partner. If you use a VAN, you may receive or send a transmission containing multiple interchanges for multiple partners.

An inbound executable map will have at least one input card whose type is in the **Inbound Transmission** sub-tree. An outbound executable map will have at least one output card whose type is in the **Outbound Transmission** sub-tree.

The main difference between inbound and outbound EDI mapping is that outbound mapping requires coordination of partner-specific information (e.g., ID numbers, enveloping, control numbers, etc.). This type of information is typically not part of application data.

Note: The examples in the following sections assume that the EDI data conforms to the ANSI X12 standard. Each executable map in the examples that follow have an input card whose type is **Partner X12 Inbound Transmission EDI** or an output card whose type is **Partner X12 Outbound Transmission EDI**.

Mapping inbound partner data

If you are just beginning to use EDI, you may be trading with one partner who happens to be one of your customers. Customer ABC sends you just one type of transaction purchase orders of ANSI X12 version 3020.

In this case, you know exactly what is in your inbound EDI data file. If you want to map the EDI data to a file containing purchase orders for your application, you could simply map each **Transaction #850** to one of your own purchase orders (Pos).

If more of your customers realize that you can trade electronically, you can accept purchase orders from more partners. Luckily, they all trade using the same version and they all send similar information within their purchase orders.

Adding envelope information to your application

If you want to incorporate EDI partner identification in your application data, you can modify your executable map rule so that envelope information is mapped. You simply add the **ISA Segment** as an argument to the functional map **Map#850ToPO**.

```
=Map#850ToPO (Transaction #850 Inbound Partner Set
V3020:.:EDIPOs,
              Partner Inbound ISA Segment Control
              ANSI:.:EDIPOs)
```

The functional map has the ISA information pertaining to the **Transaction #850**, and it can be mapped to your purchase order file.

Cross-referencing envelope information

Often EDI partner identification is not the identification method used in your application.

In this case, you need to modify your map to look up your application's partner identifier based on the EDI partner identification information.

You need the following items:

- A cross-reference data source that contains the EDI partner identification, and the corresponding customer identification code that is used within your application.
- A type tree that defines that cross-reference data.

Then, you add an input card to your executable map, which represents the cross-reference data.

Add this cross-reference table as another argument to the functional map:

```
=Map#850ToPO (Transaction #850 Inbound Partner Set
            V3020:.:EDIPOs,
            Partner Inbound ISA Segment Control
            ANSI:.:EDIPOs,
            CrossReferenceTable)
```

In the functional map, **Map#850ToPO**, you have three input cards: one for a **Transaction #850**, one for the **Partner Inbound ISA** data, and one for the cross-reference table.

Within the functional map, you would then create a rule using the `LOOKUP` function to get the internal partner identifier that corresponds to the **EDI Partner Identification** in the **Partner Inbound ISA**.

Different rules for different partners

As you expand your trading relationships, you may need to modify your maps to account for partner-specific differences.

For example, when trading with multiple customers, you may find that the information you need appears in a certain segment for one partner and in a different segment for another partner. In this case, you want to specify a different mapping rule for each partner. This can be done in various ways:

- If the differences by partner are manageable, you can use conditional logic in your mapping rules. For example, the map rule below uses the `IF` function and two different functional maps, **GetInfoFromNTE** and **GetInfoFromBEG**:

```
PO# = IF (Partner = "ShopMart",
     GetInfoFromNTE (NTE Segment IN Transaction) ,
     GetInfoFromBEG (BEG Segment IN Transaction))
```

- You can use the same inbound data file as input to multiple maps, one map for each partner. This is advisable if you receive small inbound files.
- You can create a single map with multiple output cards, one output card file per partner. This is an alternative if you have a small number of partners.
- You can use the `RUN` function to execute different maps for different partners. This is useful if you have a large number of partners.

After you get going, you will want to add different applications, for example, purchase orders, health claims, telephone bills, and so on. Partners may want to trade using different versions. How you split your inbound data will depend on the complexity of your trading relationships.

Chapter 9. Mapping outbound EDI data

If you intend to map outbound EDI data, you will need:

- A type tree for your input data, typically, the definitions of application data that you want to transform to EDI.
- An EDI type tree for your output data.
- A type tree for partner EDI information, in particular, to keep control numbers coordinated.
- The specifications that tell you what input information you want to send to one or more of your trading partners.
- Some application test data to test your map.

Setting up a partner profile

In an effort to manage the coordination of partner-specific information, you will probably need a ″partner profile″ data file that contains trading partner information such as ISA envelope identifiers and control numbers.

The **profile.mtt** type tree is a template for an EDI partner profile data file and is installed in the following directory:

*install_dir***packs\EDI_v***n.n****x12\examples\ansi**

This file is a template only and can be used in its current form or changed to meet your specific requirements.

The map source file **partner.mms** is also installed in this directory. In **partner.mms** you can use the map **Partner** to enter EDI partner control information. The map does not have an input card, it only has an output card, where you hard-code the data values into the map rules.

Note: This **Partner** map defines data for two partners.

To create another partner

1. In the **To** card, right-click on **Trading Partner(s)** and choose **Add Index** from the context menu.
2. Expand the new **Trading Partner[3]** group and enter your own partner information.

In addition to the information contained in this template, you may want to keep a separate file for GS information or expand the example file definition to fit your needs.

If you already have partner information in some other form, use it instead. You need to define the partner data in a type tree.

Mapping outbound envelopes

After you have some means of getting at EDI partner information, how you use it depends on how your input data is organized and how many interchanges you want to generate.

In mapping outbound EDI data, you must make decisions as to the number of each type of envelope you will be building in your EDI output file. Will there be more than one interchange? How many functional groups will there be in each interchange? How many transaction sets will there be in each functional group?

Creating a single interchange

The simplest scenario involves making one interchange. Suppose your input data contains purchase orders going to your partner Ron. Ron's partner information is in the partner profile file.

In this scenario, the type of your first output card would be **Partner X12 Outbound Interchange EDI**.

To specify the map rules for the **ISA** and **IEA** segments, expand each segment and enter a rule on each element.

Another option would be to use a functional map to create the **ISA** segment. You may decide to use a functional map, so that you look up the partner-of-interest only once.

In this case, the functional map, **MakeAnISA**, would have a single input card of type **Trading Partner** and a single output card of type **Partner Outbound ISA Envelope Control ANSI Data**. Then, in **MakeAnISA**, all of the elements in the **ISA** would simply be dragged-and-dropped from the **Trading Partner** input card.

Note the rules used on the **IEA** segment, assuming you are generating a single functional group within this interchange. Notice that the rule for the **InterchangeControl# Element** in the **IEA** references the **InterchangeCtrl# Element** in the **ISA**, which appears earlier in the same output card.

Creating multiple interchanges

Your input data may contain data for more than one partner. In this case, you need to create more than one interchange in your output file; you want to create at least one interchange for each partner.

Creating a single functional group envelope

This example deals with just one application, purchase orders. Each interchange will include just one functional group containing all of the purchase orders for the particular partner. To generate one functional group, index one occurrence of the functional group by right-clicking on **Outbound Partner Funct'lGroup ANSI(s)** and selecting **Add Index** from the context menu.

Expand the indexed functional group, expand the **F3020** type, and then expand the **#850** functional group. To generate the **GS Segment**, use the functional map **MakeAGS**. Use the `LOOKUP` function to get the appropriate GS partner information.

Expand the **GE Segment** and enter a rule for each of its components. The value of the **TSIncl Element** should be all of the **#850** transaction sets in the functional group. You can shorten the rule by using the reserved word `IN`. The word `IN` will include all occurrences contained within the output card. Here is the rule on the **TSIncl Element**:

```
= COUNT (Transaction #850 Outbound Partner Set V3020 IN
Interchange)
```

The value of the **GroupCtrl# Element** in the **GE** should be the same as the value of the **GroupCtrl# Element** in the **GS**. Select the rule cell for the **GroupCtrl#** in the **GE**. Then expand the **GS** component and drag and drop the **GroupCtrl#** in the **GS** into the rule cell.

Creating multiple functional group envelopes

You may need to generate multiple functional groups. You may know how many functional groups you want to generate or the number you want to generate may be based on the input data.

When the number of functional groups is known

If you are sending more than one functional group within the same interchange, you may know which application data is in the input. For example, purchase orders, invoices, and so on. In this case, you would index the number of groups you need to produce.

Then, you would proceed to make each application's functional group envelope as specified in "**Creating a single functional group envelope**" .

When the number of functional groups is dependent on the data

If the number of functional groups you want to generate is dependent on the input data, you would use a functional map to generate the functional groups.

For example, the **Header** may include data from different applications, purchase orders as well as purchase order acknowledgments and ship notices. It may be advisable to map the different kinds of application data to different functional groups. Use a functional map for each of the different functional groups in the output.

If more than one functional group is generated within the same interchange, the control numbers must be sequenced.

Using the following example,

```
GroupCtrl# Element [Last] IN Interchange + 1
```

the expression evaluates to 1 if it is the first 810 functional group being produced; it evaluates to 2 for the second 810 functional group. This technique assumes you want to generate group control numbers relative to the current interchange control number. For example, the first group's control number is 1, the second is 2, and so on.

To use an absolute numbering scheme (similar in concept to how interchange control numbers are maintained by partner), you could select the next available control number from the partner profile.

Creating transaction sets

To generate transaction sets, use a functional map. For example, to generate each #850 transaction set, use functional map **Make#850**.

Each transaction set has its own envelope. The transaction header contains the transaction code (for example, 850 for purchase orders) and a control number.

The transaction set control number is typically a relative control number that starts at 1 for the first transaction set and increments by 1 for each successive transaction set within the same functional group. However, you may use any control number scheme you wish.

To increment the transaction set control number by 1, make one of the arguments to the functional map the expression `INDEX($)`.

Note: The actual **TSCtrl# Element** is defined as text. If you want to use **TSCtrl# Element** as the input card type in the **Make#850** map, use the `TEXT` function on `INDEX($)` to convert the `INDEX` value to text.

This rule will generate a **Transaction #850** for each **POHeader Record** whose **Dealer Field** matches the **Dealer** of interest. Each time the **Make#850** map is evaluated, the last argument is evaluated to the index of the current **Transaction #850**. For example, the first time the rule evaluates, `INDEX($)` evaluates to 1. The second time the rule evaluates, `INDEX($)` evaluates to 2, and so on.

In the **Make#850** map, the evaluated result of `INDEX($)` can be mapped to the transaction control number.

The **InclSegments Element** in the **SE Segment** needs to be the count of all segments generated within the transaction set. To calculate this element, you would use the following rule, where **#850TransactionSet** is the name of the card:

```
COUNT (Segment IN #850TransactionSet) + 1
```

Segment is a partitioned type. When you use the `COUNT` function on a partitioned type, all of the types in its sub-tree are counted. All subtypes of **Segment** will be counted. You add 1 to the count to account for the **SE Segment**, which will not yet be generated at the time the rule is evaluated.

Chapter 10. Sending functional acknowledgments

This chapter explains the **audit997.mms** example map source file provided in the following directory:

*install_dir***packs\EDI_v***n.n****x12\examples\ansi\ansiack**

The **audit997.mms** maps an audit log file, generated from inbound ANSI EDI data, to ANSI functional acknowledgments (#997 transaction sets).

Using the 997 transaction set

An ANSI X12 functional acknowledgment (**#997** transaction set) acknowledges the receipt of valid and invalid functional groups within an ANSI X12 interchange. The ANSI X12 functional acknowledgment (**#997** transaction set) may also be used to acknowledge the receipt of valid and/or invalid transaction sets within a functional group.

Generating the 997

Input EDI data may include multiple ANSI X12 EDI interchanges. The example map provided generates EDI data that acknowledges the contents of the input EDI data, according to the ANSI X12 requirements. You want to generate:

- One outbound **#997** functional group for each inbound functional group.
- One **#997** transaction set in each outbound **#997** functional group.

Note: According to ANSI X12, you should not acknowledge inbound 997s, thereby preventing an endless cycle of interchanges.

Each inbound interchange for a trading partner who uses **#997**s will produce a corresponding outbound interchange. That outbound interchange contains one **Group** for each inbound **Group**. Within each **Group**, there is just one **Transaction #997 Set**.

Control numbers

When you send or receive EDI data, three envelopes have control information that you may want to track. There is one control number associated with an interchange, one control number for each functional group within that interchange, and one control number for each transaction set within a functional group.

Interchange control numbers

When you send interchanges to a trading partner, the interchange envelope has a control number that is used to uniquely identify the interchange. That control number is an integer that typically increments by one each time you send an interchange to that particular partner. As long as the control number is unique, you can track each interchange sent to that partner.

For example, you send **#850** transaction sets and **#997** transaction sets to the same trading partner. On Monday, you send your first interchange containing **#850**s to that partner. Suppose its control number has the value 1. On Tuesday, you receive an invoice (a **#810** transaction set) from that partner. You send back a functional

acknowledgment (a **#997** transaction set) in an interchange envelope whose control number has the value 2. Next, you send another **#850** enveloped in an interchange envelope whose control number has the value 3, and so on.

Functional group control numbers

Each functional group contained within an interchange also has a control number. Here, you may assign the functional group a control number that is independent from the interchange control number. This technique would allow you to track functional groups independently from the interchange they are contained in. For example, you may set up one set, or a range of sets, of unique functional group control numbers for **#850**s and another for **#856**s.

You may also choose to start at the value 1 and increment by 1 within an interchange; you would be able to track functional groups relative to the interchange in which they were contained. For example, the fifth functional group within interchange number 100 could be uniquely tracked.

Transaction set control numbers

Each transaction set within a functional group also has a control number. You can choose a scheme that fits your needs to track these. Typically, trading partners use a number relative to the functional group in which they are contained.

Functional acknowledgments and control numbers

When you send a **#997** acknowledgment there is no means of identifying the inbound interchange that is associated with the acknowledged **Group**. This can be handled in the following different ways:

- You could establish a trading relationship so that your partner sends only one **Group** per interchange and that partner uses the same control number for both the **Interchange** and the **Group**. This is a common way to use functional acknowledgments.
- Another alternative is to use unique control numbers for each type of functional group. For example, the first interchange to partner A has 3 functional **PO** groups, numbered 1, 2, and 3. The second interchange to partner A has 2 functional **PO** groups, numbered 4 and 5.
- Suppose your trading partner uses **Group** control numbers that are relative to an absolute interchange control number. Since you are sending one acknowledgment per inbound interchange, you could use the **TA1 Envelope Control** to acknowledge the interchange. However, this is not a common way to use the **TA1**, which is generally used by networks.

You may have another scheme that makes sense to you.

Restart attribute

The ability to map valid objects and ignore invalid objects depends on how the restart attributes are assigned to components of a group type. For example, you receive a transmission. A **Transmission** contains a series of interchanges. For a generic trading partner using the ANSI standards, **Interchanges** are independent objects, they probably come from different trading partners. If you want to be able to accept and map valid data from one trading partner and differentiate the valid from invalid interchanges, you will need to use the restart attribute.

If you are mapping data of a component that has the restart attribute, only the valid occurrences of that component are mapped. For example, if the restart attribute is assigned to the **Interchange** component of a **Transmission**, and an invalid **Interchange** is encountered, the data processing continues. If each

interchange in your input data is independent from the others, it makes sense to ignore the error if an invalid interchange exists.

In general, use the restart attribute on an optional component with a range maximum of (s) that is followed by a required component. This reduces processing time if there is invalid data in the data stream. An exception to this rule is applying the restart attribute for a component if the data could not possibly belong to another type, such as for the interchange component of an EDI transmission.

Restart attributes assigned to the X12 type trees

In each ANSI X12 type tree, the restart attribute has been assigned to certain components. Within a **Transmission** type, there is a restart on an **Interchange** component. Within an **Interchange** type, there is a restart on the **Funct'lGroup ANSI** component. You can keep the restart attribute on the components as they are. You can also add the restart attribute to other components.

Adding a restart at the transaction level

As shown in the **x12_val.mtt** example type tree below, you may want to use a restart attribute on the transaction set component of each functional group you receive. If you are sending out **#997**s, assign the restart attribute to a transaction set component if you want to acknowledge the valid transaction sets within that functional group.

Generating functional acknowledgments from an audit log file

To generate functional acknowledgments to send back to the partners that sent the EDI data, you can generate an audit log file. The audit log file indicates the contents of the EDI data. Then you can map the audit log data to EDI data containing functional acknowledgments.

Audit997 map

The **audit997.mms** installs in the following directory:

*install_dir***\packs\EDI_v***n.n***\x12\examples\ansi\ansiack**

The **audit997.mms** map source file contains an example of mapping from an audit log file to a transmission containing functional acknowledgments.

It is recommended to map from an audit log, rather than mapping directly from the inbound EDI data. The audit log provides a detailed description of data errors, down to the level of segments and elements, which can be mapped to the outbound functional acknowledgments.

The **audit997.mms** map contains four executable maps:

- AnsiAck
- x12_ack
- x12_val
- x12_valgen

Note: Before running the **audit997.mms**, build these four maps.

The **AnsiAck** map uses the `RUN` function to invoke other maps, which do the processing as follows:

- Output card #1 **RUN_x12_val** generates an audit log based on the specific audit settings in the **x12_val** map.

As specified by the map rule shown below, the name of the generated file is **poout4.txt.audit** and the name of the additional rejected data log is **poout4.txt.reject**.

- Output card #2 **RUN_x12_valgen** has a map rule with the `IF` function that specifies if there are errors in the input (a non-zero status is returned) then the map **x12_valgen** is run to generate a second audit log file named **poout4.txt.audgen**.
- Output card #3 **RUN_x12_ack** runs map **x12_ack** which reads the audit information from the output of the first two cards and generates the x12 functional acknowledgment output **poout4.txt.997**.

AdapterSource input file

The input to this map is the **X12_InputName.txt** text file. The functional acknowledgments are generated for the X12 input data file specified in the **X12_InputName.txt** file. The following example illustrates the **po1.txt** data file.

To generate the functional acknowledgments, the four example maps must be built and run.

To generate functional acknowledgments

1. Build the **x12_val**, the **x12_valgen**, and **x12_ack** maps.
2. Confirm that the **AdapterSource** input file **X12_InputName.txt** is in the proper location and has the correct name.
3. Build and run the **AnsiAck** map.

   Note: Only the X12 transactions and versions that are defined in the **x12_val2.mtt** type tree can be validated and processed by this example. Any unrecognized data is written to the rejected data log generated by output card #1 **RUN_x12_val** of the **AnsiAck** map.

Data audit settings

The data audit settings specify what information about the data to include in the audit log file. To audit a particular object, drag that object into the Audit Settings window.

When you audit a data object, you specify what information to include in the audit log file. You specify how to track the object, what information about its detail to include, and when to include item data. Information appears in the audit log file according to what you have selected for the map audit settings.

From the **Map** menu, choose **Organizer** to display the Organizer dialog. The **Data Audit Settings** tab displays the data audit settings used to generate the audit log file.

You can use the example when you want to map your inbound EDI data to EDI data containing functional acknowledgments.

Copying audit settings

The audit settings in the **x12_val** map are generic; many of the object names contain `ANY`. The audit settings can be used to audit EDI data that has been defined by practically any EDI type tree. Use these audit settings to generate an audit file for your EDI data by copying them to a map you have created.

To copy the Audit Settings to your EDI map

1. Open the **audit997.mms** map source file.
2. Select the **x12_val** map.
3. From the **Map** menu, choose **Copy Data Audit Settings**.

   The Copy Data Audit Settings dialog box appears. The **Map file** is the map source file (**.mms**) that contains the map to which you want to copy the audit settings.
4. Click browse to locate the map source file from which you wish to copy the data audit settings.
5. In the **Map Name** list, select the map that generates the audit log file.

   For example, select **MakeTheAuditLog**.
6. Click **OK**.

The audit settings are copied to **MakeTheAuditLog** in **edi_data.mms**.

Now you can build and run the map to generate the audit log file. Then, create another executable map that maps the audit log file to a transmission containing functional acknowledgments, as in the example map **x12_ack**.

Chapter 11. Mapping purchase orders to invoices

This chapter explains the example map that maps an ANSI X12 EDI transmission containing purchase orders (**#850** transaction sets) to and ANSI X12 EDI transmission containing invoices (**#810** transaction sets). The example illustrates how to use the SDQ segment to split the purchase order data into the appropriate invoices. The **sdq.mms** file installs in the following directory:

*install_dir***packs\EDI_v***n.n****x12\examples\ansi\sdq**

By line item

An SDQ segment problem may occur when mapping from purchase orders by line item to invoices by store. You may have an input transmission that may contain multiple interchanges from your trading partner who sends you EDI purchase orders. A purchase order may be organized so that each line item identifies a set of stores that receive some quantity of the merchandise identified in that line item. Your trading partner wants you to send an interchange for each store. The interchange must include one invoice for each inbound purchase order that references that store. Each invoice includes only the line items relevant to that particular store. You may want to receive invoices, one interchange for each store.

There are two inbound purchase orders that refer to store #0100, so the interchange for that store has two invoices. Each invoice for store #0100 references just those line items of the purchase order that had store #0100 identified in an **SDQ Segment** that was part of that line item.

The executable map, **SDQ**, has two inputs and two outputs. The inputs are an EDI transmission containing purchase orders, and a trading partner profile. The outputs are an EDI transmission containing invoices, and the updated trading partner profile.

Purchase order

The following is an example of one purchase order:

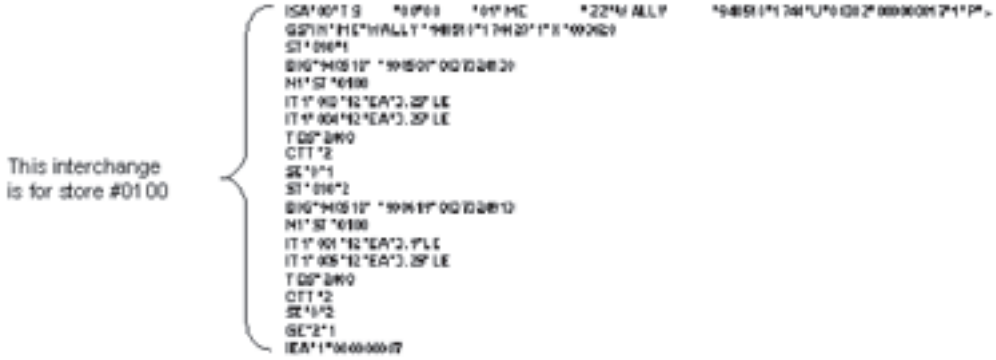
```
ISA*00*          *00*          *08*WALLY          *01*ME             *900616*0500*U*00200*000000002*0*T**
GS*PO*5012738712*18130529*900616*0500*2*X*003020VICS
ST*850*0005
BEG*00*SA*0037324913**900619
REF*DP*32
REF*IA*181305320
ITD*01*7*****30*****          N30
TD5*O****UNITED PARCEL
A PO1*001*150*EA*3.9*LE*UP*070135150611*CB*003234689*VC*1001
  PO4*25*6*EA
  SDQ*EA*92*0100*12*0131*6*0242*6*0548*12*0639*12*0686*6*0749*6*0802*6*1008*6*1027*30
  SDQ*EA*92*1041*6*1044*6*1294*6*1297*12*1800*6*1802*12
B PO1*005*156*EA*3.25*LE*UP*070135160313*CB*003257128*VC*1006
  PO4*26*6*EA
  SDQ*EA*92*0100*12*0131*6*0639*6*0686*6*0749*6*0802*6*0824*6*0847*6*1008*6*1027*30
  SDQ*EA*92*1041*6*1044*6*1241*6*1294*6*1297*12*1392*12*1800*6*1802*12
C PO1*003*144*EA*3.25*LE*UP*070135170312*CB*003257144*VC*1004
  PO4*24*6*EA
  SDQ*EA*92*0242*6*0639*12*0686*6*0749*6*0784*12*0824*6*0847*6*1008*6*1027*30*1041*6
  SDQ*EA*92*1044*6*1241*6*1294*6*1297*12*1800*6*1802*12
D PO1*002*108*EA*3.9*LE*UP*070135152310*CB*003257169*VC*1002
  PO4*18*6*EA
  SDQ*EA*92*0158*6*0639*12*0686*6*0749*6*1008*6*1027*30*1044*6*1294*6*1297*12*1800*6
  SDQ*EA*92*1802*12
E PO1*004*174*EA*3.25*LE*UP*070135140315*CB*003257185*VC*1005
  PO4*29*6*EA
  SDQ*EA*92*0158*6*0520*18*0639*12*0686*6*0749*6*0784*6*0802*6*0824*6*0847*6*1008*6
  SDQ*EA*92*1027*30*1044*6*1241*6*1294*6*1297*12*1305*6*1392*12*1800*6*1802*12
CTT*5
SE*30*0005
```

Reference

	Remarks
A	Send this line item to the 16 stores referenced in the SDQ segments
B	Send this line item to the 18 stores referenced in the SDQ segments
C	Send this line item to the 16 stores referenced in the SDQ segments
D	Send this line item to the 11 stores referenced in the SDQ segments
E	Send this line item to the 19 stores referenced in the SDQ segments

Invoices to be received

The following is an example of the invoices to be received, one interchange for each store:



The store number is indicated by the **IDCd Element** of the **SDQ Segment**. The object is to generate an interchange for each store indicated in the data from your trading partner Wally. For example, if there are a total of three different stores in the data from your trading partner, you want to generate three interchanges.

To map each output interchange, the functional map **ByStore** is used.

The UNIQUE and EXTRACT functions are used to generate one **Interchange** for each unique value of the **IDCd Element** within the **SDQ Segment**. The second argument of **ByStore** is the entire input transmission. To map just the data from your trading partner, the LOOKUP function is used to look up the information in the trading partner profile. The last argument of **ByStore** is the INDEX of the current interchange being produced, which will be used to generate the new ISA control number.

The following displays the component rule:

ByStore map

In the functional map **ByStore**, the #810 transaction set is produced by using another functional map, **ByPO**. Only the #850 transaction sets that have data for the given store are sent to the functional map **ByPO**.

ByPO map

In the functional map **ByPO**, another functional map is called to produce the **IT1 Loops**, the line items of the invoice.

ByLine map

In the map **ByLine**, the quantity invoiced is found in the **SDQ Segment** of the line item in the **PO**.

The **SDQ** map references the **ByStore** functional map. **ByStore** maps the data according to store number, and calls **ByPO**. **ByPO** maps the data according to PO, and calls **ByLine**. Finally, **ByLine** maps the data according to line item.

The Navigator displays the relationships between the maps:

Chapter 12. Mapping inbound invoices

This chapter explains the example map that maps an EDI data file of ANSI X12 3020 invoices (transaction #810) to a file that will be read into your order application. The map source file is **in_inv.mms**, which installs in the following directory:

*install_dir***packs****EDI_v***n.n****x12****examples****ansi****editoinv**

Inbound EDI #810 transaction sets

The input type tree **myedi.mtt** installs in the following directory:

*install_dir***packs****EDI_v***n.n****x12****examples****ansi**

The output tree file **flatinvc.mtt** installs in the following directory:

*install_dir***packs****EDI_v***n.n****x12****examples****ansi****editoinv**

The input type of the executable map is a **Partner X12 Inbound Transmission**. The output type is **File**. The functional map, **MakeInvoice**, will map each #810 transaction set to the output invoice. The map rule for the output component **Invoice(s)** references this functional map, and the input argument is an #810 transaction set from the input side.

In the functional map **MakeInvoice**, the output is a single output invoice, which is made up of one header record and a series of detail records. Each header record was defined, in the Type Designer, as beginning with the initiator `0`, and each detail record was defined as beginning with the initiator `1`.

The header record is made up of three components, each containing name information. The rule for each component takes the **Name Element** from the appropriate **N1** segment in the **#810**, by using the `LOOKUP` function. For example, the rule for **BillTo Field** looks up the **N1** segment where the **EntityIDCd Element** is **BT**. The rules of the three components reference three different **N1** segments:

- one for bill to
- one for remit to
- one for ship to

The rule for the **Detail Record(s)** calls another functional map, **MakeItem**. The input argument for this map is a **LoopIT1** from the input side.

The map **MakeItem** maps a **LoopIT1** to a **Detail Record**. It maps data elements from the **IT1** segment and the first **PID** segment to the item number, quantity and price component of the **Detail Record**.

That completes the mapping process. There are three maps in this example:

- **Invoices (**the executable map)
- **MakeInvoice** (functional map)
- **MakeItem (**functional map)

To build the map

1. Click the **Build map** tool.
2. Click the **Run map** tool.
3. Click **Run Results** .
4. Open the **edi_inv.txt** file to view the input EDI file.
5. Open the **flatinv.txt** file to view the output EDI file.

Chapter 13. Mapping invalid EDI data

This chapter explains the example map that maps inbound EDI invoices containing invalid data to an error report.

Mapping invalid transaction sets

The **invrejct.mms** example map source file installs in the following directory:

*install_dir***packs****EDI_v***n.n****x12****examples****ansi**\ **editoinv****reject**

It is very similar to the **in_inv.mms** map in the **editoinv** folder. The **invrejct.mms** has an additional output card that creates an error report of the invalid transaction sets.

Input type tree

The input tree used in this example is **myedi.mtt**. There is a restart attribute on the transaction set component of each inbound functional group. For example, there is a restart attribute on the **Transaction IE Inbound Partner Set V3020 (s)** component of the inbound **3020 IE** functional group. A **Transaction IE** is either an **#819**, an operating expense statement, or an **#810** transaction set, an invoice.

During the execution of a map, if a data object for a **Transaction IE** does not match the type definition, that object is rejected. Data processing continues and the next **Transaction IE** will be validated.

The output type tree

The output type tree defines the application file of invoices. In addition, it defines an additional output file for rejected data. You want the reject data file to be in a report format, with a separate heading for each invalid transaction set, indicating the **Sender ID**, the group control number and the ISA control number. So, the **ErrorFile** may contain multiple **Error(s)**, and each **Error** contains a heading and one invalid transaction set.

Notice the text item, **Bad810**, which is used to map each rejected #810 transaction set.

Because the `REJECT` function returns a text item, whatever type you use to map rejected data must be defined as a text item that has no maximum length.

Executable map

The executable map is called **InvAudit**. The second output card in the executable map will create the output file of rejected data. On the **Error(s)** component of the **ErrorFile**, the map rule refers to the **MapReject** functional map.

The functional map **MapReject** has three arguments, which are required to produce the **Error** object. The first argument is the `REJECT` of the

Transaction IE. The next argument is the **GS** segment, and the third argument is the **ISA** segment. The functional map, **MapReject**, is evaluated each time an invalid **Transaction IE** occurs.

Audit settings

The audit log is used to track invalid data. After running the map, you can read the audit log to find out information about the invalid data.

To view the audit log

1. From the **Map** menu, choose **Organizer**.
   The Organizer window appears.
2. Click the **Audit Log** tab.
   The audit information displays.

To view the audit settings

1. Select the map in the Navigator.
2. From the **Map** menu, choose **Settings**.
   The Map Settings window appears.
3. Expand the **MapAudit** setting.
   View the settings.

Functional map

The **MapReject** functional map produces an output message that says, `Error in invoice from:` the **InterchangeSenderID Element** of the **ISA**, the group control number, and the **ISA** control number. The last component of the **Error** is the **Bad810**, which is the invalid transaction set.

The input data file in this example contains three invalid transaction sets. The following explains what is wrong.

- The required **TDS** segment is missing.
- The required **LineItems Element** of the **CTT** segment is missing.
- The required **InvDate Element** of the **BIG** segment is not a date; it is text that says `indate`.

The transaction set component of the **IE** functional group has the restart attribute. Each time invalid **Transaction IE** occurs, it is mapped to the error output file. This means an error file that contains 3 invalid transaction sets is produced.

The audit log file indicates each invalid transaction set. For example, the status code E07 indicates that the transaction set is in error because it contains invalid components.

After running the map, you can view the results and see the error file containing the invalid transaction sets.

Mapping invalid line items

Map not only the bad invoices, but the bad line items within those invoices. The `REJECT` function only works on a component that has the restart attribute. To map the bad line items, the first step is to add the restart attribute to the line item component, which is an **IT1Loop**, of the invoice.

Input type tree

The **myedi.mtt** input type tree installs in the following directory:

*install_dir***packs****EDI_v***n.n****x12****examples****ansi**

Add the restart attribute to the **LoopIT1** component of the **#810 Transaction**. To find the **#810** type in the tree, expand the **ANSI** category type, expand the **V3020** category type, and navigate to **Set** → **Partner** → **Inbound** → **IE** → **Transaction** → **#810**.

Note: If you keep the restart attribute on the **LoopIT1**, it may affect how some of your other example maps work. If you don't want it to, remove it after doing this example.

Now there is a restart on both the **Transaction IE** and the **LoopIT1**.

Re-defining the output type tree

Open the **flatinvc.mtt** example type tree in the Type Designer. Save this type tree under a different name, for example **flatinvc2.mtt**.

Note: Do not overwrite the **flatinvc.mtt** type tree, which installed in the directory: *install_dir***packs****EDI_v***n.n****x12****examples****ansi****editoinv**

Now, create the **Transaction** and **LineItem** subtypes under the group type **Error**. The **Transaction** type represents a **Transaction** error and the **LineItem** type represents a **LineItem** error. The definition of a **Transaction Error** does not require changes because it is the same as the **Error** definition in the previous map. Add a few fields that are components of **LineItem Error**.

To determine which transactions contained the bad line item, create an item type **Transaction#** as a subtype of the **Field** category. Create another item type **BadLineItem** to map each rejected line item. **BadLineItem** should have an indefinite maximum size.

Next, add two components to **Error**. Drag and drop **Transaction#** and **BadLineItem** from the tree into the **Error FlatInvoice** group component window. This defines these items as components of **LineItem Error**.

Now edit the components of **LineItem Error**. Delete the inherited component **Bad810** and add **BadLineItem** and **Transaction#** to the component list.

Make a header for each **Transaction Error** and each **LineItem Error**, indicating what kind of error it is. Edit the properties of **Transaction Error** and define the initiator as the text `There is an error in the following invoice:`

Define the initiator for the **LineItem Error** as the text `There is an error in the following line item:`

Define a carriage return after each error header. It's probably easiest to define the initiator of the object following the header, which is **ErrorMsg**. So, define the initiator for **ErrorMsg** as a carriage return/linefeed.

Next, define the components of **ErrorFile** as **Transaction Error(s)** followed by **LineItem Error(s)**.

This completes the definition of the new error file.

Creating a new map source file

Create a copy of the existing **invrejct.mms** file.

To create a copy of the Invrejct map

1. In the Map Designer, open the **invrejct.mms** file, located in the following directory:
   *install_dir***packs\EDI_v***n.n****x12\examples\ ansi\editoinv\ reject**
2. From the **File** menu, choose **Save As**.
3. In the **File name** field, enter **itmrejct.mms**.
4. Edit either output card. For the **TypeTree** setting, choose the tree **flatinvc2.mtt**.
5. When you're prompted to replace all similar tree paths in all maps, choose **Yes**.
6. Click **OK** in the Edit Output Card dialog box to save the changes.

The Unresolved Rules window appears, with an unresolved rule. You want this rule, which was previously on the **Error(s)** component, to now go on the **Transaction Error(s)** component. Drag it from the Unresolved Rules window into the rule cell corresponding to the **Transaction Error(s)** output.

Transaction error functional map

Edit the output card in the **MapReject** map. Change the type from **Error FlatInvoice** to **Transaction Error FlatInvoice**.

The **Transaction** group type displays in the **flatinvc2.mtt** type tree.

The map rules stay in the same cells, so no changes are required.

Add a map rule

Add a map rule to a component in the **InvAudit** executable map. This map rule specifies a functional map that generates each line item error. Name the functional map **MapRejectItem**.

To add a map rule to generate each line item error:

1. Open the **InvAudit** executable map.
2. Enter the following map rule in the rule cell corresponding to the **LineItem Error(s)** component on the **ErrorFile** output card:

```
= MapRejectItem (REJECT
(LoopIT1 IN810:#810<>Transaction IE Inbound Partner Set
V3020:IE<>F3020<>Inbound Partner Funct'lGroup ANSI:Partner X12
Inbound Interchange:TM),
GS Segment V3020:IE<>F3020<>Inbound Partner Funct'lGroup
ANSI:Partner X12 Inbound Interchange:TM,
Partner Inbound ISA Segment Control ANSI:Partner X12 Inbound
```

```
Interchange:TM,
TSCtrl# Element:ST Segment:#810<>Transaction IE Inbound
Partner Set V3020:IE<>F3020<>Inbound Partner Funct'lGroup
ANSI:Partner X12 Inbound Interchange:TM)
```

The **LineItem Error(s)** map rule references the **MapRejectItem** functional map**.** There are four arguments in this map rule: the `REJECT` of the **LoopIT1**, the **GS**, the **ISA**, and the **Transaction** set control number.

Use the Functional Map Wizard to create the **MapRejectItem** functional map.

To create the MapRejectIem functional map:

1. From the **Rules** menu, choose **Functional Map Wizard**.

   The Functional Map Wizard dialog box appears with a yellow icon next to the first input card.
2. Select the **In1** card and click **Edit**.
3. Name the input card **BadIT1Loop**.
4. In the Card Attributes dialog box, select **flatinvc2.mtt** in the **Tree** field and the **BadLineItem** type in the **Name** field.
5. Define a meaningful name for all of the cards. Change the name of the output card from **Out** to **LineItemError**. Name the other input cards **GS**, **ISA**, and **TSCtrl#**.
6. Click **Create** to create the map.

Line item error functional map

In the **MapRejectItem** map created with the functional map wizard, drag and drop the **BadIT1Loop** input into the **BadLineItem** output.

Enter the map rules for the output components.

To use the output of the MapRejectItem functional map

1. Edit the input card in the executable map **InvAudit**.
2. Change the input data file to **edi_inv2.txt**.
3. Build the map by clicking **Build map**.
4. Run the map by clicking **Run map**.

The **edi_inv2.txt** data file contains the following errors:

- The required **QtyInvoiced Element** of the **IT1** segment is missing.
- The required **UOMCd Element** of the **IT1** segment is missing.
- The required **Inv# Element** of the **BIG** segment is missing.

Viewing the results

View the results by clicking **View run results**) icon.

The first thing in the error file is the invalid transaction, which contained the invalid **BIG** segment. The other two errors, invalid **IT1 Segment(s)** in **IT1 Loops** produced two line item errors.

Chapter 14. Tracking your EDI documents

This chapter explains the example maps that track the EDI documents you are sending and receiving. The map source files install in the following directory:

*install_dir***packs****EDI_v***n.n****x12****examples****ansi****editoinv****reject**\ **ansi****track**

The maps use inbound acknowledgments, outbound data, and document control data. The **TrackOut** executable map tracks outbound data interchanges. The **TrackAck** executable map reconciles each inbound acknowledgment with its outbound counterpart.

Your EDI data

If you are sending and receiving EDI data, you may want to do any or all of the following activities:

- Send a functional acknowledgment to a trading partner for an inbound data interchange.
- Track an outbound data interchange that you send to a trading partner.
- Reconcile an inbound acknowledgment with its corresponding outbound data interchange.

A full EDI subsystem would also include other functions:

- Archiving
- Correcting and/or re-sending data
- Creating past due notice reports

You may want additional profile factors, such as an expected time interval to receive acknowledgments.

Note: When running the maps in the track folder, run the **TrackOut** map first. Then run **TrackAck**. If you want to try it again, run the **PurgeSuspense** and **PurgeHistory** maps to reset the data.

Tracking outbound data

The **TrackOut** executable map in the **trackout.mms** map source file serves two purposes. The **TrackOut** map validates the outbound data before it is sent out. This map also demonstrates one of the functions of a document control system.

TrackOut has two inputs. The first input card, **TMOut**, is defined for an ANSI EDI transmission. This data is sent to a trading partner. The second input card, **Profile**, is your trading partner profile. The **Profile** input is used as a look-up file to determine which output consumes the control information extracted from the outbound data.

There are two output cards, **Pending** and **History**. The **Pending** output is stored in a **suspense.txt** file, which is appended with document control information about an outbound interchange for a trading partner who will be sending you a functional acknowledgment. The **History** output is stored in a **history.txt** file, which is appended with document control information about an outbound interchange for a

trading partner who is not sending you a functional acknowledgment. New control information is appended to one, not both, of the output files.

From time to time, you need to purge the **history.txt** file, after archiving the control information. As you test the sample data in the following directory:

*install_dir***packs****EDI_v***n.n****x12****examples**\ **ansi****editoinv****reject**\ **ansi****track**

You may want to use the **PurgeSuspense** and **PurgeHistory** maps that are in the **purge.mms** file.

The map rule on the **V3020 Set(s)** output uses the `IF` and `LOOKUP` functions to see if the partner being sent an interchange will be exchanging acknowledgments with you. If the answer is **Y** (for yes), and you are sending that partner some data conforming to ANSI X12 version 3020, the **Track3020** map is evaluated, and a resulting set of **Pending** records is produced. There is a similar rule for the ANSI 2003 version release set of records. You can adapt this map to another version release, if desired.

After you build and run the **TrackOut** map, the **suspense.txt** output file contains the tracking information.

The first input file **poout.txt** is purchase order data, to be sent to the trading partner, Ron. Look in the **partner.txt** file used for the second input card, to find Ron's partner identifier, **BRADLEY**, and the **Y** value for the **Send997** item.

We are sending two purchase orders to Ron, both enclosed in the same functional group. Ron wants an acknowledgment. The document control information is placed in the **suspense.txt** file. When Ron sends back an acknowledgment, you will remove it from the suspense file, and place it in the **history.txt** file.

The **suspense.txt** file layout is database-ready. There is one record type in the file. Keys that distinguish the data are in the **Key Element** of the **Record**.

The map rules for the **History** data sets are similar. These map rules are used to evaluate the **Track3020** map if the partner to whom the interchange is sent has the value `N` for the **Send997** item in the **Profile Table**. This method provides a record of what was sent to a trading partner.

The **Track3020** map produces a **Record** for each **Transaction** contained in the **Group** that is input to this map. The map rule for the **Record(s)** component shown in the rule bar above identifies the **Record3020** map that is used to evaluate each **Record**. Each **Record** is the same **Key** (from the second and third arguments to the **Record3020** map) and a unique transaction code and control number (from the first argument to the **Record3020** map).

The **Record3020** map is referenced in map rules for both **Pending** and **History** outputs. The map rule shown above produces the value `No Acknowledgment Required` when evaluated from the **History** output rules.

Reconciling inbound acknowledgments

To reconcile inbound acknowledgments with the **Pending** data in the suspense file, use the **TrackAck** executable map in the **trackack.mms** map source file.

The **TrackAck** map uses the inbound transmission file as a look-up file, if any inbound **Group** is a **#997** that matches **Key** data in the **suspense.txt** file, the **history.txt** file is appended with the new control information. The **TrackAck** map also uses the current **suspense.txt** file, and then updates it (see the second output card **Pending**) to remove a **Set of Records** that have been reconciled and placed in the **history.txt** file.

The rule shown in the rule bar below either copies or deletes a **V2003 Set** from the **pending.txt** file.

When the **history.txt** file is appended, the **By3020** or **By2003** map is evaluated whenever there is matching data to be reconciled. The **By3020** map reconciles inbound acknowledgments for trading partners that use the 3020 version release of the ANSI X12 standards. The **By2003** map is similar for those partners that use the 2003 version release.

By3020 updates each record in the **Pending** set (one for each transaction set associated with the functional group we are updating).

The following displays the details of the **History** update, from a view of the **Received3020** map:

This map copies the **Key** and **Sent** information and adds the **Received** information.

Chapter 15. Editobol.mss map

This chapter explains the **editobol.mms** map source file**,** which is installed in the following directory:

*install_dir***packs****EDI_v***n.n****x12****examples****ansi****editobol**

The **editobol.mms** map source file maps an EDI data file of ANSI X12 3020 Motor Carrier Shipment Information (transaction #204) to an application file containing bill of lading information.

Mapping inbound EDI #204 transaction sets

The input type tree file is **myedi.mtt**, and is installed in the following directory:

*install_dir***packs****EDI_v***n.n****x12****examples****ansi**

The output type tree file is **bol.mtt**, and is installed in the following directory:

*install_dir***packs****EDI_v***n.n****x12****examples****ansi****editobol**

The input type of the executable map is **Partner X12 Inbound Transmission**.

The output type is **File**, which is the user's file containing any number of bills of lading. Each bill of lading is composed of one header record and many detail records. Each header record has several fields, including repeating sets of name and address information. These sets have been defined in the type tree as a single object called **NameSet**, which is used three times. The detail record object contains five fields. These are fixed length records, where every field is mandatory and no delimiters are used. Header records are initiated by 01, and detail records by 02.

The output file is made up of **BillOfLading(s)**. The **EachBOL** functional map maps each #204 transaction set to the output bill of lading.

EachBOL functional map

In the **EachBOL** functional map, the output is a single bill of lading, which is made up of one header record and a series of detail records. Each header record type is defined in the Type Designer with the initiator 01, and each detail record was defined as beginning with the initiator 02.

The header record is made up of ten components, three of which contain repeating patterns of name and address information. The rule for each component selects the necessary element from the appropriate segment from the inbound **#204**, or assigns a literal.

In the case of the repeating name and address section of the header record, **NameSet**, the correct occurrence of this loop must be selected to properly populate the user's file with either **ShipTo**, **BillTo**, or **Consignee** name and address data. This has been accomplished by using the `IF` function to test the **N1 Loop**, then calling a functional map to create each component of that **Name Set**. For example, the rule for **NameSet[1]** tests the **N1** segment for an **EntityIDCd Element** of **CN**.

The rules of the three **NameSet** components reference three different **N1** loops: one for **Consignee**, one for **Ship To**, and one for **Bill To**.

MakeNameSet functional map

The **MakeNameSet** functional map has the **N1 Loop** as the input.

Again, in the **EachBOL** map, the rule for the **Detail Record(s)** calls another functional map, **EachDetail**. This map is responsible for creating each individual **Detail Record** within a **BillOfLading**. The input arguments for this map are the **Transaction #204** and the **LoopLX**. From these two areas come the segments that make up the **DetailRecord**.

EachDetail functional map

In the **EachDetail** map, the components of the output **Detail Record** are mapped from the **LoopLX** and the **#204** transaction set.

Chapter 16. EDI type tree abbreviations

The following table lists the reference number of each EDI element, the name of the element, and the abbreviation that is used in the EDI type trees.

Ref#	Element Name	Type Tree Abbreviation
1	Route Code	**RouteCd**
2	Number of Accepted Transaction Sets	**AcceptedTS**
3	Free Form Message	**FreeFormMsg3**
4	Air Carrier Code	**AirCarrierCd**
5	Airport Code	**AirportCd**
7	Bank Account Number	**BankAcc't#**
8	Bank Client Code	**BankClientCd**
9	Late Reason Code	**LateReasonCd**
11	Billing Code	**BillingCd**
12	Payment Pattern	**Pay'tPattern**
13	Booking Number	**Booking#**
14	Carriage Value	**CarriageVal**
16	Charge Method of Payment	**ChrgMthdOfPay't**
19	City Name	**CityName**
20	Client Bank Number	**ClientBank#**
21	Number of Shipments	**Shipments**
22	Commodity Code	**CommodityCd**
23	Commodity Code Qualifier	**CommodityCdQual'r**
24	Equipment Type	**Equip'tType**
26	Country Code	**CountryCd**
28	Group Control Number	**GroupCtrl#**
31	Adjustment Number	**Adj't#31**
32	Delivery Date	**DeliveryDate32**
33	Lading Exception Code	**LadingExceptionCd**
34	Service Standard	**ServiceStandard**
35	Disposition Code	**DispositionCd35**
39	Entitlement Code	**EntitlementCd**
40	Equipment Description Code	**Equip'tDesc'nCd**
41	Interchange Train Identification	**InterchangeTrainID**
42	Block Identification	**BlockID**
43	Error Condition Code	**ErrorCond'nCd**
44	Error Field Data	**ErrorFieldData**
45	ETA Date	**ETADate**

Ref#	Element Name	Type Tree Abbreviation
46	Ex Parte	**ExParte**
47	Export Filing Key Code	**ExportFilingKeyCd**
48	Export License Control Code	**ExportLicenseCtrlCd**
50	Export License Number	**ExportLicense#**
51	Export License Status Code	**ExportLicenseStatusCd**
52	Export License Symbol Code	**ExportLicenseSymbolCd**
54	Risk of Loss Qualifier	**RiskOfLossQual'r**
55	Flight/Voyage Number	**FlightVoyage#**
56	Type of Service Code	**ServiceCd**
58	Charge	**Chrg**
59	Freight Class Code	**FrtClassCd**
60	Freight Rate	**FrtRate**
61	Free-Form Message	**FreeFormMsg61**
62	Hazardous Material Code	**HzrdMat'lCd**
63	Hazardous Material Contact	**HzrdMat'lContact**
64	Hazardous Material Description	**HzrdMat'lDesc'n**
65	Height	**Height**
66	Identification Code Qualifier	**IDCdQual'r**
67	Identification Code	**IDCd**
68	Import License Expiration Date	**ImportLicenseExpir'nDate**
69	Import License Issue Date	**ImportLicenseIssueDate**
70	Import License Number	**ImportLicense#**
71	Inquiry Request Number	**InquiryRequest#**
72	Type of Service Offered Code	**ServiceOfferedCd**
73	Compensation Qualifier	**CompQual'r**
74	Declared Value	**DeclaredVal**
76	Invoice Number	**Inv#**
77	Flashpoint Temperature	**FlashptTemp**
78	Container Type Request Code	**ContainerTypeRequestCd**
79	Lading Description	**LadingDesc'n**
80	Lading Quantity	**LadingQty**
81	Weight	**Wt**
82	Length	**Length**
83	Licensing Agency Code	**LicensingAgencyCd**
86	Total Equipment	**TotalEquip't**
87	Marks and Numbers	**MarksAnd#**
88	Marks and Numbers Qualifier	**MarksAnd#Qual'r**

Ref#	Element Name	Type Tree Abbreviation
90	Measurement Unit Qualifier	**Meas'tUnitQual'r**
91	Transportation Method/Type Code	**Transp'nMthdTypeCd**
92	Purchase Order Type Code	**POTypeCd**
93	Name	**Name**
95	Number of Containers	**Containers**
96	Number of Included Segments	**InclSegments**
97	Number of Transaction Sets Included	**TSIncl**
98	Entity Identifier Code	**EntityIDCd**
100	Currency Code	**CurrencyCd**
102	Ownership Code	**OwnershipCd**
103	Packaging Code	**Pkg'gCd**
104	Type of Bankruptcy Code	**BankruptcyCd**
106	Event	**Event**
107	Payment Method Code	**Pay'tMthdCd107**
108	Pick-up or Delivery Code	**PickupDeliveryCd**
109	Pick-up Date	**PickupDate**
110	Application Acknowledgment Code	**App'nAck'tCd**
111	Pick-up Time	**PickupTime**
112	Pier Name	**PierName**
113	Pier Number	**Pier#**
114	Port Name	**PortName**
115	Port Function Code	**PortFunctionCd**
116	Postal Code	**PostalCd**
117	Prepaid Amount	**PpdAmt**
118	Rate	**Rate**
119	Rate Basis Number	**RateBasis#**
120	Rate Combination Point Code	**RateComb'nPtCd**
121	Rate Class Code	**RateClassCd**
122	Rate/Value Qualifier	**RateValQual'r**
123	Number of Received Transaction Sets	**Rcv'dTS**
124	Application Receiver's Code	**App'nRcv'rCd**
126	Reference Designator	**RefDesignator**
127	Reference Identification	**Ref#**
128	Reference Identification Qualifier	**Ref#Qual'r**
129	Referenced Pattern Identifier	**Ref'dPatternID**
131	Rejected Set Identifier	**RejectedSetID**

Ref#	Element Name	Type Tree Abbreviation
132	Release Code	**ReleaseCd**
133	Routing Sequence Code	**RoutingSequenceCd**
135	Sailing/Flight Date Estimated	**SailingFlightDateEst'd**
136	Sales Comment	**SalesComment**
137	Sales Reference Date	**SalesRefDate**
138	Sales Reference Number	**SalesRef#**
139	Sales Terms Code	**SalesTermsCd**
140	Standard Carrier Alpha Code	**SCAC**
141	Schedule B Code	**SchedBCd**
142	Application Sender's Code	**App'nSenderCd**
143	Transaction Set Identifier Code	**TSIDCd**
145	Shipment Identification Number	**Ship'tID#**
146	Shipment Method of Payment	**Ship'tMthdOfPay't**
147	Shipment Qualifier	**Ship'tQual'r**
148	Lading Value	**LadingVal**
150	Special Charge or Allowance Code	**SpclChrgAllowCd**
151	Authority	**Auth'y**
152	Special Handling Code	**SpclHandlingCd**
153	Special Handling Description	**SpclHandlingDesc'n**
154	Standard Point Location Code	**SPLC**
156	State or Province Code	**StateProvinceCd**
157	Status Code	**StatusCd157**
158	Status Date	**StatusDate**
159	Status Location	**StatusLoc**
160	Status Report Request Code	**StatusRptRequestCd**
161	Status Time	**StatusTime**
163	Stop Reason Code	**StopReasonCd**
165	Stop Sequence Number	**StopSequence#**
166	Address Information	**AddressInfo**
167	Tare Weight	**TareWt**
168	Tariff Agency Code	**TariffAgencyCd**
169	Tariff Item Number	**TariffItem#**
170	Tariff Item Part	**TariffItemPart**
171	Tariff Number	**Tariff#**
172	Tariff Section	**TariffSection172**

Ref#	Element Name	Type Tree Abbreviation
173	Tariff Supplement Identifier	**TariffSupplementID**
174	Terminal Name	**TerminalName**
176	Time Qualifier	**TimeQual'r**
177	Intermodal Service Code	**IntermodalServiceCd**
181	Quantity or Status Adjustment Reason Code	**QtyStatusAdj'tReasonCd**
182	Vessel Name	**VesselName**
183	Volume	**Vol**
184	Volume Unit Qualifier	**VolUnitQual'r**
186	Waybill Number	**Waybill#**
187	Weight Qualifier	**WtQual'r**
188	Weight Unit Code	**WtUnitCd**
189	Width	**Width**
190	Accomplish Code	**AccomplishCd**
191	Advances	**Advances**
192	Agent/Shipper Routing Code	**AgentShipperRoutingCd**
193	Net Amount Due	**NetAmtDue**
195	Capacity Load Code	**CapacityLoadCd**
196	Mortgagor Response Code	**MortgagorRspCd**
197	Mortgagee Information Status Code	**MortgageeInfoStatusCd**
199	Confidential Billing Request Code	**Confid'lBillingRequestCd**
200	Hazardous Materials Page	**HzrdMat'lPage**
201	Business Transaction Status	**BusinessTransc'nStatus**
202	Correction Indicator	**CorrectionIndicator**
203	Cubic Capacity	**CubicCapacity**
204	Direct Store Delivery Sequence Number	**DirectStoreDeliverySequence#**
205	Dunnage	**Dunnage**
206	Equipment Initial	**Equip'tInitial**
207	Equipment Number	**Equip't#**
208	Hazardous Material Code Qualifier	**HzrdMat'lCdQual'r**
209	Hazardous Material Class Code	**HzrdMat'lClassCd**
210	International/Domestic Code	**Int'lDomesticCd**
211	Packaging Form Code	**Pkg'gFormCd**
212	Unit Price	**UnitPrice**
213	Lading Line Item Number	**LadingLineItem#**
214	Waybill Request Code	**WaybillRequestCd**

Ref#	Element Name	Type Tree Abbreviation
215	Hazardous Classification	**HzrdClass'n**
216	Metric Qualifier	**MetricQual'r**
218	Hazardous Placard Notation	**HzrdPlacardNotation**
219	Position	**Pos'n**
220	Billed/Rated-as Quantity	**BilledRatedAsQty**
221	Billed/Rated-as Qualifier	**BilledRatedAsQual'r**
222	Hazardous Endorsement	**HzrdEndorsement**
223	Repetitive Pattern Number	**RepetitivePattern#**
224	Hazardous Material Shipping Name	**HzrdMat'lShip'gName**
225	Seal Number	**Seal#**
226	Section Seven Code	**SectionSevenCd**
227	Tariff Column	**TariffColumn**
229	Transit Registration Number	**TransitReg'n#**
230	Subsidiary Classification	**SubsidiaryClass'n**
231	Cross Reference Type Code	**CrossRefTypeCd**
232	Weight Allowance	**WtAllow**
233	Weight Capacity	**WtCapacity**
234	Product/Service ID	**ProdServiceID**
235	Product/Service ID Qualifier	**ProdServiceIDQual'r**
236	Price Identifier Code	**PriceIDCd**
237	Item List Cost	**ItemListCost**
238	Emergency Response Plan Number	**EmergencyRspPlan#**
240	Car Service Order Code	**CarServiceOrderCd**
241	Protective Service Code	**ProtectiveServiceCd**
242	Vent Instruction Code	**VentInstructionCd**
243	Transaction Reference Date	**Transc'nRefDate**
244	Transaction Reference Number	**Transc'nRef#244**
246	Certification/Clause Code	**Cert'nClauseCd**
247	Certification/Clause Text	**Cert'nClauseText**
248	Allowance or Charge Indicator	**AllowChrgIndicator**
249	Vessel Requirement Code	**VesselReq'tCd**
250	Letter of Credit Number	**LtrOfCredit#**
253	Automobile Ramp Facility Code	**AutoRampFacilityCd**
254	Packing Group Code	**PackingGroupCd**
255	Expiration Date	**Expir'nDate**
256	Manifest Type Code	**ManifestTypeCd**
257	Tariff Application Code	**TariffApp'nCd**

Ref#	Element Name	Type Tree Abbreviation
258	Quantity Cost	**QtyCost**
259	Change Type Code	**ChangeTypeCd**
260	Group Title	**GroupTitle**
261	Source of Disclosure Code	**SourceOfDisclosureCd**
262	Geography Qualifier Code	**GeogQual'rCd**
263	Rating Code	**RatingCd**
264	Census Merchandise Type Code	**CensusMerchandiseTypeCd**
265	Census Export License Identifier Code	**CensusExportLicenseIDCd**
266	Census Statistical Month Code	**CensusStatisticalMonthCd**
267	Net Explosive Quantity	**NetExplosiveQty**
268	Census Container Code	**CensusContainerCd**
269	Census Special Identifier Code	**CensusSpclIDCd**
270	Census Trade Identifier Code	**CensusTradeIDCd**
271	Subsidiary Risk Indicator	**SubsidiaryRiskIndicator**
272	Hazardous Certification Code	**HzrdCert'nCd**
273	Hazardous Certification Declaration	**HzrdCert'nDeclaration**
274	Hazardous Material Classification	**HzrdMat'lClass'n**
275	Authorization Date	**Auth'nDate**
276	Special Charge Description	**SpclChrgDesc'n**
277	UN/NA Identification Code	**UNNAIDCd**
280	Exchange Rate	**ExchangeRate**
281	Carrier Restriction Code	**CarrierRestrictionCd**
282	Terms Start Date	**TermsStartDate**
283	Terms Due Date Qualifier	**TermsDueDateQual'r**
284	Service Level Code	**ServiceLevelCd**
285	Depositor Order Number	**DepositorOrder#**
286	Product/Service Condition Code	**ProdServiceCond'nCd**
287	Authorize/ De-Authorize Code	**AuthorizeCd**
288	Prepriced Option Code	**PrepricedOptionCd**
289	Multiple Price Quantity	**MplPriceQty**
290	Price Condition Code	**PriceCond'nCd**
291	Price Condition Applies Code	**PriceCond'nAppliesCd**
292	Quantity Basis	**QtyBasis**

Ref#	Element Name	Type Tree Abbreviation
293	Promotion Condition Qualifier	**PromoCond'nQual'r**
294	Tariff Distance	**TariffDistance**
295	Distance Qualifier	**DistanceQual'r**
296	Intermediate Switch Carrier	**IntermediateSwitchCarrier**
298	Origin EDI Carrier Code	**OriginEDICarrierCd**
299	Free-form Transit Data	**FreeformTransitData**
301	Car Type Code	**CarTypeCd**
302	Damage Location on Equipment	**DmgLocOnEquip**
303	Type of Damage	**TypeOfDamage**
304	Event Code	**EventCd**
305	Transaction Handling Code	**Transc'nHandlingCd**
306	Action Code	**ActionCd306**
308	Damage Exception Indicator	**DmgExceptionIndicator**
309	Location Qualifier	**LocQual'r**
310	Location Identifier	**LocID**
311	Shipment Type Code	**Ship'tTypeCd**
312	Special Indicator Code	**SpclIndicatorCd**
313	Authority Identifier Code	**Auth'yIDCd**
315	Compensation Paid	**CompPaid**
316	Next Port of Discharge	**NextPortOfDischarge**
317	Total Compensation Amount	**TotalCompAmt**
318	Current Port of Loading	**CurrentPortOfLoading**
319	Temperature Control	**TempCtrl**
320	Scale	**Scale**
321	Intermodal Facility Code	**IntermodalFacilityCd**
322	Load/Empty Status Code	**LoadEmptyStatusCd**
323	Purchase Order Date	**PODate**
324	Purchase Order Number	**PO#**
325	Tax Identification Number	**TaxID#**
326	Request Reference Number	**RequestRef#**
327	Change Order Sequence Number	**ChangeOrderSequence#**
328	Release Number	**Release#**
329	Transaction Set Control Number	**TSCtrl#**
330	Quantity Ordered	**QtyOrdered**
331	Allowance or Charge Method of Handling Code	**AllowChrgMthdOfHandlingCd**
332	Percent	**%332**
333	Terms Basis Date Code	**TermsBasisDateCd**

Ref#	Element Name	Type Tree Abbreviation
334	Transportation Terms Qualifier Code	**Transp'nTermsQual'rCd**
335	Transportation Terms Code	**Transp'nTermsCd**
336	Terms Type Code	**TermsTypeCd**
337	Time	**Time**
338	Terms Discount Percent	**TermsDscnt%**
339	Allowance or Charge Quantity	**AllowChrgQty**
340	Allowance or Charge Code	**AllowChrgCd**
341	Allowance or Charge Number	**AllowChrg#**
342	Percent of Invoice Payable	**%OfInvPayable**
343	Installment Total Invoice Amount Due	**Install'tTotalInvAmtDue**
344	Unit of Time Period or Interval	**UnitOfTimePeriodInterval**
345	Lead Time Code	**LeadTimeCd**
346	Application Type	**App'nType**
347	Hash Total	**HashTotal**
348	Jurisdiction Code	**Juris'nCd**
349	Item Description Type	**ItemDesc'nType**
350	Assigned Identification	**AssignedID**
351	Terms Discount Days Due	**TermsDscntDaysDue**
352	Description	**Desc'n**
353	Transaction Set Purpose Code	**TSPurpCd**
354	Number of Line Items	**LineItems**
355	Unit or Basis for Measurement Code	**UOMCd**
356	Pack	**Pack**
357	Size	**Size**
358	Quantity Invoiced	**QtyInvoiced**
359	Allowance or Charge Rate	**AllowChrgRate**
360	Allowance or Charge Total Amount	**AllowChrgTotalAmt**
361	Total Invoice Amount	**TotalInvAmt**
362	Terms Discount Amount	**TermsDscntAmt**
363	Note Reference Code	**NoteRefCd**
364	Communication Number	**Comm'n#**
365	Communication Number Qualifier	**Comm'n#Qual'r**
366	Contact Function Code	**ContactFunctionCd**
367	Contract Number	**Contract#**

Ref#	Element Name	Type Tree Abbreviation
368	Shipment/Order Status Code	**Ship'tOrderStatusCd**
369	Free-form Description	**FreeformDesc'n**
370	Terms Discount Due Date	**TermsDscntDueDate**
371	Change Reason Code	**ChangeReasonCd**
372	Lading Liability Code	**LadingLiabilityCd**
373	Date	**Date**
374	Date/Time Qualifier	**DateTimeQual'r**
375	Tariff Service Code	**TariffServiceCd**
376	Test Indicator	**TestIndicator**
377	Rounding Rule Code	**RoundingRuleCd**
378	Allowance/ Charge Percent Qualifier	**AllowChrg%Qual'r**
379	Bid Type Response Code	**BidTypeRspCd**
380	Quantity	**Qty**
381	Price Reason Code	**PriceReasonCd**
382	Number of Units Shipped	**UnitsShipped**
383	Quantity Difference	**QtyDiff**
384	Gross Weight per Pack	**GrossWtPerPack**
385	Gross Volume per Pack	**GrossVolPerPack**
386	Terms Net Days	**TermsNetDays**
387	Routing	**Routing**
388	Terms Deferred Due Date	**TermsDeferredDueDate**
389	Deferred Amount Due	**DeferredAmtDue**
390	Amount Subject to Terms Discount	**AmtSubjectTermsDscnt390**
391	Discounted Amount Due	**DscntAmtDue391**
392	Bill of Lading Status Code	**BOLStatusCd**
393	Amendment Code	**AmendmentCd**
394	Warehouse Receipt Number	**WhseRcpt#**
395	Unit Weight	**UnitWt**
396	Shipment Identification	**Ship'tID**
397	Color	**Color**
398	Order Sizing Factor	**OrderSizingFactor**
399	Pallet Exchange Code	**PalletExchangeCd**
400	Unit Load Option Code	**UnitLoadOptionCd**
402	Communications ID	**Comm'nID**
403	Communications Password	**Comm'nPassword**
404	Transmission Control Number	**TransmissionCtrl#**
405	Number of Included Functional Groups	**InclFunct'lGroups**

Ref#	Element Name	Type Tree Abbreviation
406	Quantity of Pallets Shipped	**QtyOfPalletsShipped**
407	Seal Status Code	**SealStatusCd**
408	Temperature	**Temp**
409	Quantity of Pallets Received	**QtyOfPalletsRcv'd**
410	Quantity of Pallets Returned	**QtyOfPalletsReturned**
411	Quantity Contested	**QtyContested**
412	Receiving Condition Code	**Rcv'gCond'nCd**
413	Quantity Received	**QtyRcv'd**
414	Lading Quantity Received	**LadingQtyRcv'd**
415	Rate Adjustment Description Code	**RateAdj'tDesc'nCd**
416	Pallet Block and Tiers	**PalletBlockAndTiers**
417	Price Bracket Identifier	**PriceBracketID**
418	Item List Cost - New	**ItemListCostNew**
419	Item List Cost - Old	**ItemListCostOld**
420	Price New, Suggested Retail	**PriceNewSuggestedRetail**
421	Price Old, Suggested Retail	**PriceOldSuggestedRetail**
422	Promotion Condition Code	**PromoCond'nCd**
423	Promotion Status Code	**PromoStatusCd**
424	Vendor Order Number	**VendorOrder#**
426	Adjustment Reason Code	**Adj'tReasonCd**
427	Unit Price Difference	**UnitPriceDiff**
429	Check Number	**Check#**
432	Date Qualifier	**DateQual'r**
433	F.O.B. Point Code	**FOBPtCd**
434	F.O.B. Point	**FOBPt**
436	Primary Publication Authority Code	**PrimaryPublicationAuth'yCd**
437	Rate Maintenance Authority Code	**RateMaintAuth'yCd**
438	U.P.C. Case Code	**UPCCaseCd**
439	Price List Number	**PriceList#**
440	Price List Issue Number	**PriceListIssue#**
441	Tax Exempt Code	**TaxExemptCd**
442	MICR Number	**MICR#**
443	Contact Inquiry Reference	**ContactInquiryRef**
444	Purchase Order Instruction Code	**POInstructionCd**
445	Terms Exception Code	**TermsExceptionCd**
446	Terms Net Due Date	**TermsNetDueDate**
447	Loop Identifier Code	**LoopIDCd**
448	Property Damage Code	**PropertyDmgCd**

Ref#	Element Name	Type Tree Abbreviation
449	Fixed Format Information	**FixedFormatInfo**
450	In-bond Type Code	**InbondTypeCd**
451	Warehouse Lot Number	**WhseLot#**
452	Quantity Damaged/On Hold	**QtyDamagedOnHold**
455	Responsible Agency Code	**RspAgencyCd**
456	Temperature Probe Location Code	**TempProbeLocCd**
458	Dunnage Description	**DunnageDesc'n**
459	Name (30 Character Format)	**Name30CharFormat**
460	Shipment Weight Code	**Ship'tWtCd**
461	Transit Level Code	**TransitLevelCd**
462	Through Surcharge Percent	**ThruSurcharge%**
463	Paid-In Surcharge Percent	**PaidInSurcharge%**
464	Container Terms Code	**ContainerTermsCd**
465	Container Terms Code Qualifier	**ContainerTermsCdQual'r**
466	Total Stopoffs	**TotalStopoffs**
467	Priority	**Priority**
468	Port Call File Number	**PortCallFile#**
470	Priority Code	**PriorityCd**
471	Priority Code Qualifier	**PriorityCdQual'r**
472	Link Sequence Number	**LinkSequence#**
473	Order Status Code	**OrderStatusCd**
474	Master Reference (Link) Number	**MasterRefLink#**
475	Credit/Debit Adjustment Number	**CreditDebitAdj't#**
477	Credit/Debit Quantity	**CreditDebitQty**
478	Credit/Debit Flag Code	**CreditDebitFlagCd**
479	Functional Identifier Code	**Funct'lIDCd**
480	Version / Release / Industry Identifier Code	**VersionReleaseIndustryIDCd**
481	Trace Type Code	**TraceTypeCd**
482	Payment Action Code	**Pay'tActionCd**
483	Counseling Status Code	**CounselingStatusCd**
484	Evaluation Rating Code	**EvaluationRatingCd**
485	Results Code	**ResultsCd**
486	Product Transfer Movement Type Code	**ProdTransferMovementTypeCd**
487	Maintenance Operation Code	**MaintOperationCd**
488	Percent	**%488**

Ref#	Element Name	Type Tree Abbreviation
489	Loop Level Number	**LoopLevel#**
490	Note Identification Number	**NoteID#**
491	Data Element Type	**DEType491**
493	Tariff Number Suffix	**Tariff#Suffix**
495	Condition Segment Logical Connector	**Cond'nSegLogicalConnector**
496	Level	**Level496**
497	Sub Level	**SubLevel**
498	Condition Code	**Cond'nCd**
499	Condition Value	**Cond'nVal**
500	Rate Level	**RateLevel**
501	Customs Documentation Handling Code	**CustomsDoc'nHandlingCd**
502	Type of Locomotive Maintenance Code	**LocomotiveMaintCd**
503	Block 20 Code	**Block20Cd**
504	Chemical Analysis Percentage	**ChemAnalysis%**
505	Partition Indicator	**PartitionIndicator**
506	(DFI) ID Number Qualifier	**DFIID#Qual'r**
507	(DFI) Identification Number	**DFIID#**
508	Account Number	**Acc't#**
509	Originating Company Identifier	**OriginatingCoID**
510	Originating Company Supplemental Code	**OriginatingCoSupple'tCd**
511	Rail Car Plate Size Code	**RailCarPlateSizeCd**
512	Import/Export Code	**ImportExportCd**
514	Reporting Code	**ReportingCd**
515	Number of Transaction Sets Totalled	**TSTotalled**
516	Total Qualifier	**TotalQual'r**
517	Data Element Totalled	**DETotalled**
518	Total	**Total**
519	Time Period Qualifier	**TimePeriodQual'r519**
521	Product Transfer Type Code	**ProdTransferTypeCd**
522	Amount Qualifier Code	**AmtQual'rCd**
529	Inventory Transaction Type Code	**InventoryTransc'nTypeCd**
531	Agent Shipment ID Number	**AgentShip'tID#**
533	Water Movement Code	**WaterMovementCd**
534	Inland Transportation Code	**InlandTransp'nCd**

Ref#	Element Name	Type Tree Abbreviation
535	Rail Retirement Activity Code	**RailRetirementActivityCd**
536	Nature of Claim Code	**NatureOfClaimCd**
537	Employment Code	**EmploymentCd**
538	Charge/Allowance Qualifier	**ChrgAllowQual'r**
539	Vehicle Identification Number	**VehicleID#**
540	Damage Type Code	**DmgTypeCd**
541	Damage Severity Code	**DmgSeverityCd**
542	Labor Hours	**LaborHours**
543	Labor Rate	**LaborRate**
544	Supporting Evidence Code	**SupportingEvidenceCd**
545	Unemployed Reason Code	**UnemployedReasonCd**
546	Status Code	**StatusCd546**
547	Interest Type Code	**InterestTypeCd**
548	Decline/Amend Reason Code	**DeclineAmendReasonCd**
549	Carriers Delivery Receipt Number	**CarriersDeliveryRcpt#**
550	Total Labor Cost	**TotalLaborCost**
551	Total Miscellaneous Costs	**TotalMiscCosts**
552	Total Repair Cost	**TotalRepairCost**
553	Authorization Identification	**Auth'nID**
554	Assigned Number	**Assigned#**
555	Labor Operation Identifier	**LaborOperationID**
556	Damage Area Code	**DmgAreaCd**
557	Part Name	**PartName**
558	Reservation Action Code	**ReservationActionCd**
559	Agency Qualifier Code	**AgencyQual'rCd**
560	Special Services Code	**SpclServicesCd**
561	Service Marks and Numbers	**ServiceMarksAnd#**
562	Rate or Value Type Code	**RateValTypeCd**
563	Sales Requirement Code	**SalesReq'tCd**
564	Do-Not-Exceed Action Code	**DoNotExceedActionCd**
566	Product/Service Substitution Code	**ProdServiceSub'nCd**
567	Equipment Length	**Equip'tLength**
568	Electronic Form Note Reference Code	**ElectronicFormNoteRefCd**
569	Account Number Qualifier	**Acc't#Qual'r**
570	Scale Type Code	**ScaleTypeCd**
571	Tare Qualifier Code	**TareQual'rCd**

Ref#	Element Name	Type Tree Abbreviation
572	Weight Allowance Type Code	**WtAllowTypeCd**
573	Freight Station Accounting Code	**FrtStationAcc'tCd**
574	Claim Profile	**ClaimProfile**
575	City Name Qualifier Code	**CityNameQual'rCd**
576	Abbreviated Customer Name	**AbbrevCustomerName**
577	Net Tons	**NetTons**
578	Equipment Status Code	**Equip'tStatusCd**
579	Type of Consist Code	**ConsistCd**
580	Amendment Type Code	**AmendmentTypeCd**
581	Customs Entry Type Code	**EntryTypeCd**
582	Bill of Lading Type Code	**BOLTypeCd**
583	Factory Car Order Number	**FactoryCarOrder#**
584	Employment Status Code	**EmploymentStatusCd**
585	Payroll Status Code	**PayrollStatusCd**
586	Request for Quote Reference Number	**RFQRef#**
587	Acknowledgment Type	**Ack'tType**
589	Position in Set	**Pos'nInSet**
590	Wages Paid Code	**WagesPaidCd**
591	Payment Method Code	**Pay'tMthdCd591**
592	Lading Description Qualifier	**LadingDesc'nQual'r**
594	Frequency Code	**FreqCd**
595	Compartment ID Code	**CompartmentIDCd**
597	Vessel Code	**VesselCd**
598	Bill of Lading/Waybill Number	**BOLWaybill#**
599	Manifest Unit Code	**ManifestUnitCd**
600	Place of Receipt by Pre-carrier	**PlaceOfRcptByPrecarrier**
601	Customs Entry Number	**Entry#**
602	Customs Shipment Value	**CustomsShip'tVal**
603	In-bond Control Number	**InbondCtrl#**
604	Consolidation Code	**ConsolidationCd**
605	Deficiency Judgement Code	**DeficiencyJudgementCd**
607	Number of Days	**Days**
608	Credit File Variation Code	**CreditFileVariationCd**
609	Count	**Count**
610	Amount	**Amt**
611	Disposition Code	**DispositionCd611**

Ref#	Element Name	Type Tree Abbreviation
612	Credit Report Merge Type Code	**CreditRptMergeTypeCd**
613	Statement Number	**Statement#**
614	Store Number	**Store#**
615	Time Period Qualifier	**TimePeriodQual'r615**
616	Number of Periods	**Periods**
617	Switch Type Code	**SwitchTypeCd**
618	Implementation Transaction Set Syntax Error Code	**Implement'nTSSyntaxErrorCd**
619	Zone	**Zone**
620	Implementation Segment Syntax Error Code	**Implement'nSegSyntaxErrorCd**
621	Implementation Data Element Syntax Error Code	**Implement'nDESyntaxErrorCd**
622	Number of Loads	**Loads**
623	Time Code	**TimeCd**
624	Century	**Century**
625	COD Method of Payment Code	**CODMthdOfPay'tCd**
626	Excess Transportation Reason Code	**ExcessTransp'nReasonCd**
627	Excess Transportation Responsibility Code	**ExcessTransp'nRspCd**
628	Hierarchical ID Number	**Hier'lID#**
629	Alternation Precedence Code	**AltPrecedenceCd**
630	Minimum/Weight Logic	**MinWtLogic**
631	Number of Rates	**Rates**
632	Rate Application Type Code	**RateApp'nTypeCd**
633	Loading Restriction	**LoadingRestriction**
634	Factor Amount	**FactorAmt**
635	Rate Request/Response Code	**RateRequestRspCd**
636	Sequence Number Qualifier	**Sequence#Qual'r**
637	Unit Conversion Factor	**UnitConversionFactor**
638	Rule 260 Junction Code	**Rule260JunctionCd**
639	Basis of Unit Price Code	**BasisOfUnitPriceCd**
640	Transaction Type Code	**Transc'nTypeCd**
641	Status Reason Code	**StatusReasonCd**
642	Week	**Week**
643	Lading Percentage	**Lading%**
644	Lading Percent Qualifier	**Lading%Qual'r**
645	Related Company Indication Code	**RelatedCoIndicationCd**

Ref#	Element Name	Type Tree Abbreviation
646	Quantity Shipped to Date	**QtyShippedDate**
647	Application Error Condition Code	**App'nErrorCond'nCd**
648	Price Multiplier Qualifier	**PriceMultiplierQual'r**
649	Multiplier	**Multiplier**
650	Rating Remarks Code	**RatingRemarksCd**
653	Discount Terms Type Code	**DscntTermsTypeCd**
654	Discount Base Qualifier	**DscntBaseQual'r**
655	Discount Base Value	**DscntBaseVal**
656	Discount Control Limit Qualifier	**DscntCtrlLimitQual'r**
657	Discount Control Limit	**DscntCtrlLimit**
659	Basis of Verification Code	**BasisOfVerificationCd**
660	Contract Suffix	**ContractSuffix**
662	Relationship Code	**RelationshipCd**
663	Quantity Units Received or Accepted	**QtyUnitsRcv'dAccepted**
664	Quantity Units Returned	**QtyUnitsReturned**
665	Residue Indicator Code	**ResidueIndicatorCd**
666	Division Type Code	**DivisionTypeCd**
667	Quantity in Question	**QtyInQuestion**
668	Line Item Status Code	**LineItemStatusCd**
669	Currency Market/Exchange Code	**CurrencyMktExchangeCd**
670	Change or Response Type Code	**ChangeRspTypeCd**
671	Quantity Left to Receive	**QtyLeftRcv**
672	Resource Authorization Code	**ResourceAuth'nCd**
673	Quantity Qualifier	**QtyQual'r**
674	Reciprocal Switch Code	**ReciprocalSwitchCd**
675	Schedule Type Qualifier	**SchedTypeQual'r**
676	Schedule Quantity Qualifier	**SchedQtyQual'r**
677	Item Depth	**ItemDepth**
678	Ship/Delivery or Calendar Pattern Code	**ShipDeliveryCalendarPatternCd**
679	Ship/Delivery Pattern Time Code	**ShipDeliveryPatternTimeCd**
680	Forecast Qualifier	**ForecastQual'r**
681	Forecast Timing Qualifier	**ForecastTimingQual'r**
682	Part Release Status Code	**PartReleaseStatusCd**
683	Catalog Purpose Code	**CatalogPurpCd**
684	Catalog Number	**Catalog#**

Ref#	Element Name	Type Tree Abbreviation
685	Catalog Version Number	**CatalogVersion#**
686	Catalog Revision Number	**CatalogRev#**
687	Class of Trade Code	**ClassOfTradeCd**
688	Restrictions/Conditions Qualifier	**RestrictionsCond'nQual'r**
689	Occupancy Code	**OccupancyCd**
690	Docket Identification	**DocketID**
691	Revision Number	**Rev#**
692	Conveyance Code	**ConveyanceCd**
693	Docket Type Code	**DocketTypeCd**
694	Percentage Division	**%Division**
695	Independence Code	**IndependenceCd**
696	Action Code	**ActionCd696**
697	Docket Control Number	**DocketCtrl#**
698	Rate Distribution Code	**RateDistrib'nCd**
699	Commodity/Geographic Logical Connector Code	**CommodityGeogLogicalConnectorCd**
701	Information Type	**InfoType**
702	Financial Transaction Code	**FinancialTransc'nCd**
703	Financial Information Type Code	**FinancialInfoTypeCd**
704	Paperwork/Report Action Code	**PaperworkRptActionCd**
705	Trade Union Code	**TradeUnionCd**
706	Entity Relationship Code	**EntityRelationshipCd**
707	Rating Category Code	**RatingCategoryCd**
708	Rating Summary Value Code	**RatingSummaryValCd**
709	Communications Environment Code	**Comm'nEnvironmentCd**
712	Category Reference Code	**CategoryRefCd**
713	Installment Group Indicator	**Install'tGroupIndicator**
714	Goods and Services Tax Reason Code	**GoodsAndServicesTaxReasonCd**
715	Functional Group Acknowledge Code	**Funct'lGroupAckCd**
716	Functional Group Syntax Error Code	**Funct'lGroupSyntaxErrorCd**
717	Transaction Set Acknowledgment Code	**TSAck'tCd**
718	Transaction Set Syntax Error Code	**TSSyntaxErrorCd**
719	Segment Position in Transaction Set	**SegPos'nInTS**

Ref#	Element Name	Type Tree Abbreviation
720	Segment Syntax Error Code	**SegSyntaxErrorCd**
721	Segment ID Code	**SegIDCode**
722	Element Position in Segment	**ElementPos'nInSeg**
723	Data Element Syntax Error Code	**DESyntaxErrorCd**
724	Copy of Bad Data Element	**CopyOfBadDE**
725	Data Element Reference Number	**DERef#**
726	Real Estate Property Condition Code	**REPropertyCond'nCd**
728	Returnable Container Load Make-Up Code	**ReturnContainerLoadMakeUpCd**
729	Category	**Category**
730	Subcategory	**Subcategory**
731	Transit Direction Code	**TransitDirectionCd**
732	Transit Time Direction Qualifier	**TransitTimeDirectionQual'r**
733	Transit Time	**TransitTime**
734	Hierarchical Parent ID Number	**Hier'lParentID#**
735	Hierarchical Level Code	**Hier'lLevelCd**
736	Hierarchical Child Code	**Hier'lChildCd**
737	Measurement Reference ID Code	**Meas'tRefIDCd**
738	Measurement Qualifier	**Meas'tQual'r**
739	Measurement Value	**Meas'tVal**
740	Range Minimum	**RangeMin**
741	Range Maximum	**RangeMax**
742	Route Description	**RouteDesc'n**
743	Returnable Container Freight Payment Responsibility Code	**ReturnContainerFrtPay'tRspCd**
744	Print Option Code	**PrintOptionCd**
745	Pre-Cooled (Rule 710) Code	**PreCooledRule710Cd**
746	Protective Service Rule Code	**ProtectiveServiceRuleCd**
747	Rebill Reason Code	**RebillReasonCd**
748	Movement Authority Code	**MovementAuth'yCd**
749	Supplementary Information Qualifier	**SupplementaryInfoQual'r**
750	Product/Process Characteristic Code	**ProdProcessCharCd**
751	Product Description Code	**ProdDesc'nCd**
752	Surface/Layer/Position Code	**SurfaceLayerPos'nCd**

Ref#	Element Name	Type Tree Abbreviation
753	Packaging Characteristic Code	**Pkg'gCharCd**
754	Packaging Description Code	**Pkg'gDesc'nCd**
755	Report Type Code	**RptTypeCd**
756	Report Transmission Code	**RptTransmissionCd**
757	Report Copies Needed	**RptCopiesNeeded**
758	Hazardous Mnemonic Code	**HzrdMnemonicCd**
759	Reportable Quantity Code	**RptQtyCd**
760	Limited Quantity Indication Code	**LimitedQtyIndicationCd**
761	Equipment Number Check Digit	**Equip't#CheckDigit**
762	Waybill Response Code	**WaybillRspCd**
763	Total Statement Amount	**TotalStatementAmt**
765	Day of Month	**DayOfMonth**
766	U.P.C./EAN Consumer Package Code	**UPCEANConsumerPkgCd**
767	Market Area Code Identifier	**MktAreaCdID**
768	Quantity Must Purchase	**QtyMustPurchase**
769	Exception Number	**Exception#**
770	Option Number	**Option#**
771	Market Area Code Qualifier	**MktAreaCdQual'r**
772	Interchange Agreement Status Code	**InterchangeAgreementStatusCd**
773	Quantity Free	**QtyFree**
781	Statement Format Code	**StatementFormatCd**
782	Monetary Amount	**M'Amt**
783	Planning Schedule Type Code	**PlanningSchedTypeCd**
784	Length of Binary Data	**LengthOfBinaryData**
785	Binary Data	**BinaryData**
786	Security Level Code	**SecurityLevelCd**
787	Record Length	**RecordLength**
788	Block Length	**BlockLength**
789	Drawing Sheet Size Code	**DrawingSheetSizeCd**
790	Entity Title	**EntityTitle**
791	Entity Purpose	**EntityPurp**
792	Entity Status Code	**EntityStatusCd**
795	Revision Level Code	**RevLevelCd**
796	Revision Value	**RevVal**
797	Security Technique Code	**SecurityTechniqueCd**
799	Version Identifier	**VersionID**

Ref#	Element Name	Type Tree Abbreviation
800	Compression Technique	**CompressionTechnique**
801	Interchange Format	**InterchangeFormat**
802	Program Identifier	**ProgramID**
803	File Name	**FileName**
804	Block Type	**BlockType**
805	Canadian Hazardous Notation	**CanadianHzrdNotation**
806	EPA Waste Stream Number Code	**EPAWasteStream#Cd**
807	Waste Characteristics Code	**WasteCharCd**
808	Hazardous Material Shipment Information Qualifier	**HzrdMat'lShip'tInfoQual'r**
809	Hazardous Material Shipment Information	**HzrdMat'lShip'tInfo**
810	Inner Pack	**InnerPack**
811	Obligation Type Code	**ObligationTypeCd**
812	Payment Format Code	**Pay'tFormatCd**
813	Station Type Code	**StationTypeCd**
814	Nesting Code	**NestingCd**
815	Property Inspection Qualifier	**PropertyInspectionQual'r**
816	Occupancy Verification Code	**OccupancyVerificationCd**
817	Tax Information Identification Number	**TaxInfoID#**
818	Name Control Identifier	**NameCtrlID**
819	Language Code	**LanguageCd**
820	Report Section Name Code	**RptSectionNameCd**
821	Safety Characteristic/ Hazard Code	**SafetyCharHzrdCd**
822	Source Subqualifier	**SourceSubqualifier**
824	Security Originator Name	**SecurityOriginatorName**
825	Security Recipient Name	**SecurityRecipientName**
826	Owners Share	**OwnersShare**
827	Promotion Amount Qualifier	**PromoAmtQual'r**
828	Dollar Basis For Percent	**DollarBasis%**
829	Fuel Type	**FuelType**
831	Inspection Location Type Code	**InspectionLocTypeCd**
832	Ramp Identification	**RampID**
833	Automotive Manufacturers Code	**AutoMfrCd**
834	Inspector Identity Code	**InspectorIDCd**

Ref#	Element Name	Type Tree Abbreviation
835	Supplemental Inspection Code	**Supple'tInspectionCd**
836	Vehicle Deck Position Code	**VehicleDeckPos'nCd**
837	Vehicle Type Code	**VehicleTypeCd**
838	Dealer Code	**DealerCd**
839	Bay Location	**BayLoc**
844	Inbound Condition Hold Code	**InboundCond'nHoldCd**
845	Chassis Type	**ChassisType**
846	Contract Status Code	**ContractStatusCd**
847	Order/Item Code	**OrderItemCd**
848	Product/Date Code	**ProdDateCd**
849	Location Code	**LocCd**
850	Status Report Code	**StatusRptCd**
851	Nesting	**Nesting**
852	Address Type Code	**AddressTypeCd**
853	Damage Reason Code	**DmgReasonCd**
854	Vessel Type Code	**VesselTypeCd**
855	Peg Code	**PegCd**
856	Rate Level Qualifier Code	**RateLevelQual'rCd**
857	Pre-Price Quantity Designator	**PrePriceQtyDesignator**
858	Retail Pre-Price	**RetailPrePrice**
859	Activity Code	**ActivityCd**
860	D-U-N-S Number	**DUNS#**
861	Supplier's Delivery/Return Number	**SupplierDeliveryReturn#**
862	Receiver's Location Number	**Rcv'rLoc#**
863	X-Peg	**XPeg**
864	Y-Peg	**YPeg**
865	Total Deposit Dollar Amount	**TotalDepositDollarAmt**
866	Integrity Check Value	**IntegrityCheckVal**
867	Signature	**Signature**
868	Initiator Code	**InitiatorCd**
869	Adjustment Number	**Adj't#869**
870	Receiver Delivery/Return Number	**Rcv'rDeliveryReturn#**
871	Supplier's Location Number	**SupplierLoc#**
872	Physical Delivery or Return Date	**PhysDeliveryReturnDate**
873	Product Ownership Transfer Date	**ProdOwnershipTransferDate**

Ref#	Element Name	Type Tree Abbreviation
874	Space Management Reference Code	**SpaceMngtRefCd**
875	Maintenance Type Code	**MaintTypeCd**
876	Alternate Tiers per Pallet	**AltTiersPerPallet**
877	Vessel Stowage Location	**VesselStowageLoc**
878	Cash Register Item Description	**CashRegisterItemDesc'n**
879	Coupon Family Code	**CouponFamilyCd**
880	Dated Product Number of Days	**DatedProdDays**
881	Deposit Value	**DepositVal**
883	Pallet Type Code	**PalletTypeCd**
884	Pallet Tiers	**PalletTiers**
885	Pallet Blocks	**PalletBlocks**
886	Nonconformance Report Status Code	**NonconformRptStatusCd**
887	Nonconformance Resultant Response Code	**NonconformResultantRspCd**
888	Nonconformance Determination Code	**NonconformDetermnCd**
889	Follow-up Action Code	**FollowupActionCd**
890	Carrier/Route Change Reason Code	**CarrierRouteChangeReasonCd**
891	Shipping Date Change Reason Code	**Ship'gDateChangeReasonCd**
892	Line Item Change Reason Code	**LineItemChangeReasonCd**
893	Warehouse Detail Adjustment Identifier	**WhseDetailAdj'tID**
894	Batch Type Code	**BatchTypeCd**
895	Availability	**Availability**
897	Vessel Code Qualifier	**VesselCdQual'r**
898	Incentive Grain Rate Indicator Code	**IncentiveGrainRateIndicatorCd**
899	Unload Terminal Elevator Code	**UnloadTerminalElevatorCd**
901	Reject Reason Code	**RejectReasonCd**
902	Section Designator	**SectionDesignator**
903	Envelope Indicator	**EnvelopeIndicator**
904	Requirement Designator	**Req'tDesignator**
905	Maximum Use	**MaxUse**
906	Level Number	**Level#**
909	Loop Name	**LoopName**
910	Loop Repeat Count	**LoopRepeatCount**

Ref#	Element Name	Type Tree Abbreviation
911	Position in Segment	**Pos'nInSeg**
912	Relation Code	**RelationCd**
913	Data Element Type	**DEType913**
914	Minimum Length	**MinLength**
915	Maximum Length	**MaxLength**
916	Code List Reference	**CdListRef**
918	Code Value	**CdVal**
921	Discipline Type Code	**DisciplineTypeCd**
922	Electronic Form Standards Type Code	**ElectronicFormStandardsTypeCd**
923	Prognosis Code	**PrognosisCd**
924	Full or Partial Indicator	**FullPartialIndicator**
926	Data Maintenance Number	**DataMaint#**
927	Damage Code Qualifier	**DmgCdQual'r**
928	Special Services Quantity	**SpclServicesQty**
930	Regulatory Agency Code	**RegulatoryAgencyCd**
931	Page Width	**PageWidth**
932	Page Length Lines	**PageLengthLines**
933	Free-Form Message Text	**FreeFormMsgText**
934	Printer Carriage Control Code	**PrinterCarriageCtrlCd**
935	Measurement Significance Code	**Meas'tSignificanceCd**
936	Measurement Attribute Code	**Meas'tAttributeCd**
937	Test Administration Method Code	**TestAdminMthdCd**
938	Test Medium Code	**TestMediumCd**
939	Sample Process Status Code	**SampleProcessStatusCd**
940	Sample Selection Method Code	**SampleSelectionMthdCd**
942	Sample Frequency Value per Unit of Measurement Code	**SampleFreqValPerUOMCd**
943	Sample Description Code	**SampleDesc'nCd**
944	Sample Direction Code	**SampleDirectionCd**
945	Position Code	**Pos'nCd**
949	Confidence Limit	**ConfidenceLimit**
950	Statistic Code	**StatisticCd**
951	Balance Type Code	**BalanceTypeCd**
952	Adjustment Application Code	**Adj'tApp'nCd**
953	Interest Rate	**InterestRate**
954	Percent	**%954**

Ref#	Element Name	Type Tree Abbreviation
955	Tax Jurisdiction Code Qualifier	**TaxJuris'nCdQual'r**
956	Tax Jurisdiction Code	**TaxJuris'nCd**
957	Application Batch Identifier	**App'nBatchID**
958	Application Item Identifier	**App'nItemID**
959	Payment Cancellation Type	**Pay'tCancellationType**
960	Request for Quote Type Code	**RFQTypeCd**
961	Data Element New Content	**DENewContent**
962	Receiving Advice or Acceptance Certificate Type Code	**Rcv'gAdviceAcceptCertTypeCd**
963	Tax Type Code	**TaxTypeCd**
964	Cost Code	**CostCd**
965	Issuing Carrier Identifier	**IssuingCarrierID**
967	Tray Count	**TrayCount**
969	Rate Basis Qualifier	**RateBasisQual'r**
970	Tariff Add-On Factor	**TariffAddOnFactor**
972	Tariff Class Adjustment Reference	**TariffClassAdj'tRef**
973	Tariff Item Suffix	**TariffItemSuffix**
974	Tariff Reference Flag	**TariffRefFlag**
975	Tariff Restriction Description	**TariffRestrictionDesc'n**
976	Tariff Restriction ID Code	**TariffRestrictionIDCd**
977	Tariff Restriction Value	**TariffRestrictionVal**
978	Tariff Section	**TariffSection978**
980	Tariff Section ID Code	**TariffSectionIDCd**
981	Tariff Value Code	**TariffValCd**
982	Data Source Code	**DataSourceCd**
983	Hazardous Class Qualifier	**HzrdClassQual'r**
984	Hazardous Material Shipping Name Qualifier	**HzrdMat'lShip'gNameQual'r**
985	N.O.S. Indicator Code	**NOSIndicatorCd**
986	Special Commodity Indicator Code	**SpclCommodityCd**
987	Cryptographic Service Message (CSM) Message Class Code	**CSMMsgClassCd**
988	Cryptographic Service Message (CSM) Field Tag	**CSMFieldTag**
989	Cryptographic Service Message (CSM) Field Contents	**CSMFieldContents**
990	Security Type	**SecurityType**

Ref#	Element Name	Type Tree Abbreviation
991	Authentication Key Name	**Authent'nKeyName**
992	Authentication Service Code	**Authent'nServiceCd**
993	Encryption Key Name	**EncryptionKeyName**
994	Encryption Service Code	**EncryptionServiceCd**
995	Length of Data	**LOD**
996	Initialization Vector	**IV**
997	Hash or Authentication Code	**HashAuthent'nCd**
998	Delayed Repayment Qualifier Code	**DelayedRepaymentQual'rCd**
1000	Service Characteristics Qualifier	**ServiceCharQual'r**
1002	Car Hire Detail/Summary Code	**CarHireDetailSummaryCd**
1003	Account Type Code	**Acc'tTypeCd**
1004	Percent Qualifier	**%Qual'r**
1005	Hierarchical Structure Code	**Hier'lStructureCd**
1006	Account Description Code	**Acc'tDesc'nCd**
1007	Rate Source	**RateSource**
1008	Case Type Code	**CaseTypeCd**
1009	Court Type Code	**CourtTypeCd**
1010	Cycle Month Hours	**CycleMonthHours**
1011	Association of American Railroads (AAR) Pool Code	**AARPoolCd**
1012	Court Event Type Code	**CourtEventTypeCd**
1013	Notice Type Code	**NoticeTypeCd**
1014	Car Type Group Code	**CarTypeGroupCd**
1015	Mileage Settlement Code	**MileageSettlementCd**
1016	Penalty Code	**PenaltyCd**
1017	Claim Type Code	**ClaimTypeCd**
1018	Exponent	**Exponent**
1019	Invoice Type Code	**InvTypeCd**
1020	Sampling Sequence Qualifier	**SamplingSequenceQual'r**
1021	Sampling Sequence Value	**SamplingSequenceVal**
1023	Hazard Zone Code	**HzrdZoneCd**
1024	Number of Tank Compartments	**TankCompartments**
1025	Loading or Discharge Location Code	**LoadingDischargeLocCd**
1026	Vessel Material Code	**VesselMat'lCd**
1028	Claim Submitter's Identifier	**ClaimSubmitterID**
1029	Claim Status Code	**ClaimStatusCd**
1030	Gasket Type Code	**GasketTypeCd**

Ref#	Element Name	Type Tree Abbreviation
1031	Trailer Lining Type Code	**TrailerLiningTypeCd**
1032	Claim Filing Indicator Code	**ClaimFilingIndicatorCd**
1033	Claim Adjustment Group Code	**ClaimAdj'tGroupCd**
1034	Claim Adjustment Reason Code	**ClaimAdj'tReasonCd**
1035	Name Last or Organization Name	**NameLastOrgName**
1036	Name First	**NameFirst**
1037	Name Middle	**NameMiddle**
1038	Name Prefix	**NamePrefix**
1039	Name Suffix	**NameSuffix**
1041	Sequence Value	**SequenceVal**
1042	Load or Device Code	**LoadDeviceCd**
1043	Diameter	**Diameter**
1044	Hose Type Code	**HoseTypeCd**
1045	Inlet or Outlet Material Type Code	**InletOutletMat'lTypeCd**
1046	Inlet or Outlet Fitting Type Code	**InletOutletFittingTypeCd**
1047	Miscellaneous Equipment Code	**MiscEquip'tCd**
1048	Business Function Code	**BusinessFunctionCd**
1049	Tax Payment Type Code	**TaxPay'tTypeCd**
1050	Taxpayer Verification	**TaxpayerVerification**
1051	Tax Amount	**TaxAmt**
1053	Market Exchange Identifier	**MktExchangeID**
1054	Commodity Identification	**CommodityID**
1056	Vehicle Production Status	**VehicleProductionStatus**
1062	Vehicle Service Code	**VehicleServiceCd**
1065	Entity Type Qualifier	**EntityTypeQual'r**
1066	Citizenship Status Code	**CitizenshipStatusCd**
1067	Marital Status Code	**MaritalStatusCd**
1068	Gender Code	**GenderCd**
1069	Individual Relationship Code	**Indiv'lRelationshipCd**
1070	Type of Residence Code	**ResidenceCd**
1071	General Expense Qualifier	**GeneralExpenseQual'r**
1072	Rounding System Code	**RoundingSystemCd**
1073	Yes/No Condition or Response Code	**YesNoCd**
1074	Type of Real Estate Asset Code	**REAssetCd**

Ref#	Element Name	Type Tree Abbreviation
1075	Status of Plans for Real Estate Asset Code	**StatusOfPlansREAssetCd**
1076	Real Estate Loan Security Instrument Code	**RELoanSecurityInstrumentCd**
1077	Property Value Estimate Type Code	**PropertyValEstTypeCd**
1078	Property Ownership Rights Code	**PropertyOwnershipRightsCd**
1079	Contact Method Code	**ContactMthdCd**
1080	Assumption Terms Code	**AssumptionTermsCd**
1081	Loan Purpose Code	**LoanPurpCd**
1082	Purpose of Refinance Code	**PurpOfRefinanceCd**
1083	Type of Downpayment Code	**DownpaymentCd**
1084	Loan Buydown Type Code	**LoanBuydownTypeCd**
1085	Loan Payment Type Code	**LoanPay'tTypeCd**
1086	Loan Rate Type Code	**LoanRateTypeCd**
1089	Source of Interest Rate Change Code	**SourceOfInterestRateChangeCd**
1090	Improvement Status Code	**ImprovementStatusCd**
1091	Buydown Source Code	**BuydownSourceCd**
1093	Real Estate Loan Type Code	**RELoanTypeCd**
1094	Vehicle Status	**VehicleStatus**
1095	Year Within Century	**YearWithinCentury**
1096	County Designator	**CountyDesignator**
1097	Mortgage Insurance Application Type	**MortInsApp'nType**
1098	Mortgage Insurance Premium Source Code	**MortInsPremiumSourceCd**
1099	Mortgage Insurance Certificate Type Code	**MortInsCertTypeCd**
1100	Mortgage Insurance Coverage Type Code	**MortInsCoverageTypeCd**
1101	Lien Priority Code	**LienPriorityCd**
1102	Mortgage Insurance Renewal Option Code	**MortInsRenewalOptionCd**
1103	Loan Documentation Type Code	**LoanDoc'nTypeCd**
1104	Name Component Qualifier	**NameComponentQual'r**
1105	Mortgage Insurance Duration Code	**MortInsDurationCd**
1106	Address Component Qualifier	**AddressComponentQual'r**
1107	Name Type Code	**NameTypeCd**
1108	Month of the Year Code	**MonthOfTheYearCd**
1109	Race or Ethnicity Code	**RaceEthnicityCd**

Ref#	Element Name	Type Tree Abbreviation
1110	Vehicle Dimension	**VehicleDimension**
1113	Coupon Distribution Media Code	**CouponDistrib'nMediaCd**
1122	Vent Setting Code	**VentSettingCd**
1123	Offer Basis Code	**OfferBasisCd**
1126	Academic Degree Code	**AcademicDegreeCd**
1127	Interline Settlement System Status Action or Dispute Code	**InterlineSettlementActionCd**
1129	Adjustment Reason Code Characteristic	**Adj'tReasonCdChar**
1130	Primary or Contingent Code	**PrimaryContingentCd**
1131	Level of Individual, Test, or Course Code	**LevelOfIndiv'lTestCd**
1132	Instructional Setting Code	**InstructionalSettingCd**
1133	Other Program Participation and Services Code	**OtherProgramParticipationCd**
1134	Other Program and Services Funding Source Code	**OtherProgramFundingSourceCd**
1135	Placement Criteria Code	**PlacementCriteriaCd**
1136	Code Category	**CdCategory**
1137	Medical Code Value	**MedicalCdVal**
1138	Payer Responsibility Sequence Number Code	**PayorRspSequence#Cd**
1139	Session Code	**SessionCd**
1140	Floor Type Code	**FloorTypeCd**
1141	Academic Credit Type Code	**AcademicCreditTypeCd**
1142	Academic Grade or Course Level Code	**AcademicGradeCd**
1143	Coordination of Benefits Code	**CoordinationOfBenefitsCd**
1144	Academic Grade Point Average	**AcademicGradePtAvg**
1145	Class Rank	**ClassRank**
1146	Disability Type Code	**DisabilityTypeCd**
1147	Basis for Academic Credit Code	**BasisAcademicCreditCd**
1148	Academic Grade Qualifier	**AcademicGradeQual'r**
1149	Occupation Code	**OccupationCd**
1150	Course Repeat or No Count Indicator Code	**CourseRepeatNoCountCd**
1151	Academic Quality Points	**AcademicQualityPoints**
1152	Override Academic Course Source Code	**OverrideAcademicCd**

Ref#	Element Name	Type Tree Abbreviation
1153	Academic Field of Study Level or Type Code	**AcademicFieldOfStudyCd**
1154	Work Intensity Code	**WorkIntensityCd**
1155	Educational Test or Requirement Code	**EducationalTestReq'tCd**
1156	Test Norm Type Code	**TestNormTypeCd**
1157	Test Norming Period Code	**TestNormingPeriodCd**
1158	Subtest Code	**SubtestCd**
1159	Test Score Interpretation Code	**TestScoreInterpretationCd**
1160	Test Score Qualifier Code	**TestScoreQual'rCd**
1161	Product Option Code	**ProdOptionCd**
1162	Show Code	**ShowCd**
1163	Ticket Catagory Code	**TicketCategoryCd**
1164	Network or Schedule Data Type	**NetworkSchedDataType**
1165	Confidentiality Code	**ConfidentialityCd**
1166	Contract Type Code	**ContractTypeCd**
1167	Sample Selection Modulus	**SampleSelectionModulus**
1168	Door Type Code	**DoorTypeCd**
1171	Milestone Number Identification	**Milestone#ID**
1172	Claim Response Reason Code	**ClaimRspReasonCd**
1173	Task ID Qualifier	**TaskIDQual'r**
1174	Task Identifier	**TaskID**
1175	Relationship Task Identifier	**RelationshipTaskID**
1176	Employment Class Code	**EmploymentClassCd**
1178	Level	**Level1178**
1179	Customs Entry Type Group Code	**CustomsEntryTypeGroupCd**
1180	Resource Code (or Identifier)	**ResourceCdID**
1181	Resource Type	**ResourceType**
1185	Public Record or Obligation Code	**PublicRecordObligationCd**
1186	Type of Income Code	**IncomeCd**
1187	Type of Account Code	**Acc'tCd**
1188	Type of Personal Property Code	**PersonalPropertyCd**
1189	Type of Credit Account Code	**CreditAcc'tCd**
1193	Program Type Code	**ProgramTypeCd**
1195	Financial Information Code	**FinancialInfoCd**

Ref#	Element Name	Type Tree Abbreviation
1196	Breakdown Structure Detail Code	**BreakdownStructureDetailCd**
1197	Financial Transaction Status Code	**FinancialTransc'nStatusCd**
1198	Contracting Funding Code	**ContractingFundingCd**
1199	Appropriation Code	**AppropriationCd**
1201	Information Status Code	**InfoStatusCd**
1202	Flexible Spending Account Selection Code	**FlexibleSpendingAcc'tSelectionCd**
1203	Maintenance Reason Code	**MaintReasonCd**
1204	Plan Coverage Description	**PlanCoverageDesc'n**
1205	Insurance Line Code	**InsLineCd**
1207	Coverage Level Code	**CoverageLevelCd**
1209	Underwriting Decision Code	**UnderwritingDecisionCd**
1211	Drug House Code	**DrugHouseCd**
1212	Health-Related Code	**HealthRelatedCd**
1213	Current Health Condition Code	**CurrentHealthCond'nCd**
1214	Salary Grade	**SalaryGrade**
1215	Identification Card Type Code	**IDCardTypeCd**
1216	Benefit Status Code	**BenefitStatusCd**
1218	Medicare Plan Code	**MedicarePlanCd**
1219	Consolidated Omnibus Budget Reconciliation Act (COBRA) Qualifying Event Code	**COBRAQualifyingEventCd**
1220	Student Status Code	**StudentStatusCd**
1221	Provider Code	**ProviderCd**
1222	Provider Specialty Code	**ProviderSpecialtyCd**
1223	Provider Organization Code	**ProviderOrgCd**
1224	Contribution Code	**ContributionCd**
1225	Disposition Code	**DispositionCd1225**
1226	Repair Action Code	**RepairActionCd**
1227	Repair Complexity Code	**RepairComplexityCd**
1228	Casual Part Condition Code	**CasualPartCond'nCd**
1229	Complaint Code	**ComplaintCd**
1230	Type of Product Service Code	**ProdServiceCd**
1231	Operation Environment Code	**OperationEnvironmentCd**
1232	Purchase Category	**PurchaseCategory**
1233	Service Classification Code	**ServiceClass'nCd**
1234	Severity Condition Code	**SeverityCond'nCd**

Ref#	Element Name	Type Tree Abbreviation
1236	Payment Type Code	**Pay'tTypeCd**
1237	Move Type Code	**MoveTypeCd**
1238	Bay Type Code	**BayTypeCd**
1239	Capacity Qualifier	**CapacityQual'r**
1240	Facility Characteristic Code	**FacilityCharCd**
1241	Facility Characteristic Code Qualifier	**FacilityCharCdQual'r**
1242	Demand Area	**DemandArea**
1243	Financial Status	**FinancialStatus**
1244	National Motor Freight Transportation Association Location Name	**NatlMotorFrtTransAssocLocName**
1245	Vehicle Identification Number (VIN) Plant Code	**VINPlantCd**
1246	Special Rate Code	**SpclRateCd**
1250	Date Time Period Format Qualifier	**DateTimePeriodFormatQual'r**
1251	Date Time Period	**DateTimePeriod**
1252	Health Screening Type Code	**HealthScreeningTypeCd**
1253	Immunization Type Code	**ImmunizationTypeCd**
1254	Immunization Status Code	**ImmunizationStatusCd**
1255	Disease Condition Type Code	**DiseaseCond'nTypeCd**
1256	Medical Treatment Type Code	**MedicalTreatmentTypeCd**
1257	Special Education Program Participation Code	**SpclEdProgramParticipationCd**
1258	Academic Grade	**AcademicGrade**
1260	Reported Data ID Code	**RptDataIDCd**
1261	Reported Data Response	**RptDataRsp**
1262	Loan Type Code	**LoanTypeCd**
1264	Delayed Repayment Reason Code	**DelayedRepaymentReasonCd**
1265	Interest Payment Code	**InterestPay'tCd**
1266	Major Course of Study	**Major**
1267	Dependency Status Code	**DependencyStatusCd**
1268	Applicant Type Code	**ApplicantTypeCd**
1270	Code List Qualifier Code	**CdListQual'rCd**
1271	Industry Code	**IndustryCd**
1272	Horsepower	**Horsepower**
1273	Direction Facing	**DirectionFacing**
1274	Train Delay Reason Code	**TrainDelayReasonCd**
1275	Fumigated/Cleaned Indicator	**FumigatedCleanedIndicator**

Ref#	Element Name	Type Tree Abbreviation
1276	Machine Separable Indicator Code	**MachineSeparableIndicatorCd**
1277	Canadian Wheat Board (CWB) Marketing Class Code	**CWBMarketingClassCd**
1278	Canadian Wheat Board (CWB) Marketing Class Type Code	**CWBMarketingClassTypeCd**
1280	Direction Identifier Code	**DirectionIDCd**
1282	Treasury Symbol Number	**TreasurySymbol#**
1283	Budget Activity Number	**BudgetActivity#**
1284	Object Class Number	**ObjectClass#**
1285	Reimbursable Source Number	**ReimbursableSource#**
1286	Transaction Reference Number	**Transc'nRef#1286**
1287	Accountable Station Number	**AccountableStation#**
1288	Paying Station Number	**PayingStation#**
1292	Returns Disposition Code	**ReturnsDispositionCd**
1293	Return Request Reason Code	**ReturnRequestReasonCd**
1294	Return Response Reason Code	**ReturnRspReasonCd**
1295	Participant Status Code	**ParticipantStatusCd**
1296	Special Processing Type	**SpclProcessingType**
1297	Work Status Code	**WorkStatusCd**
1300	Service, Promotion, Allowance, or Charge Code	**ServicePromoAllowChrgCd**
1301	Agency Service, Promotion, Allowance, or Charge Code	**AgencyServicePromoAllowChrgCd**
1302	Shipper's Export Declaration Requirements	**ShipperExportDeclarationReq't**
1303	Use of Language Indicator	**UseOfLanguageIndicator**
1304	Mark Code Type	**MarkCdType**
1306	U.S. Government License Type	**USGovtLicenseType**
1307	Loan Status Code	**LoanStatusCd**
1308	Contract Action Code	**ContractActionCd**
1309	Acquisition Data Code	**AcquisitionDataCd**
1310	Financing Type Code	**FinancingTypeCd**
1311	Calculation Operation Code	**CalculationOperationCd**
1312	Test Period or Interval Qualifier	**TestPeriodIntervalQual'r**
1313	Test Period or Interval Value	**TestPeriodIntervalVal**
1314	Admission Source Code	**AdmissionSourceCd**

Ref#	Element Name	Type Tree Abbreviation
1315	Admission Type Code	**AdmissionTypeCd**
1316	Ambulance Transport Code	**AmbulanceTransportCd**
1317	Ambulance Transport Reason Code	**AmbulanceTransportReasonCd**
1318	Approval Code	**ApprovalCd**
1319	Basis of Cost Determination Code	**BasisOfCostDetermnCd**
1320	Basis of Days Supply Determination Code	**BasisOfDaysSupplyDetermnCd**
1321	Condition Indicator	**Cond'nIndicator**
1322	Certification Type Code	**Cert'nTypeCd**
1325	Claim Frequency Type Code	**ClaimFreqTypeCd**
1327	Copay Status Code	**CopayStatusCd**
1328	Diagnosis Code Pointer	**DiagnosisCdPointer**
1329	Dispense as Written Code	**DispenseAsWrittenCd**
1330	Dosage Form Code	**DosageFormCd**
1331	Facility Code Value	**FacilityCd**
1332	Facility Code Qualifier	**FacilityCdQual'r**
1333	Record Format Code	**RecordFormatCd**
1334	Health Care Professional Shortage Area Code	**HPSACd**
1335	Insulin Dependent Code	**InsulinDependentCd**
1336	Insurance Type Code	**InsTypeCd**
1337	Level of Care Code	**LevelOfCareCd**
1338	Level of Service Code	**LevelOfServiceCd**
1339	Procedure Modifier	**ProcedureModifier**
1340	Multiple Procedure Code	**MplProcedureCd**
1341	National or Local Assigned Review Value	**NatlLocalAssignedReviewVal**
1342	Nature of Condition Code	**NatureOfCond'nCd**
1343	Non-Institutional Claim Type Code	**NonInstitutionalClaimTypeCd**
1344	Non-Visit Code	**NonVisitCd**
1345	Nursing Home Residential Status Code	**NursingHomeResidentialStatusCd**
1346	Nutrient Administration Method Code	**NutrientAdminMthdCd**
1347	Nutrient Administration Technique Code	**NutrientAdminTechniqueCd**
1348	Oxygen Equipment Type Code	**OxygenEquip'tTypeCd**
1349	Oxygen Test Condition Code	**OxygenTestCond'nCd**
1350	Oxygen Test Findings Code	**OxygenTestFindingsCd**

Ref#	Element Name	Type Tree Abbreviation
1351	Patient Signature Source Code	**PatientSignatureSourceCd**
1352	Patient Status Code	**PatientStatusCd**
1354	Diagnosis Related Group (DRG) Code	**DRGCd**
1355	Prescription Denial Override Code	**PrescriptionDenialOverrideCd**
1356	Prescription Origin Code	**PrescriptionOriginCd**
1357	Prior Authorization Type Code	**PriorAuth'nTypeCd**
1358	Prosthesis, Crown or Inlay Code	**ProsthesisCrownInlayCd**
1359	Provider Accept Assignment Code	**ProviderAcceptAssignmentCd**
1360	Provider Agreement Code	**ProviderAgreementCd**
1361	Oral Cavity Designation Code	**OralCavityDesignationCd**
1362	Related-Causes Code	**RelatedCausesCd**
1363	Release of Information Code	**ReleaseOfInfoCd**
1364	Review Code	**ReviewCd**
1365	Service Type Code	**ServiceTypeCd**
1366	Special Program Code	**SpclProgramCd**
1367	Subluxation Level Code	**SubluxationLevelCd**
1368	Tooth Status Code	**ToothStatusCd**
1369	Tooth Surface Code	**ToothSurfaceCd**
1370	Unit Dose Code	**UnitDoseCd**
1371	Unit Rate	**UnitRate**
1373	Measurement Method or Device	**Meas'tMthdDevice**
1375	Interim Hazardous Material Regulatory Number	**InterimHzrdMat'lRegulatory#**
1376	Investor Reporting Action Code	**InvestorReportingActionCd**
1378	Waybill Cross-Reference Code	**WaybillCrossRefCd**
1379	Loan Verification Code	**LoanVerificationCd**
1382	Oxygen Delivery System Code	**OxygenDeliverySystemCd**
1383	Claim Submission Reason Code	**ClaimSubmissionReasonCd**
1384	Patient Location Code	**PatientLocCd**
1387	Rate Qualifier	**RateQual'r**
1390	Eligibility or Benefit Information	**EligibilityBenefitInfo**
1392	Market Profile	**MktProfile**

Ref#	Element Name	Type Tree Abbreviation
1393	Media Type Identifier	**MediaTypeID**
1395	Configuration Type Code	**Config'nTypeCd**
1396	Equipment Use Code	**Equip'tUseCd**
1397	Inquiry Response Code	**InquiryRspCd**
1398	Inquiry Selection Code	**InquirySelectionCd**
1400	Hierarchy Code	**HierarchyCd**
1401	Proposal Data Detail Identifier Code	**ProposalDataDetailIDCd**
1402	Equipment Attribute Code	**Equip'tAttributeCd**
1403	Implant Type Code	**ImplantTypeCd**
1404	Implant Status Code	**ImplantStatusCd**
1413	Usage Indicator	**UsageIndicator**
1415	Specimen Kit Type Code	**SpecimenKitTypeCd**
1420	Title Insurance Services Code	**TitleInsServicesCd**
1422	Damage Status Code	**DmgStatusCd**
1423	License Plate Type	**LicensePlateType**
1425	Recovery Classification Code	**RecoveryClass'nCd**
1426	Recovery Condition Code	**RecoveryCond'nCd**
1428	Master In-bond Type Code	**MasterInbondTypeCd**
1429	Construction Type	**ConstructionType**
1430	Day Rotation	**DayRotation**
1431	Preference	**Preference**
1432	Business Purpose of Assurance	**BusinessPurpOfAssurance**
1434	Domain of Computation of Assurance Digest	**DomainOfAssuranceDigest**
1435	Assurance Originator	**AssuranceOriginator**
1436	Assurance Recipient	**AssuranceRecipient**
1437	Date/Time Reference	**DateTimeRef**
1438	Assurance Text	**AssuranceText**
1439	Assurance Token Parameter Code	**AssuranceTokenParameterCd**
1440	Assurance Digest	**AssuranceDigest**
1442	Assurance Token Parameter Value	**AssuranceTokenParameterVal**
1443	Assurance Reference Number	**AssuranceRef#**
1460	Part of Body Code	**PartOfBodyCd**
1461	Cause of Injury Code	**CauseOfInjuryCd**
1462	Initial Treatment Code	**InitialTreatmentCd**
1463	Nature of Injury Code	**NatureOfInjuryCd**

Ref#	Element Name	Type Tree Abbreviation
1464	Source of Injury Code	**SourceOfInjuryCd**
1465	Proximity Code	**ProximityCd**
1466	Location Type Code	**LocTypeCd**
1468	Reason Stopped Work Code	**ReasonStoppedWorkCd**
1469	Affected Area or Section Code	**AffectedAreaSectionCd**
1470	Number	**#**
1472	Report Section Number	**RptSection#**
1473	Pricing Methodology	**PricingMethodology**
1476	Language Proficiency Indicator	**LanguageProficiencyIndicator**
1482	Mechanical Car Code	**Mech'lCarCd**
1484	Problem Log Reason Code	**ProblemLogReasonCd**
1485	Service Commitment Type Code	**ServiceCommitmentTypeCd**
1487	Retrip Reason Code	**RetripReasonCd**
1488	Bad Order Reason Code	**BadOrderReasonCd**
1489	Hold Reason Code	**HoldReasonCd**
1490	Association of American Railroads Car Grade Code	**AARCarGradeCd**
1491	Parameter Trace Registration Type Code	**ParameterTraceReg'nTypeCd**
1492	Parameter Trace Type Code	**ParameterTraceTypeCd**
1493	Output Event Selection Code	**OutputEventSelectionCd**
1494	Transportation Condition Code	**Transp'nCond'nCd**
1496	Property Description Qualifier	**PropertyDesc'nQual'r**
1497	Notification Entity Qualifier	**NotificationEntityQual'r**
1499	Rate Application Code	**RateApp'nCd**
1511	Type of Deduction	**Deduction**
1514	Delay Reason Code	**DelayReasonCd**
1520	Display Type Code	**DisplayTypeCd**
1521	Marketing Type Code	**MarketingTypeCd**
1522	Coupon Type Code	**CouponTypeCd**
1523	Labor Activity Code	**LaborActivityCd**
1525	Request Category Code	**RequestCategoryCd**
1526	Policy Compliance Code	**PolicyComplianceCd**
1527	Exception Code	**ExceptionCd**
1528	Component Data Element Position in Composite	**ComponentDEPos'nInComposite**
1540	Net Cost Code	**NetCostCd**

Ref#	Element Name	Type Tree Abbreviation
1543	Equipment Orientation Code	**Equip'tOrientationCd**
1546	Preferential Duty Criteria Code	**PreferentialDutyCriteriaCd**
1550	Index Qualifier	**IndexQual'r**
1551	Message Text	**MsgText**
1554	Tag Status Code	**TagStatusCd**
1555	Automatic Equipment Identification Consist Confidence Level Code	**AutoEquip'tIDConfidenceLevelCd**
1556	Train Termination Status Code	**TrainTerminationStatusCd**
1557	Movement Type Code	**MovementTypeCd**
1558	Academic Summary Source	**AcademicSummarySource**
1559	Automatic Equipment Identification Site Status Code	**AutoEquip'tIDSiteStatusCd**
1560	Interchange Type Code	**InterchangeTypeCd**
1561	Rail Junction Settlement Role Code	**RailJunctionSettlementRoleCd**
1564	Protocol ID	**ProtocolID**
1565	Look-up Value	**LookupVal**
1566	Keying Material	**KeyingMat'l**
1567	One-time Encryption Key	**OnetimeEncryptionKey**
1568	Algorithm ID	**AlgorithmID**
1569	Algorithm Mode of Operation	**AlgorithmModeOfOperation**
1570	Filter ID Code	**FilterIDCd**
1571	Compression ID	**CompressionID**
1572	Security Value Qualifier	**SecurityValQual'r**
1573	Encoded Security Value	**EncodedSecurityVal**
1574	Assurance Algorithm	**AssuranceAlgorithm**
1575	Hashing Algorithm	**HashingAlgorithm**
1576	Inspected/Weighed Indicator Code	**InspectedWeighedIndicatorCd**
1577	Hazardous Material Regulations Exception Code	**HzrdMat'lRegulationsExceptionCd**
1578	Export Exception Code	**ExportExceptionCd**
1585	Shape Code	**ShapeCd**
1588	Political Party Affiliation Code	**PoliticalPartyAffiliationCd**
1589	Harbor Maintenance Fee (HMF) Exemption Code	**HMFExemptionCd**
1603	Route of Administration	**RouteofAdmin**
1606	Animal Disposition Code	**AnimalDispositionCd**

Ref#	Element Name	Type Tree Abbreviation
1607	Test Type Code	**TestTypeCd**
1608	Non-Numeric Test Value	**NonNumericTestValue**
1611	Observation Type Code	**ObservationTypeCd**
1612	Tissue or Specimen Disposition Code	**TissueOrSpecimenDispositionCd**
1614	Sub-Location	**SubLocation**
1615	Observation Distribution	**ObservationDistribution**
1616	Observation Severity	**ObservationSeverity**
1617	Neoplasm Code	**NeoplasmCd**
1618	Linkage Identifier	**LinkageID**
1619	Parturition Status Code	**ParturitionStatusCd**
1620	Offspring Count Code	**OffspringCountCd**
1622	Offspring/Fetus Status Code	**OffspringFetusStatusCd**
I01	Authorization Information Qualifier	**Auth'nInfoQual'r**
I02	Authorization Information	**Auth'nInfo**
I03	Security Information Qualifier	**SecurityInfoQual'r**
I04	Security Information	**SecurityInfo**
I05	Interchange ID Qualifier	**InterchangeIDQual'r**
I06	Interchange Sender ID	**InterchangeSenderID**
I07	Interchange Receiver ID	**InterchangeRcv'rID**
I08	Interchange Date	**InterchangeDate**
I09	Interchange Time	**InterchangeTime**
I10	Interchange Control Standards Identifier	**InterchangeCtrlStandardsID**
I11	Interchange Control Version Number	**InterchangeCtrlVersion#**
I12	Interchange Control Number	**InterchangeCtrl#**
I13	Acknowledgment Requested	**Ack'tRequested**
I14	Test Indicator	**TestIndicatorI14**
I15	Component Element Separator	**ComponentElementSeparator**
I16	Number of Included Functional Groups	**#InclFunct'lGroups**
I17	Interchange Acknowledgment Code	**InterchangeAckCd**
I18	Interchange Note Code	**InterchangeNoteCd**
I19	Interconnect Mailbag Version Number	**InterconnectMailbagVersion#**
I20	Interconnect Mailbag Logon ID	**InterconnectMailbagLogonID**
I21	Interconnect Mailbag Password	**InterconnectMailbagPassword**

Ref#	Element Name	Type Tree Abbreviation
I22	Interconnect Mailbag ID Qualifier Code	**InterconnectMailbagIDQual'rCd**
I23	Interconnect Mailbag Sender ID	**InterconnectMailbagSenderID**
I24	Interconnect Mailbag Receiver ID	**InterconnectMailbagRcv'rID**
I25	Interconnect Mailbag Date	**InterconnectMailbagDate**
I26	Interconnect Mailbag Time	**InterconnectMailbagTime**
I27	Interconnect Mailbag Time Code	**InterconnectMailbagTimeCd**
I28	Interconnect Mailbag Control Number	**InterconnectMailbagCtrl#**
I29	Interconnect Mailbag Test Indicator	**InterconnectMailbagTestIndicator**
I30	Interconnect Mailbag Acknowledgment Count	**InterconnectMailbagAckCount**
I31	Interconnect Mailbag Interchange Count	**InterconnectMailbagICCount**
I32	Interconnect Mailbag Acknowledgment Action Code	**InterconnectMailbagAckActionCd**
I33	Interconnect Mailbag Error Code	**InterconnectMailbagErrorCd**
I34	Grade of Service Code	**GradeOfServiceCd**
I35	Delivery Date	**DeliveryDateI35**
I36	Delivery Time	**DeliveryTime**
I37	Delivery Time Code	**DeliveryTimeCd**
I38	Service Request Handler ID Qualifier	**ServiceRequestHandlerIDQual'r**
I39	Service Request Handler ID	**ServiceRequestHandlerID**
I40	Interchange Action Code	**InterchangeActionCd**
I41	Interchange Action Date	**InterchangeActionDate**
I42	Interchange Action Time	**InterchangeActionTime**
I43	Error Reason Code	**ErrorReasonCd**
I44	Reported Interchange Start Segment ID	**ReportedInterchangeStartSegID**
I45	Reported Interchange Control Number	**ReportedInterchangeCtrl#**
I46	Reported Interchange Date	**ReportedInterchangeDate**
I47	Reported Interchange Time	**ReportedInterchangeTime**
I48	Reported Interchange Sender ID Qualifier	**ReportedInterchangeSenderIDQual**
I49	Reported Interchange Sender ID	**ReportedInterchangeSenderID**
I50	Reported Interchange Receiver ID Qualifier	**ReportedInterchangeRcv'rIDQual'r**

Ref#	Element Name	Type Tree Abbreviation
I51	Reported Interchange Receiver ID	**ReportedInterchangeRcv'rID**
I52	First Reference ID Qualifier	**FirstRefIDQual'r**
I53	First Reference ID	**FirstRefID**
I54	Second Reference ID Qualifier	**SecondRefIDQual'r**
I55	Second Reference ID	**SecondRefID**
I56	Reference Code Qualifier	**RefCdQual'r**
I57	Reference Code	**RefCd**
I66	Exchange Block Sequence	**ExchangeblockSequence**
I67	Exchange Block Type Identifier	**ExchangeBlockTypeID**
I68	Exchange Block Length	**ExchangeBlockLength**

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
577 Airport Blvd., Suite 800
Burlingame, CA 94010
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, a nd represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and service marks

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries, or both:

AIX
AIX 5L
AS/400
Ascential
Ascential DataStage
Ascential Enterprise Integration Suite
Ascential QualityStage
Ascential RTI
Ascential Software
Ascential
CICS
DataStage
DB2
DB2 Universal Database
developerWorks
Footprint
Hiperspace
IBM
the IBM logo
ibm.com
IMS
Informix
Lotus
Lotus Notes
MQSeries
MVS
OS/390
OS/400
Passport Advantage
Redbooks
RISC System/6000
Roma
S/390
System z
Trading Partner
Tivoli

WebSphere
z/Architecture
z/OS
zSeries

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

This product includes software developed by the Eclipse Project (http://www.eclipse.org/).

IBM WebSphere Transformation Extender Pack for EDI, Version 2.7

Index

Numerics

A

C

D

E

F

I

L

Printed in USA