TIBCO Foresight® EDISIM®

Analyzer User's Guide

Version 6.20.0

May 2021

Contents

1 EDISIM Analyzer

Where Analyzer Fits

TIBCO Foresight® EDISIM® Analyzer comes with EDISIM® to accomplish these tasks for you:

- Checks EDI data to be sure that it complies with a guideline, MIG, or published standard that is in the EDISIM database.
- Reads EDI data files and creates a guideline or MIG from them. This feature is called DocStarter and is described in DocStarter on page 47.

Analyzer compares EDI data to a standard, MIG, or guideline. It finds syntax discrepancies and provides immediate visual cross-references between data and error messages.

What Analyzer Checks

Data checked by Analyzer

- X12-based EDI
- EDIFACT-based EDI
- EANCOM
- GENCOD
- ODETTE
- TRADACOMS
- Many industry standards like VICS and UCS

The data must include interchange and functional group enveloping (except in EDIFACT, where functional groups are optional). It can be wrapped, padded, or in just about any format that is recognizable as EDI data, such as:

- Each line ending with a segment terminator followed by a carriage return (CR) and line feed (LF).
- Each line ending with a CR/LF (CR/LF is the segment terminator).
- Blanks padding the interchange trailer or not.
- CR/LF embedded in mid-segment (simulated fixed length records containing wrapped data).

Conditions checked by Analyzer

Extended Business Rules

In addition to the checking described below, you can use EDISIM's extensive set of business rules to set up your own custom checking. See **BusinessRules.pdf** in EDISIM's Documentation directory.

The following is a partial list of items checked by Analyzer.

Delimiters

- Checks segment, element, and subelement delimiters.
- Checks for extra trailing delimiters in segments and composites. These can be extra delimiters, or delimiters for omitted elements at the end of the segment.
- Checks the EDIFACT UNA Service String Advice, if present, to be sure it is fixed length and contains delimiters.

Enveloping

- Verifies that interchange and functional group envelopes are present and paired up correctly: each X12 ISA segment has a corresponding IEA, each EDIFACT UNB segment has a corresponding UNZ, nothing is outside the enveloping, etc.
- Checks if the functional group header (X12 GS or EDIFACT UNG) includes the standard name (the version, release, and industry code). If it does, Analyzer checks the syntax and codes in the X12 GS and GE segments and the EDIFACT UNG and UNE against the dictionary definition of the standard, guideline, or MIG being used for the compliance checking.
- Analyzer can check the following enveloping segments against the dictionary segments of a specified standard:
  - X12: ISA, GS, GE, and IEA
  - EDIFACT: UNA, UNB, UNG, UNE, and UNZ
  - Checks for an empty envelope.
  - EDIFACT only: Analyzer supports ISO 9735 versions 2, 3, and 4.

- X12 only: Checks that the interchange header is the right size and fixed length (X12 ISA and ICS).

Counts and Control Numbers (Message Reference Numbers)

- Verifies that the segment counts in the transaction set or message trailers are correct.
- Compares the control numbers or message reference numbers in the X12 ST and SE, and in the EDIFACT UNH and UNT.
- For X12, checks hierarchical loops. HL segments - checks child and parent references: Does the child exist? Does the parent exist?
- For X12, checks CTT counts, if the guideline is set up for CTT checking.

Segments, Elements, and Composites

- Checks whether the segment, element, or composite is in the dictionary.
- Checks for empty segments.
- Checks the number of elements in a segment.
- Checks whether a mandatory element, segment, X12 loop, or EDIFACT group is missing.
- Checks for the presence of recommended segments, composites, and elements.
- Checks for the absence of not recommended segments, composites, and elements.
- Checks whether a "must be used" segment, composite, or element is missing.
- Checks whether a "not used" segment, composite, or element is included.
- Checks that X12 syntax rules and EDIFACT dependency notes are fulfilled (conditions such as "If this element is used, then that element must also be used").
- Checks for business rules such as "If the third element in the XXX segment contains a value of 03, then segment ZZZ must be used. Otherwise, ZZZ is not used."

Values

- Checks length and data type for values in enveloping segments.
- If an element has a code values list, Analyzer checks to be sure the value in the EDI file is one of them.
- If an element has an application values list (created by Standards Editor), Analyzer checks to be sure that the value in the EDI file is one of them if it is not a code value.
- Checks the length and type (numeric, etc.) of each value.
- Checks whether unprintable data appears in an element (exception: BIN segment may contain unprintable data).
- If the guideline or MIG has a business rule that requests date/time checking, Analyzer checks the date to see if it matches the format specified by the qualifier.

Unused Loops and Groups

- If an X12 loop or EDIFACT group is included in the data, even though the guideline or MIG says it is unused, Analyzer will give you a diagnostic on the trigger (first) segment. However, to prevent a blizzard of diagnostics, the segments in the loop or group will not be flagged. Instead, Analyzer performs a full compliance check on the enclosed segments, exactly as if they were marked as used.

Ambiguous Loops, Groups, or Segments

- A standard may have two loops or segments with the same name but they are actually quite different. Please see Analyzing Ambiguous Loops, Groups and Segments on page 35 for a description of how Analyzer uses code values to tell them apart.

Diagnostics

In addition to error messages, Analyzer displays informative messages telling you what file you are analyzing, notifying you of the end of an interchange, reporting number of segments in a transaction set or message, etc.

When Analyzer compares your EDI data to a guideline or MIG, it also expects your guideline or MIG to conform to the rules set down in the underlying standard. If you violate that standard when creating your guideline or MIG, you may see diagnostics pointing out the deviation.

Why use Analyzer to Check Compliance?

- Analyzer checks EDI data against published standards (X12, EDIFACT, etc.), industry guidelines (VICS, UCS, etc.) or company guidelines/ MIGs.
- Personnel who test trading partners do not necessarily have ready access to the translator. Analyzer provides "desktop" checking.
- Analyzer's dual-panel display is intuitive and easy to navigate.
- Analyzer's compliance and informational messages are complete and easily understood.
- The print reports with interleaved data and diagnostics are clear.
- Analyzer checks thoroughly, especially with its ability to identify whether a segment is inside or outside a loop, to distinguish between alternate definitions of loops/groups or segments, to enforce business rules, and to check application values.
- Analyzer can enforce your own business rules ("if" statements of your own choosing).
- Analyzer is fast, averaging about 250 segments per second.

2 Tutorial

Checking EDI for Compliance to X12

1. Open Analyzer with **Start | Programs |** *<TIBCO_HOME>* **| EDISIM | Analyzer**.
2. Select an EDI file with **File | Open |** *< EDISIM's Samples directory>* **| Bad850.txt**.
3. Select standard **X12-4010** and click **OK**.
4. Dismiss the EDI Compliance Check box by clicking **OK**.
5. Click on the first line in the top pane.
6. Use your cursor down arrow to highlight each message in the top pane. Notice that the bottom pane highlights the corresponding segment.

   This is the first error:

   ```
   Syntax Error for ITD03 (D.E. 338) at col. 10 (R 1/6) "0010": Too many
   leading zeros.
   ```

   Notice that the bottom pane shows the erroneous ITD03 data in red.
7. Right-click on the error message and choose **Diagnostic Message Type.**

   This message is an error.
8. Continue scrolling down and reviewing the other errors.
9. Close Analyzer.

Checking EDI for Compliance to a Company Guideline

Prepare by importing a sample company guideline:

1. Open Standards Editor with **Start | Programs |** *<TIBCO_HOME>* **| EDISIM | Standards Editor**.
2. Dismiss the Welcome box by clicking **OK**.
3. Choose **File | Import | Import Single .SEF and open**.
4. Go to EDISIM's **Samples** directory, choose **OUR850.sef**, and click **Open**.
5. Accept the name **OUR850**.
6. After it finishes importing, close Standards Editor.

Analyze Bad850.txt with the company guideline:

1. Close Analyzer if it is open. This refreshes the list of standards and guidelines.
2. Open **Analyzer** and open **Bad850.txt**.
3. Select company guideline **OUR850** and click **OK**.
4. Dismiss the EDI Compliance Check box.
5. Notice that the X12 messages again appear, along with others that reflect the specifications in OUR850, the sample company guideline.

Checking EDI for Compliance to EDIFACT

1. Open Analyzer with **Start | Programs |** *<TIBCO_HOME>* **| EDISIM | Analyzer**.
2. Choose **File | Open** and select **Testpom.txt** in EDISIM's Samples directory.
3. Select standard **D96A** and click **OK**.
4. Dismiss the EDI Compliance Check box.
5. Starting at the first line in the top pane, use your cursor arrows to move down through the messages.

   Notice that the corresponding segment in the bottom pane is highlighted.

The blue and red diagnostics show ways that this EDI file does not comply with EDIFACT specifications.

6. Right-click on the red message in the top pane and choose **Diagnostic Message Type**.

   The missing mandatory element is red because it is a fatal error.

7. Continue scrolling and reviewing. When finished, close Analyzer.

3 Basic Analyzer Use

Starting Analyzer

There are three methods you can use to start Analyzer.

- **Start | Program Files |** ***<TIBCO_HOME>*** **| EDISIM | Analyzer**.
- Open the EDISIM folder on your desktop and double click on Analyzer.
- Double click on file **FSAnalyz.exe** from Windows Explorer. It is in EDISIM's **Bin** folder. This is available for standalone installations, but not for network installations.

Analyzer may run at the same time as the other EDISIM products, but it must have exclusive access to the file containing the EDI data.

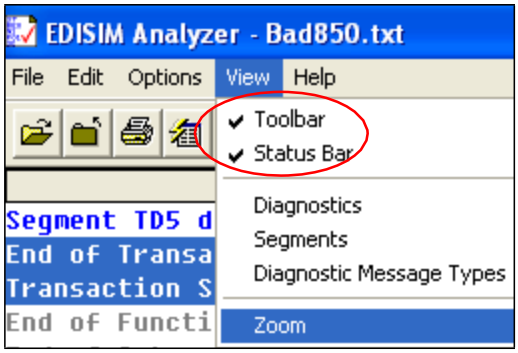
Keyboard

Menus and dialog boxes have choices that can be selected with the keyboard as well as the mouse.

Key	Menu Selections	Action
Alt		Hold down while pressing the first letter of a menu name. Once a menu is dropped down, press the underlined key alone to make your choice.
Ctrl		Hold down when using shortcut keys, which are listed on the menus. Example: drop down the File menu. You can see the shortcut keys listed on the right. *Ctrl*+**o** is the same as File | Open.
Ctrl+Home Ctrl+End		Goes to top or bottom of the contents of the current window.
Esc		Cancel.
F6	View | Diagnostics View | Segments	Moves between Diagnostics pane and Segments pane.
F7	View | Zoom	Zooms (expands) currently active pane. This is a toggle.
Home End		Goes to the beginning or end of the current line.
Pg Up Pg Dn		Scrolls up or down one screen.

Toolbar and Status Bar

Use the View menu to display or hide the Toolbar and Status Bar.



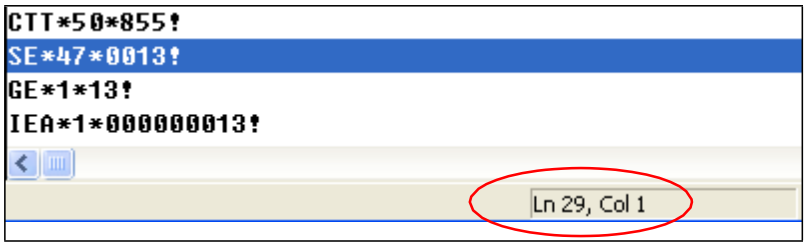
Toolbar

The toolbar is the row of icons at the top left of the Analyzer screen:

To see a brief explanation of a button, rest your mouse cursor on it for a moment.

Status Bar

The status bar is the last line at the bottom of the screen. It shows the number of the current line and the first column that is showing if you scroll right. At times, it displays brief messages.



Choosing a File to Analyze

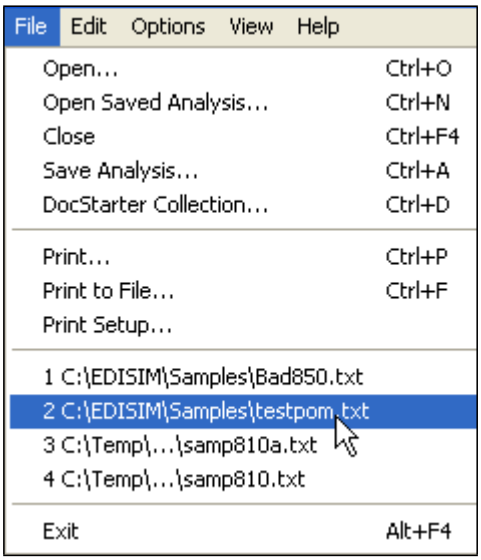
You have these ways to tell Analyzer which file contains the data to be analyzed:

- Click on the open folder icon on the toolbar: 
  You will see a File Open dialog box where you can choose the data file.

- Use the menu items **File | Open**.

  You will see a File Open dialog box where you can choose the data file.

- Click on one of the recently opened files listed at the bottom of the File menu. This begins analyzing the file immediately.



- Drag the data file from Windows Explorer and drop it in either Analyzer pane.

Choosing a Standard, Guideline, or MIG

After selecting a data file, you normally see the Select Standard dialog box, which asks you to choose the standard, guideline, or MIG to be used for compliance checking.

Under certain circumstances, Analyzer will skip this box and use the first standard, guideline, or MIG that could match the data file. To have Analyzer always displays this box before an analysis, select **Options | Analyzer Profile | Ask for Standard Name**.

Using the Select Standard Dialog Box

After you choose an EDI data file to analyze, either of the following causes Analyzer to stop and ask you to choose a standard:

- If you have **Options | Analyzer Profile | Ask for Standard Name** checked (this is the default setting). This lets you be sure of which standard is being used.
- If there is some ambiguity concerning which one to use. Several entries in the list may have the same Version-Release-Industry code (VRI), or perhaps no entry will have it. For guidelines, VRIs can be seen in Standard Editor's **Properties**.

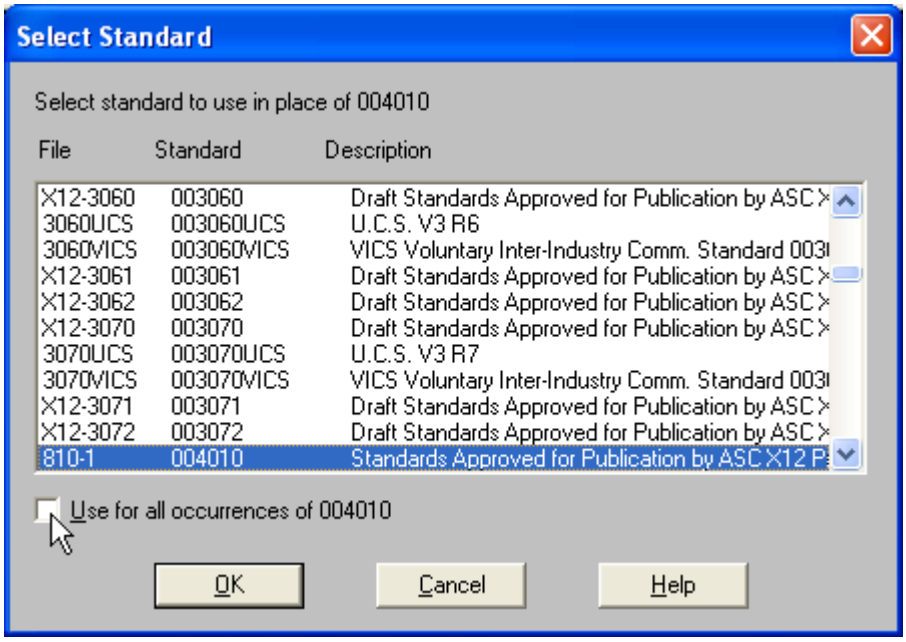
In this dialog box:

1. Highlight the desired standard, guideline, or MIG.

   Analyzer highlights the first standard, guideline, or MIG whose version, release, and industry identifier code (VRI) matches. The list is in alphabetic order by the Standard column.

   If no match is found, the highlight will be on the line below the standard that would have matched. You can scroll up or down from there.

2. If you'd like, select "Use for all occurrences" at the bottom.



   This is convenient if you have multiple transactions or messages in the same file. It uses one standard, guideline, or MIG to analyze the entire file.

   Otherwise, Analyzer will display the Select Standard dialog box if it finds other interchange or functional group headers.

3. Choose **OK**.

Running the Analysis

After resolving any question about which standard, guideline, or MIG is to be used, the analysis begins. Segments and diagnostics scroll as the file processes.

Tip You can speed up the analysis by 20-30% by stopping the scrolling. To do this, click on a segment or diagnostic while the file processes.

After the entire file is processed, choose **OK** or press ***Enter*** to dismiss the Analysis Completed box and view the results.

Viewing the Results

Synchronized Scrolling

The top pane shows diagnostics and the bottom pane shows each segment in the data file.

If you scroll through one pane using your cursor arrows, the other one will scroll accordingly. As you highlight a diagnostic in the top pane, its corresponding segment will highlight. As you highlight a segment in the bottom pane, any diagnostic pertaining to it also highlights.

Which Pane is Active?

The active pane has a blue header. To make a pane active, use any of these methods:

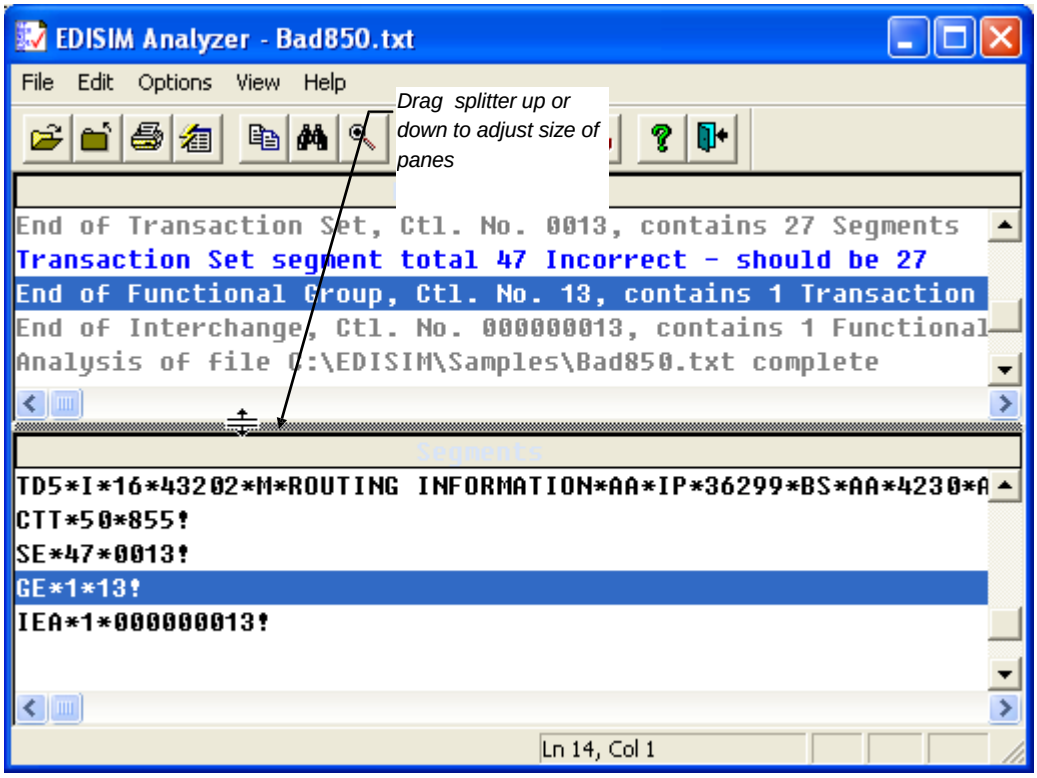
- Click anywhere in it with your mouse.
- Press **F6** to toggle between panes.
- Use the View menu:

  View | Diagnostics or Alt+d

  View | Segments or Alt+s

Sizing a Pane

To move the border (the "splitter bar") between panes, move your mouse cursor over it until it turns into double arrows and then drag the bar up or down.



To zoom the active pane so that it fills the entire window, press **F7** or double click on its header. To return a zoomed pane to its previous size, press **F7** again or double-click on its header again. The toolbar magnifying glass icon also toggles zoom on and off.

What do the Diagnostics Mean?

The diagnostics are color-coded according to their severity. You can configure the severity and colors of diagnostics, as explained in **Customizing your Analysis** on page 25.

You can see a list of diagnostic messages and explanations in these places:

- In the generic word processor file **Anamsgs.rtf** in EDISIM's BIN folder. You may wish to copy this file to another name and add your own notes to it.
- Near the bottom of the Contents tab under **Help | Index** in Analyzer.
- In the section Diagnostics (page 49).

Copying

After analyzing, you can copy one **selected** diagnostic or segment from the active pane to the Windows clipboard with:

- **Edit | Copy**
- *Ctrl*+**c**
- Toolbar button: 

From the clipboard, the copied diagnostic or segment can be pasted anywhere that uses the Windows clipboard: Notepad, WordPad, most Windows word processors, most e-mail programs, and many other products.

Example segments for Standards Editor: To copy a segment from Analyzer and then paste it into a level note in Standards Editor:

1. Copy the segment from the bottom pane of Analyzer.
2. Highlight the segment in Standards Editor.
3. Click the edit note line in the bottom detail pane.
4. Select the note level from the drop list.
5. Click anywhere in the large white note area.
6. Press *Ctrl*+**v**, or **Edit | Paste**.

Printing

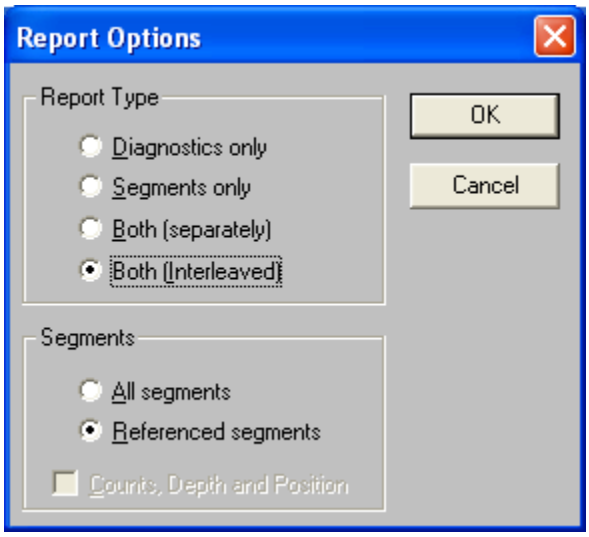
After analyzing, you can print the results to a printer or to a file.

To print a report:

- Choose **File | Print**…
- Type *Ctrl*+**p**…
- Or click the printer icon on the toolbar.

Report Options

The Report Options dialog box appears when you choose **File | Print** or **File | Print to File**. It lets you choose what to print.



For Report Type, choose one of these:

Diagnostics only
The diagnostics in the top pane.

Segments only
The data segments in the bottom pane.

Both (separately)
The contents of the top pane and then the contents of the bottom pane.

Both (Interleaved)
Segments and diagnostics, with each diagnostic printed just after the segment to which it refers.

Then, for Segments, choose one of these (which only have an effect if Report Type is Segments, Both, or Interleaved).

All segments
Every segment, regardless of whether it had a diagnostic.

Referenced
Only segments that have an associated diagnostic.

Tip: Interleaved and Referenced Segments are a good combination.

When finished, choose **OK**.

When you are printing to a file, the Report Options dialog box contains an option for Counts, Depth and Position (see Printing to a File below).

Printing to a File

File | Print to File lets you save the report in a plain text file. This selection:

1. Displays the Report Options dialog box described in Report Options above.

   The Counts, Depth and Position option is active when printing to a file and is described below.

2. Displays a dialog box that asks for page size, margins, etc. The last two options, "Eject last page" and "Use Formfeeds," can get the file ready to print on certain printers. "Eject last page" inserts linefeeds to fill up a page, in case you print the file. It calculates the number that will be required, considering the margins. The "Use Formfeeds" option inserts a formfeed at the end of the page.

3. Displays a Windows file dialog box that lets you choose the name of the file where the information is to be stored.

   You cannot append to an existing file.

Counts, Depth and Position

If you choose **File | Print to File** and select the **Counts, Depth and Position** option, you get a file that shows five additional columns. You must be using interleaved for this option to work. The resulting print file will be easier to understand if you select **Both (interleaved)** and **All segments**.

Prev

The Prev column is used for each loop or group trigger segment. It shows:

- The record number of the previous iteration of the loop or group.
- If there is no previous iteration of the loop or group, it shows the record number of the trigger (first) segment of the enclosing loop or group.
- If there is no enclosing loop or group, it shows the record number of the first segment in the document (the ST or UNH).

Examples of loops and groups that are not nested within other loops or groups

X12 Example (see page 22):

- See **A** in the example below, where the N1 trigger segment shows 3 in the Prev column. This is the record number of the ST segment, the first segment in the "parent" of the N1 loop. There is only one iteration of the N1 loop here.
- See **D** in the example below. The second iteration of the LM loop has a Prev of 16, which is the record number of the first iteration of the LM loop (see C).

EDIFACT Example (see page 22):

See **B** in the example below, where the first RFF trigger segment shows 3 in the Prev column. This is the record number of the UNH segment, the first segment in the "Parent" of the RFF group. The data contains three iterations of the RFF group, but each iteration consists of only the RFF segment. The second and the third consecutive RFF group show the record number of its predecessor.

- See **E** in the EDIFACT example. The second iteration of the NAD segment has a Prev of 10, which is the record number of the first iteration of the NAD group (see D).
- See **H** in the EDIFACT example, where the trigger segment of a LIN group shows 3 in the Prev column. This is the record number of the UNH segment, the first segment in the "parent" of the LIN group.

Examples of loops and groups that are nested within other loops or groups:

X12 Example (see page 21):

- See **E** in the X12 example, where a PO1 loop contains a CTP loop. The PO1 loop shows 3 in the Prev column. This is the record number of the ST segment. The ***first*** CTP loop shows 42 in the Prev column, the record number of the PO1 loop that contains it.
- For **E** in the X12 example, the ***second*** CTP loop shows 43 in the Prev column, which is the record number of the previous (and first) iteration of the CTP loop.

EDIFACT Example (see page 22):

- See **E** in the EDIFACT example, where a NAD group contains a CTA group. The trigger segment of the NAD group shows 3 in the Prev column, the record number of its "parent" - the UNH. The CTA group shows 10 in the Prev column, the record number of the NAD trigger segment of the containing group.
- See **F** in the EDIFACT example, which shows a RFF group embedded in a NAD group. The Prev of 15 is the record number of the containing NAD group's trigger segment.

Dp

The depth of the segment, loop, or group.

X12 Example (see page 21):

- The ST, BEG, and REF segments are all at depth 1 because they are not in a loop.
- The N1 segment is at depth 1 because it is the trigger segment in a non-nested loop.
- The N2 through REF segments are at depth 2 because they are non-trigger segments within the N1 loop.
- The CTP segment (Rcd 43 at Table 2, position 040) is at depth 2 because it is the trigger segment of a loop that is nested within the PO1 loop.
- The CUR segment at Rcd 44 is a depth of 3 because it is a non-trigger segment in a nested loop.

EDIFACT Example (see page 22):

- The UNH through DTM segments are at depth 1 because they are not in a group.
- The RFF segments are at a depth of 1 because they are trigger segments in non-nested groups. See **C** in the EDIFACT example.
- The NAD segment is at depth 1 because it is the trigger segment in a non-nested group. See **D** in the EDIFACT example.
- The LOC and FII segments are at depth 2 because they are non-trigger segments within the NAD group.
- The CTA segment is at depth 2 because it is the trigger segment of a group that is nested within the NAD group.
- The COM segment is at depth 3 because it is a non-trigger segment in the nested CTA group. This is also true of the DTM segment at **G**, which is a non-trigger segment in the nested RFF group.

Occur

The Occur column shows how many instances of the segments, loops, or groups have occurred so far at that location.

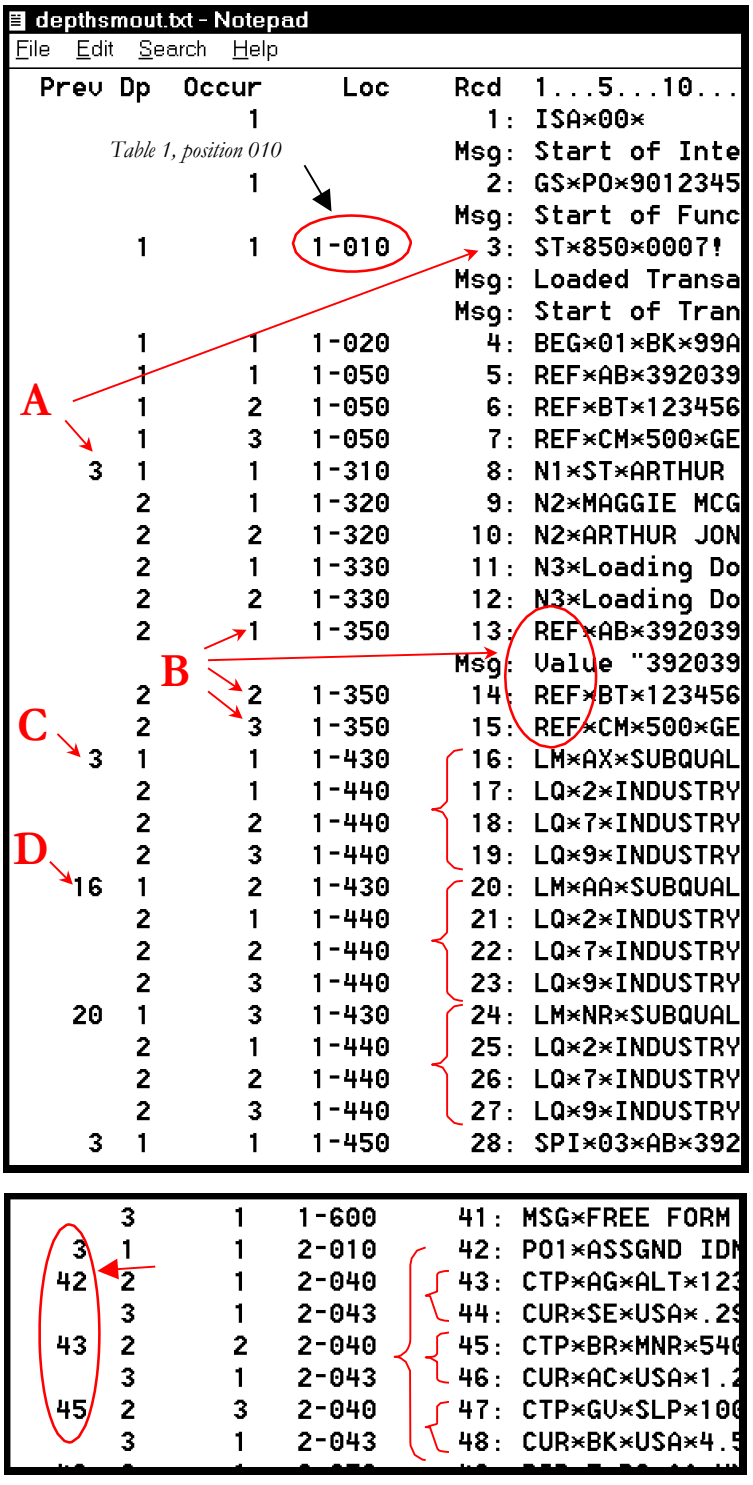
X12 Example (see page 21):

- See **B** in the X12 example, where three consecutive REF segments have 1, 2, and 3 in the Occur column.
- See **E**, where three consecutive CTP loops have occurrences of 1, 2, and 3.

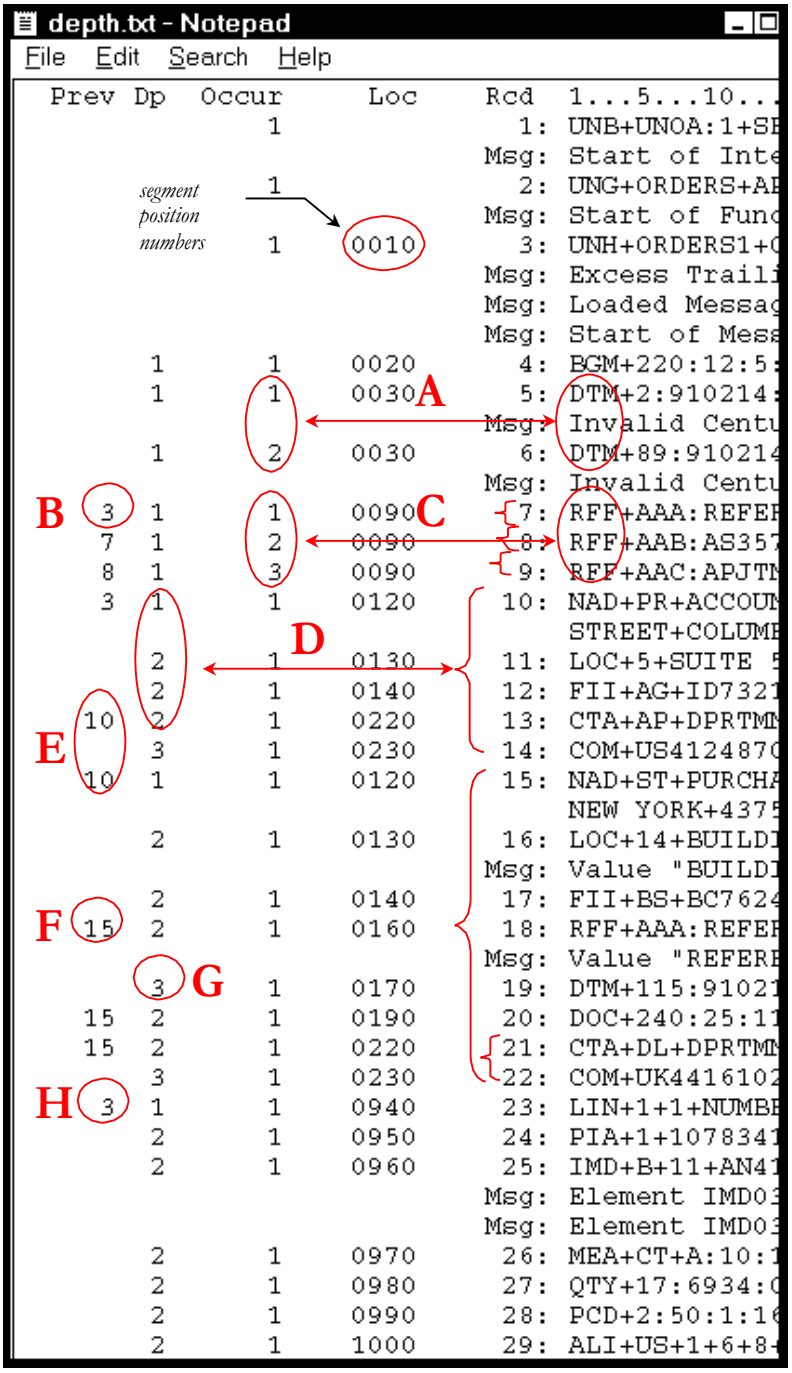
EDIFACT Example (see page 22):

- See **A**, where two consecutive DTM segments have 1 and 2 in the Occur column.
- See **C**, where three consecutive RFF loops (each containing only the RFF trigger segment) have 1, 2, and 3 in the Occur column.

X12 Example


depthsmout.txt - Notepad
File Edit Search Help
Prev Dp Occur Loc Rcd 1...5...10...
1 1: ISA*00*
Table 1, position 010 Msg: Start of Inte
1 2: GS*PO*9012345
Msg: Start of Func
1 1 1-010 3: ST*850*0007!
Msg: Loaded Transa
Msg: Start of Tran
1 1 1-020 4: BEG*01*BK*99A
1 1 1-050 5: REF*AB*392039
A
1 2 1-050 6: REF*BT*123456
1 3 1-050 7: REF*CM*500*GE
3 1 1 1-310 8: N1*ST*ARTHUR
2 1 1-320 9: N2*MAGGIE MCG
2 2 1-320 10: N2*ARTHUR JON
2 1 1-330 11: N3*Loading Do
2 2 1-330 12: N3*Loading Do
2 1 1-350 13: REF*AB*392039
B
Msg: Value "392039
2 2 1-350 14: REF*BT*123456
C
2 3 1-350 15: REF*CM*500*GE
3 1 1 1-430 16: LM*AX*SUBQUAL
2 1 1-440 17: LQ*2*INDUSTRY
2 2 1-440 18: LQ*7*INDUSTRY
D
2 3 1-440 19: LQ*9*INDUSTRY
16 1 2 1-430 20: LM*AA*SUBQUAL
2 1 1-440 21: LQ*2*INDUSTRY
2 2 1-440 22: LQ*7*INDUSTRY
2 3 1-440 23: LQ*9*INDUSTRY
20 1 3 1-430 24: LM*NR*SUBQUAL
2 1 1-440 25: LQ*2*INDUSTRY
2 2 1-440 26: LQ*7*INDUSTRY
2 3 1-440 27: LQ*9*INDUSTRY
3 1 1 1-450 28: SPI*03*AB*392
3 1 1-600 41: MSG*FREE FORM
3 1 1 2-010 42: PO1*ASSGND IDN
42 2 1 2-040 43: CTP*AG*ALT*123
3 1 2-043 44: CUR*SE*USA*.2
43 2 2 2-040 45: CTP*BR*MNR*540
3 1 2-043 46: CUR*AC*USA*1.2
45 2 3 2-040 47: CTP*GV*SLP*100
3 1 2-043 48: CUR*BK*USA*4.5

EDIFACT Example



Choosing and Setting up a Printer

To see which printer Analyzer will use, choose **File | Print Setup**.

This is a typical Windows Print Setup dialog box.

After choosing a printer, subsequent Analyzer print jobs will go to the printer that you selected. This setting lasts until you exit Analyzer.

The View Menu

The View menu lets you change what is displayed on Analyzer's screen.

Toolbar The toolbar is the set of icons at the top left, just below the menus. By default, it displays on the screen. It will have a check mark next to it on the View menu if it is displayed. You can toggle the check mark with a mouse click or by dropping down the View menu and then pressing **T**.

Status Bar The status bar is the bottom line on the Analyzer screen. It shows informative messages, line number and position number. By default, it displays on the screen. It will have a check mark next to it on the View menu if it is displayed. You can toggle the check mark with a mouse click or by dropping down the View menu and then pressing **B**.

Diagnostics Choosing Diagnostics from the View menu is one way to make the Diagnostics (top) pane active. The **F6** key also toggles between the Diagnostics and the Segments pane. If you have a mouse, clicking anywhere in the Diagnostics pane will make it active. *Alt*+**d** also activates the Diagnostics pane.

Segments Choosing Segments from the View menu is one way to make the Segments (bottom) pane active. The **F6** key also toggles between the Diagnostics and the Segments pane. If you have a mouse, clicking anywhere in the Segments pane will make it active. *Alt*+**s** also activates the Segments pane.

Diagnostic Message Types Choosing Diagnostic Message Types from the View menu lets you display or hide certain message types. You can do this under **Options | Analyzer Profile | Filter** also.

Zoom Choosing Zoom from the View menu makes the active pane (Segments or Diagnostics) as large as possible. Zoom is a toggle, so that choosing it again will restore the pane to its previous size. Other ways to toggle the zoom for the active pane: **F7** and the toolbar magnifying glass: 

Closing an EDI File

An open EDI file closes automatically when you open another file or exit Analyzer. You have two other ways to do this:

- Choose **File | Close**, or ...
- Click on the closing folder on the toolbar: 

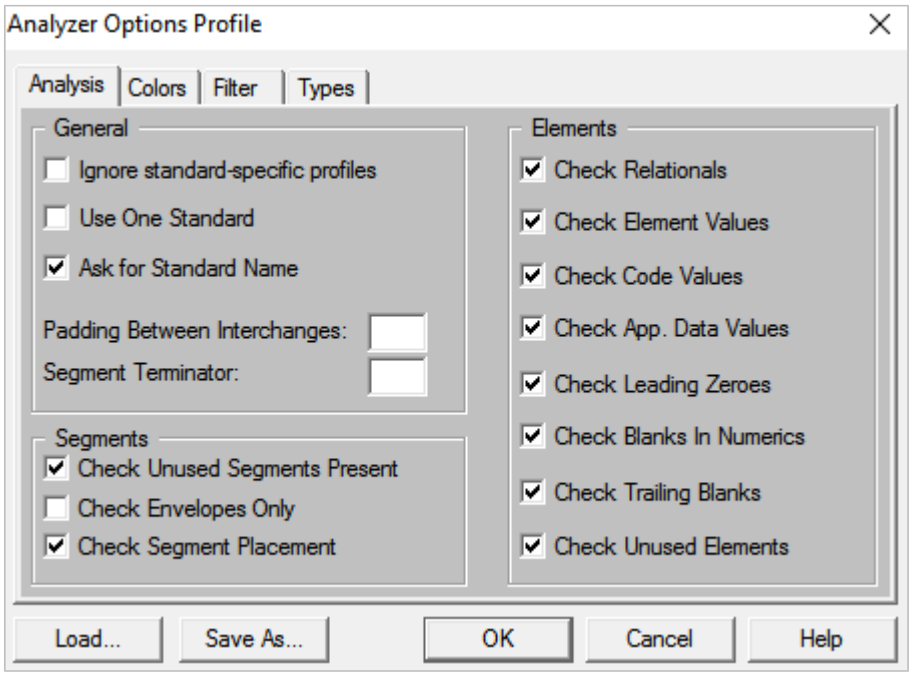
4 Advanced Topics

Customizing your Analysis

To Do This…	Use This…	See …
Select conditions to be checked	**Options | Analyzer Profile**	Analysis Options below
Suppress display of certain diagnostics	**Options | Analyzer Profile | Filter**	Page 28
Change diagnostic colors	**Options | Analyzer Profile | Colors**	Page 28
Change a diagnostic's type	Right click on diagnostic and choose Diagnostic Message Type or **Options | Analyzer Profile | Types**	Page 31
Save and reuse settings	**Options | Save Profile** **Options | Load Profile**	Page 32
Set up an automatic profile for a specific standard, guideline, or MIG	**Options | Save Profile**	Page 33

Analysis Options

To modify how Analyzer processes the EDI data file, choose **Options | Analyzer Profile** and change the options on the Analysis tab. These are:



Ignore standard-specific profiles

You can set up profiles that automatically activate when using specific standards, guidelines, or MIGs (see Standard-Specific Profiles on page 33). This option prevents Analyzer from automatically using these profiles. Default: off.

Use One Standard

This option controls whether a new standard, guideline, or MIG can be selected whenever a new functional group is encountered. Normally, each functional group may be analyzed against a different standard, guideline, or MIG. The one used depends on the state of the "Ask for Standard Name" option. If the "Use One Standard" option is selected, the entire transaction file will be analyzed against the first standard, guideline, or MIG selected. Default: off.

Ask for Standard Name

This option controls whether Analyzer will ask you to choose a standard each time a new functional group is encountered. If this option is selected, you will be asked each time; otherwise, you will be asked only if there is an ambiguity about which standard, guideline, or MIG to use. However, if you select "Use for all occurrences of_____" when you select a standard, guideline, or MIG, you will not be asked again until Analyzer finishes with the file. Default: on.

Padding between Interchanges

Sometimes a translator places padding characters (blanks, etc.) after the IEA or UNZ to fill it out to a certain length. If so, type it here. This can be a keyboard character or hex. If you use hex, type a lowercase x before the value as in `x0D`. Default: empty.

Segment Terminator

Analyzer usually finds the identity of the segment terminator from the interchange header. If the interchange header contains an error or is not the normal one, you should include the terminator here (type its hex value (preceded with x) or press the character on your keyboard). Otherwise, Analyzer will not be able to determine where segments end, and the analysis will be invalid. Default: empty.

Check Unused Segments Present

If this is selected, you'll get a warning if a segment is present in the data, but the guideline or MIG says it's unused. Default: on.

Check Envelopes Only

Only interchange enveloping, group enveloping, transaction set or message headers and trailers. For each guideline or MIG, Standards Editor's **Properties** dialog box specifies where the enveloping definitions reside. Default: off.

Check Segment Placement

If this is selected, Analyzer checks each segment as defined at a particular location within a transaction set or message. Otherwise, Analyzer can only check segments against the generic dictionary definitions. Default: on.

Check Relationals

If this is selected, Analyzer will check for element relationals such as "If you use this element, you must also use element xxx." X12 users call these syntax rules, and EDIFACT users call them dependency notes. User-defined business rules ("If you use code XXX in the BEG02, then you must use this N1 loop") are always checked, regardless of this option's setting. Default: on.

Check Element Values

If this is selected, Analyzer will check every element value to be sure it is the correct type and length. Default: on.

Check Code Values

If this is selected, Analyzer will check the data's code values to be sure they are actually on the list of allowed code values. Default: on.

Check App. Data Values

If this is selected, Analyzer will check data for elements that have application values, and to be sure the value is actually on the list. Application values are case sensitive. They can be set up in Standards Editor to provide additional value testing beyond code value checking. For

instance, they let you provide a list of acceptable values that may contain lower case, special characters, or very long values. If Analyzer encounters a value that is not on the list, in an element that has an application value list attached, you will receive a diagnostic message. Default: on.

Check Leading Zeroes

If this is selected, then you'll get a warning if you have leading zeros in a numeric field. Default: on.

Check Blanks in Numerics

If this is selected, then you'll get a warning if you have leading blanks in a real or numeric element. Default: on.

Check Trailing Blanks

If this is selected, you'll get a warning if you have trailing blanks in an alphanumeric field. Default: on.

Check Unused Elements

If this is selected, you'll get a warning if data is present for an element, but the guideline or MIG says the element is unused. Default: on.

OK

Exits the Analyzer Options Profile dialog box, putting the options into effect for the next analysis. The selected options remain in effect until you exit Analyzer or change them.

Load

Brings up an **Open Analyzer Profile** dialog box so that you can select a previously saved Analyzer profile (APF) file.

Save As…

Brings up a **Save Profile As** dialog box that lets you save the current settings to an external file for re-use later. Use file type **APF**. You can distribute this file to others who have the same version of Analyzer. It is not necessary to save the profile to use it in the current session.

Filtering and Coloring the Diagnostics

By default, all diagnostics are displayed, and have colors representing their severity level or "type." However, by using the **Colors, Filter,** and **Types** tabs in the Analyzer Options Profile dialog box, you can do the following:

- Change the color of certain diagnostic types.
- Suppress display of certain diagnostic types.

- Change the type of a diagnostic.

You can do this before or after an analysis, since it affects only how diagnostics are displayed, and not what characteristics of the file were actually checked.

Each diagnostic type has its own color, listed in the chart in Changing Diagnostic Message Types on page 31.

To customize the colors for each type, choose **Options | Analyzer Profile | Colors** tab. Click on a color that you want to change, and select from the color chart.

Other choices on the Colors tab:

Default Colors Resets the colors to the original settings.

No Colors Displays all diagnostics in black.

Save As Saves all profile settings to an external file that can be opened in another session. Save profiles to EDISIM's BIN folder for convenient opening later. You can give a copy of this file to others with the same version of Analyzer.

Load Lets you choose a file containing a saved profile.

Filtering

By default, all diagnostics display in the top pane. You can hide, or "filter," selected diagnostic message numbers or types.

To hide all diagnostic messages of a certain type

Right-click on a diagnostic that you want to hide and choose **Diagnostic Message Type**. Notice which type has the black dot.

Now that you know the type, you have two ways to hide it.

- Choose **View | Diagnostic Message Types** and deselect the type that you wish to hide. This is the easiest way to turn on or off a single type.
- Or, choose the **Options | Analyzer Profile | Filter** tab and deselect one or more types. This is the easiest way to turn on or off multiple types.

Either method allows you to toggle diagnostic types on and off.

To hide only one particular diagnostic number

- Identify the diagnostic's number by right clicking on it.
- Change that number to a type that you can stop displaying – such as Ignore, User #1 or User #2. This is described in Changing Diagnostic Message Types on page 31.
- Use **View | Diagnostic Message Types** to turn off display of that type.

Filtering and Coloring Example

1. **Analyze an EDI file**

   If you do not have one, you can use the example files and guidelines/MIGs provided by TIBCO Foresight:

   For X12: Check to be sure that you have a guideline called CLASS850. If not, go into Standards Editor, choose **File | Import | Import Single SEF and open**, and import **Class850.sef** from EDISIM's **AddlStds** folder. Then, enter Analyzer, open **Testpo.txt** in EDISIM's **Samples** folder, and analyze it against **CLASS850**.

   For EDIFACT: Check to be sure that you have a MIG called MYORDERS. If not, go into Standards Editor, choose **File | Import | Import Single SEF and open**, and import **Myorders.sef** from EDISIM's **AddlStds** folder. Then, enter Analyzer, open **Testpom.txt** in EDISIM's **Samples** folder, and analyze it against **MYORDERS**.

2. **Filter**

   Notice the gray diagnostics in the top pane. They are informative diagnostics pointing out filenames, enveloping segments, etc. Let's assume that we do not want to see gray diagnostics. First, we need to determine their type. **Right-click** on a gray diagnostic and choose **Diagnostic Message Type**. Note that its type is **Informational**. Close the popup menu.

   Now, we want to stop displaying informational (gray) diagnostics. Choose the **Options | Analyzer Profile | Filter** tab. Clear **"Informational" types** and choose **OK**. They are no longer visible.

3. **Change type**

   Find a diagnostic about a code value violation. Let's assume that we want these diagnostics to be pink. We do not want to change the blue color of other diagnostics. To do this, we have to change the Diagnostic Message Type for code value violations.

   Right-click on one in the top pane. From the pop-up menu, choose **Diagnostic Message Type** and then **User #1**. We are going to move these errors into the User #1 diagnostic type.

   Scroll down in the diagnostics pane and notice that diagnostics about code value violations are now green. They have been separated into their own color.

4. **Change colors**

   We want to change the color of the User #1 diagnostics. Choose the **Options | Analyzer Profile | Colors** tab and press the **dark green square** next to User Type #1. Choose **hot pink** and click **OK** twice.

5. **Save settings for future use**

Choose **Options | Save Profile** and enter a filename. Use file type APF.

Changing Diagnostic Message Types

Diagnostics are pre-assigned to one of these types:

Type	Default Color	Default Meaning
Ignore	light gray	For your use.
Informational	dark gray	Marks enveloping segments, cancellation of analysis, etc. Not an error.
Warning	olive	Minor EDI data errors such as leading zeros in numerics, or assumptions that Analyzer had to make when data was ambiguous.
Error	blue	Typical EDI data errors such as wrong codes, missing mandatory items, etc.
Fatal	red	Error that prevents analysis from completing.
User 1	dark green	For your use.
User 2	light green	For your use.

You have two ways to see or change a diagnostic's type.

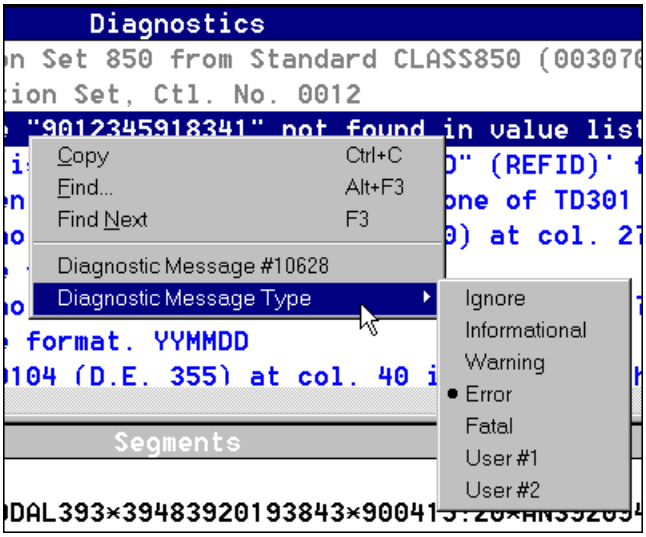
1. **Right-click** on the diagnostic in the top pane, and notice the Diagnostic Message #.

   Choose **Diagnostic Message Type**. The one with the dot is the current type.

   To change, click another type. This changes the type of all diagnostics with the same number.

   or …

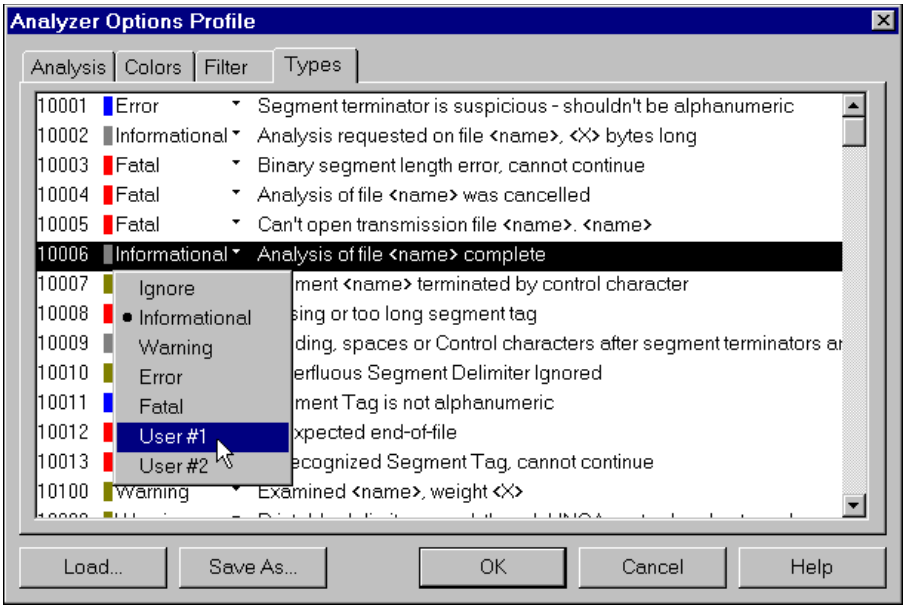
2. **Right-click** on the diagnostic in the top pane and note its Diagnostic Message number.



Close the pop-up menu.

Choose **Options | Analyzer Profile | Types** tab and scroll down to the diagnostic number.

To change, click on the diagnostic type (Error, Informational, etc.). From the popup list, choose another type.



Saving and Opening Profiles

You can save profiles for future use, load previously saved profiles, and create standard-specific profiles.

Saving a Profile

To save all current settings to a file for future use:

- Choose **Options | Save Profile**.
- Or, choose **Options | Analyzer Profile** and use the **Save As** button on any of the tabbed panels.

These methods open the Save Profile As dialog box, where you can specify a folder and filename. Analyzer will add file type APF.

Using a Saved Profile

To use a saved profile, choose **Options | Load Profile** or **Options | Analyzer Profile | Load**. Select the APF file that contains the profile that you want to use. By default, APF files are in EDISIM's BIN folder.

Changing a Saved Profile

To change a profile after it has been saved, open it as described in Using a Saved Profile above, make the changes under **Options | Analyzer Profile**, and resave it.

Setting up a Default Profile

When saving a profile, note the bottom line in the Save Profile As dialog box: "Save as the Default Profile for Analyzer."

If you select this, Analyzer will automatically load this profile each time it starts.

Restoring the Default Profile

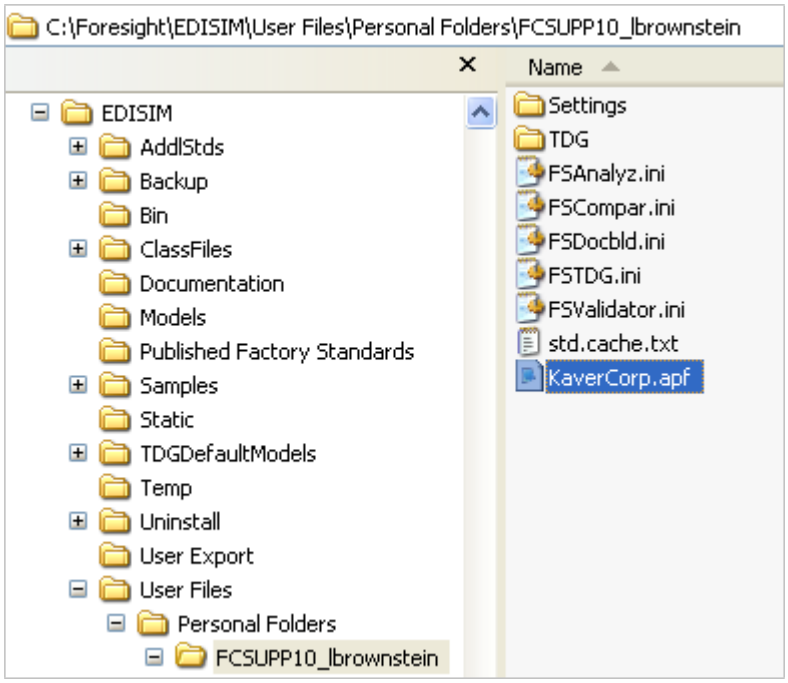
The TIBCO Foresight-supplied default profile that ships with Analyzer is $Fsdeflt.apf. If you have set up your own default profile, and you now wish to use TIBCO Foresight's default profile, choose **Options | Load Profile** to open **$Fsdeflt.apf** in EDISIM's **\Bin** folder. This will remain in effect until you choose another profile, change the settings, or exit Analyzer.

To again use the TIBCO Foresight default profile as your default, save it to a new name and select the "Save as the Default Profile for Analyzer" option.

Standard-Specific Profiles

Standard-specific profiles are profiles that have been saved in a specific location and with a specific name. To create one, save the profile ***with the same name*** as the standard, guideline, or MIG to which it applies, using file type APF.

Store it in your own personal folder under EDISIM's User Files folder. This profile will be used any time you analyze with your KaverCorp guideline:



If present, a standard-specific profile is automatically used with every analysis against its corresponding standard.

To prevent use of a standard-specific profile, choose **Options | Analyzer Profile | Ignore standard-specific profiles**.

A standard-specific profile will be used only until a new standard, guideline, or MIG is chosen. At that time, either a new standard-specific profile will be used (if applicable), or the session's current profile will be used.

Example

When analyzing against the guideline named 3040EAST, you never wish to be notified of "leading zero" violations. Therefore, you go into the Analyzer Options Profile dialog box and uncheck the option concerning leading zeros. You then use **Save As** to save this profile as:

EDISIM\User Export\<*your folder***>\3040EAST.apf**

From now on, whenever you analyze against 3040EAST, Analyzer will automatically use that profile and you will not be notified of leading zeros.

Then, one day, you *do* wish to hear about leading zeros when analyzing against 3040EAST. You go into the Analyzer Options Profile box and check "Ignore standard-specific profiles." Analyzer then uses the default profile.

You can modify and re-save standard-specific profiles just as you can any other saved profile.

Analyzing Ambiguous Loops, Groups and Segments

Assume that your data has two loops or groups with the same name but they are actually different. For example, you have five varieties of the HL loop in the X12 856 transaction set, each with different segments used. The first code in each HL loop is different: SHIPMENT, ORDER, PACK, TARE, and ITEM.

Analyzer has to be able to distinguish between:

- Consecutive segments with the same ID.
- Consecutive loops that have loop triggers (first segments) with the same ID.

When looking at EDI data, Analyzer checks the first *used* element with code values in the data's loop or segment. Unused elements, even if they have code values defined, are not used to differentiate between consecutive segments, loops, or groups. When Analyzer finds the code value in this element, it checks the same element in the standard, guideline, or MIG. When it finds a match, it uses that segment, loop, or group to check the data's compliance.

If it does not find a match, it checks the next element with code values and tries to match up the data to one of the segments or loops in the guideline or MIG. Analyzer continues comparing code values until it finds one that matches one of the segments or loops but not the other(s). Therefore, it is important to customize the code values in these elements when developing a guideline or MIG in Standards Editor.

Segment Example (X12)

Assume that your guideline has these segments:

050 REF (Max Used=3)

128 Reference Number	Code Value=07
127 Reference Number	Applic Values= 1120,1121
352 Description	

050 REF (Max Used =3)

128 Reference Number	Code Value=11
127 Reference Number	Applic Value=1220
352 Description	

Part of your data looks like this:

REF*__11__*1220*MAPRA NUMBER! **←1**

REF*__07__*1120*CURRENT VERSION NUMBER! **←2**

REF*07*1123*CUSTOMER VERSION NUMBER! ←3

How does Analyzer know which REF in the guideline to compare data against? Analyzer looks at the first code value in each REF (shown in bold in the example data above) and then matches it up with the code value list in the standard or guideline. When a match is found, it compares against that segment.

Analyzer will interpret the data above as follows:

- Analyzer sees the 11 in the data file's first REF, and then checks the code values in the guideline's two REF segments. It matches it up with the code value in the guideline's second REF at 050.
- Analyzer sees the 07 in the data's second REF, then compares against the guideline's first REF at 050.
- Analyzer looks in the third REF01 and sees the 07, then uses the first REF at 050 for comparison. During the comparison, it will notice that the data 1123 in REF02 is not on the application value list (1120 or 1121) in the guideline. An error message will be generated.

The REF segments can appear in any order, and still be matched up correctly, if the first codes for these segments are customized in the guideline.

Group Example (EDIFACT)

Assume that your MIG has RFF groups defined as follows:

0070 RFF group (Max Used=1) Required

RFF segment

C056 Composite with 1153 Reference Qualifier code value ALK, first 3 component data elements required.

DTM segment Conditional

0070 RFF group (Max Used=3) Conditional

RFF segment

C056 Composite with 1153 Reference Qualifier code values AAS, ACE, DM, first 2 component data elements required, other not used.

DTM segment not used

Part of your data looks like this:

```
RFF+AAS:T783'
RFF+DM:c1098555:300'
RFF+ALK:I991041:24'
DTM+3:991029:101'
```

How does Analyzer know which RFF in the MIG to compare data against? Analyzer looks at the first code value in each RFF and then matches it up with the code value list in the MIG. When a match is found, it compares against that segment.

Analyzer will interpret the data above as follows:

- Analyzer sees the AAS in the data file's first RFF, and then checks the code values in the MIG's two RFF segments. It matches it up with the code value in the MIG's second RFF at 0070. It would not have allowed a DTM after this RFF segment.
- Analyzer sees the DM in the data's second RFF, then compares against the MIG's second RFF at 0070. It notices that the third component data element was included, and issues an error message. It would not have allowed a DTM after this RFF segment.
- Analyzer looks in the third RFF and sees the ALK, then uses the first RFF at 0070 for comparison. It allows the following DTM segment, which is conditional in the ALK variation of the RFF loop.

The RFF groups can appear in any order, and still be matched up correctly, if the first codes for these segments are customized in the MIG.

Bounded Loops

X12 only

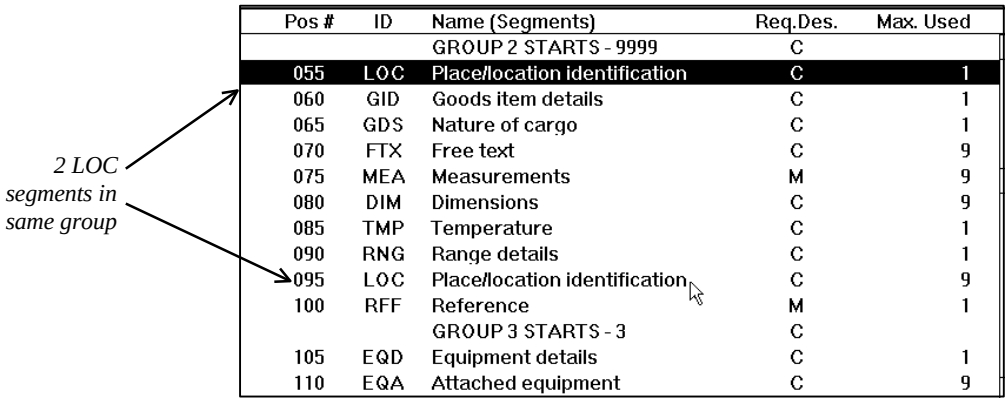
Some standards distinguish between two loops with the same name by surrounding one or both of them with LS (Loop Start) and LE (Loop End) segments. These are called bounded loops because they are bounded by LS and LE segments.

The one data element in the LS and LE segments contains a unique loop ID. If the standard or guideline uses bounded loops when defining a transaction set, the EDI data must use LS and LE segments around any such loop.

Group Segments: First Segment Repeated Later in a Group

EDIFACT only

For an example, look at the UN 92.1, BAPLIE message. LOC is the first segment in the group. It also appears at 095 later in the same group.



Pos #	ID	Name (Segments)	Req.Des.	Max. Used
		GROUP 2 STARTS - 9999	C	
055	LOC	Place/location identification	C	1
060	GID	Goods item details	C	1
065	GDS	Nature of cargo	C	1
070	FTX	Free text	C	9
075	MEA	Measurements	M	9
080	DIM	Dimensions	C	9
085	TMP	Temperature	C	1
090	RNG	Range details	C	1
095	LOC	Place/location identification	C	9
100	RFF	Reference	M	1
		GROUP 3 STARTS - 3	C	
105	EQD	Equipment details	C	1
110	EQA	Attached equipment	C	9

The second LOC is conditional (the first segment is always mandatory if the group is used). Therefore, after the first LOC, how will Analyzer know whether a particular LOC is the start of another group instance, or if it is the LOC at 095?

Analyzer interprets the above situation in this way: It checks for the presence of the mandatory segment RFF, which is after the second LOC in the standard. If it has occurred, then the LOC will be considered the first segment in a new group cycle. If it has not occurred, the LOC will be assumed to be the one at 095.

Ambiguous HL Loop Example (X12)

Your data may have two consecutive loops with the same name but they are actually quite different. For example, you may have "shipment" HLs and "order" HLs within an 856.

Analyzer can distinguish between such loops and analyze specifically for each type of HL. It looks in the first coded ID element of the loop: in the case of the HL, it checks the third element's code: is it an S or an O?

It then compares against that customized HL with that code in the guideline. This means that the guideline must have both an "orders" and a "shipment" HL loop, each with the appropriate code in the HL03. You can include customized loops when creating a guideline with Standards Editor. Industry guidelines such as VICS contain multiple definitions of the HL.

If your guideline included HL loops embedded within each other, the embedding must be done at the end of the enclosing loop. That is, no segment from the first loop can appear after segments from the second loop. Analyzer will enforce the nesting.

If your guideline includes HL loops appearing one after the other, Analyzer will understand the nesting by the code values in the first element of each HL.

Ambiguous N1 Loop Example (X12)

The guideline may contain two consecutive versions of the N1 loop. One might be a "Bill To" version and the other a "Ship To" version. These use different segments, elements, and values.

The first element in the N1 loop is a 2-character code for the type of N1: Ship to, Bill to, etc. In the guideline, the "Bill To" N101 element was set up with BT as the only legal code value, and the "Ship To" N101 has ST as the only legal code value.

Analyzer will analyze against the correct version of the loop if the code values have been customized in the guideline, and if the code value in the data matches that in one of the loops.

The alternative segments or loops must be contiguously defined – so be wary turning off the display of unused segments in Standards Editor.

Character Sets

Control Characters

If Analyzer finds a control character (hex 00-1F, 7F, and 80-9F) in data, it displays a "suspicious character" diagnostic. It does ignore these characters in binary objects, as in the BIN segment's element 785, which has data type B.

Control characters are acceptable in delimiters and separators.

EDIFACT Character Sets

Please see EDIFACT Enveloping and Syntax Levels on page 42.

Escape Characters

EDIFACT only

If the segment terminator, data element separator, or subelement separator appears in the data itself, it should be preceded with an escape character (usually a question mark) to

indicate that it is data rather than a separator. For instance, your segment terminator is a single quote. Your data is **`Bob's Restaurant`** so your data should read **`Bob?'s Restaurant`**.

To include a `?` in the data, use two consecutive `??`. To include `?` in the data, use two consecutive `??`. To include two `??`, use `????` since each `?` escapes only one character.

Analyzer handles escape characters correctly. (TDG does not automatically insert the escape character. This is so users can force errors if desired.)

To change the escape character from a question mark, the data file being analyzed should start with a UNA Service String Advice that contains a different escape character. See EDIFACT Enveloping and Syntax Levels on page 42.

Analyzing the Enveloping

Please see "What to do about Enveloping" in the Standards Editor documentation for details about how to set up custom enveloping for your guidelines and MIGs.

Enveloping Methods

When Analyzer starts reading your data file, it must try to determine what standard, guideline, or MIG to use for analyzing.

Analyzer will interpret the interchange control header and trailer (ISA and IEA for X12, or UNB and UNZ for EDIFACT) according to a generic format stored in the special enveloping standards X12ICS or UN1ICS. It then has enough information to find the VRI (Version, Release, and Industry Identifier Code) from the GS or UNG segment. Once it has the VRI, it looks in the list of guidelines, MIGs, and TIBCO Foresight-supplied standards to find one that matches. If there is any ambiguity, it will prompt for a standard to use for compliance checking. You can force Analyzer to *always* ask for a standard under the Analyzer **Options | Analyzer Profile** menu.

It then looks in the dictionary of the standard, guideline, or MIG used for compliance checking. If it finds group enveloping (GS and GE or UNG and UNE) there, it will check against these segments. If not, it checks against the generic group enveloping in X12ICS or UN1ICS.

You can Analyzer to analyze against Standard Editor's envelope settings in this way: before opening the data file, go into **Options | Analyzer Profile** and select "Use one standard." This method will use just one standard for the entire data file, regardless of how many transaction sets or messages it contains.

Default Envelope Handling

If Analyzer sees an ISA, it knows to expect X12 data and it loads a standard called X12ICS, which contains generic X12 enveloping. If it sees a UNA or UNB, it knows to expect EDIFACT data and it loads standard UN1ICS, which contains generic EDIFACT enveloping.

From the X12ICS or UN1ICS standards, it has enough information to find the VRI (Version, Release, and Industry Identifier Code) from the GS or UNG segment. Analyzer checks the interchange and group segments against those in X12ICS or UN1ICS. It now has the VRI, so it looks in the list of standards, guidelines, and MIGs to find one that matches. If there is any ambiguity, it will prompt for a standard to use for compliance checking.

Analyze Against Enveloping in Current Standard's Dictionary

Analyzer will look in the dictionary of the guideline or MIG for the ISA, GS, GE, IEA (X12) or the UNB, UNG, UNE, and UNZ (EDIFACT). It will compare the data's enveloping segments to these and report any deviations.

Analyze against Enveloping in a Separate Standard's Dictionary

Analyzer will automatically look in the *dictionary* of that enveloping standard for the ISA, GS, GE, IEA (X12) or the UNB, UNG, UNE, and UNZ (EDIFACT). It will compare the data's enveloping segments to these and report any deviations.

ICS Segment

X12 only

ICS is a predecessor of the ISA envelope, still used in certain industries. If Analyzer finds an ICS segment in the data, it will check it against a generic ICS segment that is predefined in Analyzer. Likewise, the corresponding ICE will be processed at the end of the interchange. ICS and ICE are not defined in any TIBCO Foresight-supplied standard.

The ICS segment is a fixed length record that doesn't use an element delimiter to separate elements. Each ICS element is mandatory and is a fixed length. The total length of the ICS, including the leading segment tag, and excluding the trailing segment terminator, is 67 bytes:

Elem		Description	Type/Min/Max/Pos
"ICS"(omit quotes)		Segment Tag	
ICS01	D.E. 700	Data Element Separator	AN 01/01 col 4
ICS02	D.E. 701	Sub-Element Separator	AN 01/01 col 5
ICS03	D.E. 726	Inter. Cntl. Standards Id	ID 04/04 col 6
ICS04	D.E. 703	Inter. Cntl. Version No.	ID 05/05 col 10

ICS05	D.E. 704	Inter. ID Qualifier	ID 02/02 col 15
ICS06	D.E. 705	Inter. Sender ID (Orig.)	ID 15/15 col 17
ICS07	D.E. 704	Inter. ID Qualifier	ID 02/02 col 32
ICS08	D.E. 706	Inter. Receiver ID (Dest.)	ID 15/15 col 34
ICS09	D.E. 707	Inter. Submit Date	DT 06/06 col 49
ICS10	D.E. 708	Inter. Submit Time	TM 04/04 col 55
ICS11	D.E. 709	Interchange Control Number	N0 09/09 col 59
ICS12	D.E. 702	Segment Terminator	AN 01/01 col 68

EDIFACT Enveloping and Syntax Levels

The UNB

The UNB starts with :

UNB+*syntax identifier*:*syntax version*+ … etc.

Example:

`UNB+`**`UNOA:1`**`+6145551212:12:B4+6145551234:12+991028:1430+1+++A+1++1'`

Syntax identifiers in the UNB can be:

`UNOA` (versions 1-4)

`UNOB` (versions 1-4)

`UNOC` (versions 3-4)

`UNOD` (version 4)

`UNOE` (version 4)

`UNOF` (version 4)

Syntax version numbers in the UNB can be `1,2,3,` or `4`

The syntax identifier and syntax versions interact as follows.

Syntax identifier	Syntax version	Character set	Default Delimiters	Escape character
UNOA	1, 2, or 3	A-Z, 0-9, some special characters	**:** IS 1 component data element **+** IS 3 data element **'** IS 4 segment'	?
UNOB - UNOF	1, 2, or 3	A-Z, a-z, 0-9,	hex	none

		some special characters		needed due to hex delimiters
UNO*x* (any)	4		: IS 1 component data element + IS 3 data element ' IS 4 segment'	?

Analyzer looks in the UNB for the syntax level and enforces these levels as specified by ISO 9735. For UNO*x*, Analyzer can enforce UNOC, which uses ISO 8859-1, Part 1 Latin Alphabet 1.

For UNOA and UNOB details, see http://www.unece.org/trade/untdid/texts/d422_d.htm, which is sponsored by United Nations Centre for the Facilitation of Procedures and Practices for Administration, Commerce and Transport

For UNOC character set details, view the charts at http://czyborra.com/charsets/iso8859.html.

The UNA

With all of the choices available in the UNB, why use the UNA?

The answer: there is no UNB combination above that allows all of these:

- lower case data (UNOB and later)
- syntax versions 1, 2, or 3
- and printable delimiters

The UNA Service String Advice comes to the rescue here by overriding the default delimiters, and by providing an escape character.

After the upper case characters UNA, there are six characters that indicate:

1. component data element separator
2. data element separator
3. decimal notation (comma or full stop in decimal numbers: 132.5 or 132,5)
4. release character (insert a space if not used)
5. a space (for future use)
6. segment terminator

The UNA applies to only one interchange – the one introduced by the UNB that immediately follows it. In this example, the UNA specifies printable delimiters for the UNB-UNZ segments that follow it. If another interchange had been added to the same file, the UNA would not apply to it.

```
UNA:+.? '
UNB+UNOB:2+SENDER+RECIPIENT+990601:1030+8801'
UNH+690+ORDERS:D:93A:UN:EAN007'
  .
  .
  .
UNT+20+690'
UNH+691+ORDERS:D:93A:UN:EAN007'
  .
  .
  .
UNT+12+691'
UNZ+2+8801'
```

Analyzer understands the UNA, and will interpret the EDI data accordingly.

EDISIM's TDG will automatically include a UNA in the test data file, if you change the terminators in the enveloping. You do not need to add the UNA to the MIG with Standards Editor.

Application Value Lists in Enveloping Segments

For enveloping segments, Analyzer must have application lists attached in the dictionary if they are to be checked.

Saving an Analysis

Saving

You can save an analysis and then later open it for immediate viewing. To save:

- Analyze the data.
- If the EDI Compliance Check dialog is still on the screen, click **OK**.
- Choose **File | Save Analysis**. Use file type .SAF.

Tip Saving an analysis creates 7 files (.SAF, .MIX, .MSF, .MSQ, .SIX, .TIX, and .TOP). You might wish to save them to EDISIM's Temp folder or another location where they will be easy to find and delete.

Viewing

When you view the saved analysis, you will need:

- The 7 saved analysis files.
- The original EDI file in the same location where it was analyzed.

To view:

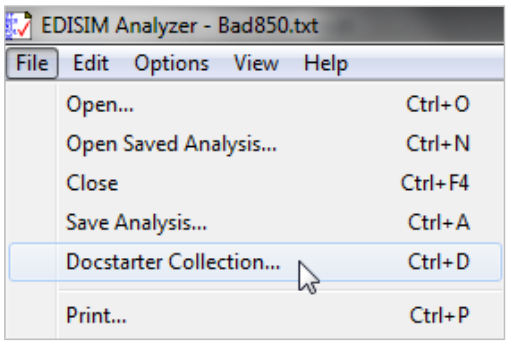
- From a blank Analyzer screen, choose **File | Open Saved Analysis**.
- Open the .SAF file.

The results should appear on the screen immediately.

5 DocStarter

Overview

DocStarter lets you start with one or more EDI data files and create an EDISIM guideline or MIG based on them.



The EDI files can contain one or more transactions or messages. Within each file, you can use any or all transactions or messages to build the guideline or MIG.

The resulting guideline or MIG will contain one or more transaction sets or messages, with segments, elements, and codes that are customized to match the data in the EDI file.

The guideline or MIG will be as complete and accurate as possible, based on the EDI data files. However, you will probably want to further refine it by editing it with Standards Editor, or by using DocStarter with more EDI files as you get them.

Tips

The more files you have, the better your new guideline or MIG will be.

In general, you will want to use only error-free EDI data to create your guideline or MIG.

For step-by-step details see **Docstarter.pdf** located in EDISIM's Documentation directory

.

6 Diagnostics

Explanations

The Analyzer help file contains explanations for diagnostic messages. These are listed by diagnostic number and also alphabetically by diagnostic text.

Diagnostic Explanations in Numeric Order

10001 Segment terminator is suspicious - shouldn't be alphanumeric

For X12 data, Analyzer looks at the last character in the ISA to identify the segment terminator. For EDIFACT data, it looks in the UNA. The segment terminator should be something that does not appear in the actual data.

10003 Binary segment length error, cannot continue

The BIN01 specifies the length in bytes for the binary data in the BIN02. However, end of file was reached before that number of bytes was supplied.

10005 Can't open transmission file *name*

The file may be in use by another application.

10007 Segment *xxx* terminated by control character

Is there a segment terminator and is it correct?

10008 Missing or Too Long Segment Tag

The segment tag (the segment's ID, which is the first item in the segment: ST, UNH, etc.) is incorrect. Check the tag and delimiters.

10009 Padding, spaces or Control characters after segment terminators are ignored

Look after the segment for blanks, tabs, or other extraneous characters.

10010 Superfluous Segment Delimiter Ignored

Does the file have two consecutive segment terminators?

10011 Segment Tag is not alphanumeric

Segment tags (the first characters in each segment: ST, BEG, UNH, etc.) must be alphanumeric and start with a letter. Also check nearby segment delimiters.

10012 Unexpected end-of-file

Analyzer stopped because it became lost when processing a segment.

10013 Unrecognized Segment Tag, cannot continue

The segment tag (characters before the first delimiter in the segment) may be numeric, too long, or empty. Processing stops.

10200 Printable delimiters used, though UNOA syntax level not used

This UNA specified binary delimiters. This is supposed to override delimiters specified in the UNB. However, the UNB uses printable ones like : + ' so Analyzer will assume that these are going to be used in the rest of the interchange. This error can happen if the data includes multiple interchanges in the same file, but include only one UNA. A UNA only applies to the UNB that immediately follows it. Example UNA and UNB:

```
UNA:+.? '
UNB+UNOB:1+012345678100001+987654321399999+950322:1431+0+++++TYPE TEST 202R-01+1'
```

10201 Could not interpret functional group header segment

Analyzer could not understand the GS or UNG in the data. Analyzer looks in one of two places for a GS or UNG definition: in the dictionary of the standard or guideline that contains the transaction set/message or, if it isn't there, in the dictionary of the interchange standard or guideline.

The interchange standard, by default, is either UN1ICS or X12ICS. It may be another standard or guideline if the user chose "Other Standard" when saving the transaction(s)/message(s) or setting its properties in Standards Editor.

10202 Encountered Functional Group Header outside of Interchange

This functional group envelope header (GS or UNG) does not appear to be within an interchange (ISA-IEA or UNB-UNZ). At present, groups can only be included within interchanges. This is because Analyzer uses the interchange envelope to verify delimiters and separators, as explained below. For details on how Analyzer determines where to look for enveloping, please see "enveloping segments" in the index of the Standards Editor or Analyzer manuals.

X12: The functional group header segment GS points to the application standard under which the GS itself should be interpreted. But Analyzer cannot break apart and interpret the GS in order to extract GS08, the Version-Release-Industry Code, without knowing its

structure. It needs to have a definition of the GS in the interchange standard (either in the one selected by the user, or in the default X12ICS interchange standard). Once Analyzer finds the GS08, it uses the standard listed there, which describes the transaction sets or messages. Newer application standards have a GS in the segment dictionary, and Analyzer uses it to actually validate the GS. Otherwise, the interchange standard is used to validate the GS.

EDIFACT: Analyzer uses a similar process to the X12 explanation above. If it cannot find the UNG in the dictionary of the standard that contains the transaction set/message, Analyzer uses the interchange standard (UN1ICS or the one chosen during Standards Editor's Save As / Properties) to check the syntax of the UNG.

10203 Functional Group ID "*xx*" mismatch - <*Transaction set or Message*> is type "*yy*"

Examples*:*

"Functional Group ID "BANSTz" mismatch - Message is type "BANSTA." " The message identifier in the UNG erroneously contained "BANSTz."

"Functional Group ID "PQ" mismatch - Transaction Set is type "PO." "The GS01 erroneously contained "PQ" but the ST01 contained "850."

EDIFACT: The contents of the UNG Message Type Identifier and the UNH Message Type Identifier are different. Example:

```
UNG+BANSTz+APPL SENDER ID:12+.. etc.
UNH+BANSTA1+BANSTA:92:1:UN:+COMM ACCESS REF… etc.
```

If no UNG-UNE segments are used, then different EDIFACT message types may be in a single interchange.

X12: The contents of GS01 (Functional Identifier Code) do not correspond to the ST01 (Transaction Set Identifier Code) as required by ASC X12. An exception is the 980 Functional Group Totals, which assumes the functional group ID of the containing functional group. In this case, TDG's FuncID function will return "" and Analyzer will not display a message about the difference. This message also appears if the GS is missing from both the application standard (the one that contains the transaction set) or in the interchange standard (X12ICS or the one chosen when saving as / setting properties in Standards Editor).

10204 Can't Interpret Control Segment *xxx* - Missing from Standard *yyy*

This serious error occurs when the enveloping standard could be found, but its dictionary does not include the interchange or group enveloping segments, or when the application standard's dictionary does not include the document header (ST or UNH). Check that X12ICS.STD or UN1ICS.STD is in the Static folder and contains these segments in the dictionary, or contact TIBCO Foresight technical support.

10205 ICS must be exactly 67 bytes long, excl. segment terminator

The ICS is a fixed length record, which doesn't use the element delimiter to separate elements.

10207 Invalid Syntax identifier - UNO*x*, where *x* is alpha, expected

To determine the allowable character sets, Analyzer looks at the UNB01-01 (Syntax Level). It sets the syntax level to the last alpha character in the 4-character code … usually A, B, or C. This in effect determines the default separators and delimiters. If a UNA segment precedes this UNB, then the delimiters set there will be used instead of whatever is specified in the UNB.

10208 Can't interpret segment without proper Envelopes

This segment is not enclosed in recognizable interchange, functional group, or transaction set/message enveloping segments. Check all envelope pairings. For EDIFACT, these must be interchange (UNB and UNZ) and message (UNH and UNT) headers and trailers. If included, UNG and UNE must be paired. For X12, these must be interchange (ISA and IEA), functional group (GS and GE), and transaction set (ST and SE) headers and trailers.

10209 Functional Trailer Group ignored - No corresponding func grp exists

This GE or UNE does not appear to have a corresponding GS or UNG.

10211 Skipping for the start of interchange

Are all enveloping segments present and correct? This diagnostic can appear if Analyzer becomes lost during an analysis and cannot realign itself. It skips the rest of the interchange in which it became lost.

10212 Could not load standard *xxx* to interpret <*ISA or UNB*> interchange header segment

Examples:

X12: *" Could not load standard X12ICS to interpret ISA interchange header segment."* This message appeared on the ISA when the default enveloping standard X12ICS.STD was not available in the Edisim50\Static folder.

EDIFACT: *"Could not load standard UN1ICS to interpret UNB interchange header segment."* This message appeared on the UNB when the default enveloping standard UN1ICS.STD was not available in the \Edisim50\Static folder.

Explanation: Analyzer cannot load the standard that contains the Interchange and Functional Group envelopes. When the guideline (or "local standard") was saved in Standards Editor, the user chooses where Analyzer should look for its enveloping (via Standard Editor's **File | Save As | Properties**). If the user chose "Default," then Analyzer looks in standard UN1ICS (EDIFACT) or X12ICS (X12) for the enveloping segments. These standards must be in EDISIM's STATIC folder. If **Other Standard** was chosen when saving from Standards Editor, the user specifically selected a standard where Analyzer should look for enveloping. This standard must be available as an .STD file in EDISIM's Static or User Files / Public Guidelines directories and must be readable (plain text) by Analyzer. If Analyzer could not load the standard, then it could not interpret either the interchange envelope header or the functional group header. For more information, look under "Enveloping" in the index of the Standards Editor or Analyzer manuals.

10213 No corresponding Header for <*Transaction Set or Message*> Trailer

This document trailer does not appear to have a corresponding document header. There are either too few document headers or too many document trailers.

10214 UNA has no meaning except immediately before UNB

The UNA cannot appear *within* an interchange.

10217 UNA conflicts with delimiters used in UNB

The delimiters in the UNA do not match what is actually being used in the UNB and with the contents of the first element in the UNB. Since they don't match, Analyzer is going to use whatever is in the 4th character position of the UNB for the segment delimiter and whatever is at the end of the UNB for the segment terminator. The sub-element separator that is being used will be shown in another diagnostic message.

10218 Could not interpret UNA - ignored

Analyzer needs to interpret the UN/EDIFACT Service String Advice (UNA) at least as far as the end of the release indicator field (see below). The last field of the UNA is reserved. When included, the UNA must appear immediately before the Interchange Header (UNB) segment and begin with the upper case characters UNA followed by six characters selected by the sender to indicate, in order:

Type Length	Req	Description	Notes
an 1	M	COMPONENT DATA ELEMENT SEPARATOR	
an 1	M	DATA ELEMENT SEPARATOR	
an 1	M	DECIMAL NOTATION	Comma or full stop
an 1	M	RELEASE INDICATOR	If not used, insert space character
an 1	M	Reserved for future	Insert space character
an 1	M	SEGMENT TERMINATOR	

Example: `UNA:+.? '`

10219 UNA is too short

The UN/EDIFACT Service String Advice (UNA) is a fixed field record of length 8 plus the segment terminator. It may appear before a UNB to override the default delimiters given in the first element of the UNB. A UNA only has meaning right before an interchange header, and it only affects the interchange that immediately follows it. If the transmission contains more than one interchange, a single UNA will only override delimiters for one of them. Without the UNA, default delimiters come from the first element in the UNB, or from the delimiters that are actually being used in the UNB.

Example of a UNA: **`UNA:+.? '`**

For a description of the UNA layout, see the message 10218 Could not interpret UNA - ignored.

10220 Missing Functional Group Header

A GS may be missing or out of order. EDIFACT doesn't require functional group envelopes, since you can deduce the application standard from the UNH segment. X12 cannot tolerate missing functional groups, since the VRI (Version-Release- Industry Code) only exists in the GS segment.

10221 UNA overrides previous UNA

The data contains two consecutive UNA segments. Each UNA segment must be immediately followed by a UNB and will affect only that interchange, even if there are other interchanges in the same file.

10222 Unsupported Interchange Type

Analyzer does not recognize the ID of the interchange header segment, which determines which standard to use. Analyzer supports ISA, ICS, UNB, and BG.

10500 Couldn't read XrefSeg at *xxx*

xxx is a hex address. An essential resource is unavailable. Is the disk drive full?

10600 Value "*xxx*" is an invalid '*name*' for *name*

This diagnostic is often paired with message 10617.

The data is invalid for this kind of field. For example, the value may contain leading zeros. Analyzer will suggest a possible value.

10601 Application Value "*xxx*" not found in value list "*listname*" for <*element and location*>

Explanation: An application values list called *listname* has been attached to this element at this location. The value *xxx* does not appear on that list (application values are case sensitive so capitalization makes a difference).

Application values are attached in Standards Editor and can be viewed in Standards Reference and Editor. Please see "Application Value Lists" in the SE manual. Checking application values is optional. Look under **Options | Analyzer Profile | Check App. Data Values** in the Analyzer menus.

Example: *"Application Value "MAGGIE MCGILL" not found in value list "cust1" for N102 (D.E. 93) at col. 7."* This message appeared because the 2nd element in the N1 at this location (N1**02**, data element 93) has an application values list attached, but that list does not include the value "MAGGIE MCGILL".

Example involving Business Rule: "*Application Value "ARTHUR JONES" not found in value list "HOSPITAL*" which is used for N102 (D.E. 93) at col. 7 because DoctypeBEG02'EQ'BL" This message appeared because the N102, element 93, contains ARTHUR JONES. This value is not in the Application Value List HOSPITAL, which is attached when the element pointed to by variable DoctypeBEGO2contains a BL. A business rule attached by a Standards Editor user is enforcing this.

10602 Assuming Sub-Element Delimiter is *x*, rather than *y*

Examples:

(EDIFACT): "*Assuming Sub-Element Delimiter is '@', rather than ':'."* The UNA gave the sub-element delimiter as ":". However, the UNB used @ to separate sub-elements. Analyzer is assuming that the rest of the file uses @.

(X12): *"Assuming Sub-Element Delimiter is ':', rather than '@'" The* last element in the ISA (just before the segment terminator) contains "@", but the current segment appears to be using the colon for a sub-element delimiter.

Explanation: Even though the ISA, UNA or appropriate defaults may have been used to supply the sub-element delimiter, it's likely that an altogether different character is actually being used to separate sub-elements within a composite data element.

For X12, check the ISA and then the segments that have composites.

For EDIFACT, check the UNA (if present), the UNB, and the segments that have sub-elements.

10603 Expected sequence number *n* (repetitions of *yyy*) at *location*

Example: *"Expected sequence number 1 (repetitions of DNA) at DNA01 (D.E. SEQA) at col. 5"*

TRADACOMS only. Repeating segments, or segments within loops, will have sequence numbers at the front of the segment explicitly giving the repetition occurrence. For example, a single repeating DNA segment in the Header section of Product File will have a single SEQA in position 1, which will have the repeat count of the segment. Likewise, the loop trigger of the NOI loop in the detail section will have a single SEQA giving the repeat count of the NOI loop. Nested loops and segments immediately contained within the NOI loop will have both SEQA and SEQB - SEQA is the same as SEQA in the NOI segment, and SEQB is the occurrence of the loop or segment, respectively. This pattern of sequence numbers is repeated up to 4 times (SEQA through SEQD), accommodating that much nesting.

10604 Blank Value supplied for *name*, should be omitted

The optional element contains blanks, but it should be omitted entirely.

10605 *and* 10606 Code Value xxx not <*used or found*> for <*element and location*>

Example: *"Code Value "850" not used for ST01 (D.E. 143) at col. 4."*

Code value 850 is marked as not used in the guideline or industry subset for the ST's 1st element (ST**01**, data element 143), although it may be a valid code for the underlying X12 standard.

If the diagnostic says *xxx* was not **used**, it is in the dictionary for this segment and element, but is marked as not used at the current location. If the diagnostic says *xxx* was not **found**, the code value was not in the dictionary for this segment and element. In the example, "Col. 4" means the code 850 starts in the fourth character position in the segment, after "ST*".

10605 and 10617 Code Value *xxx* not found for *element* - possibly *xxx* was intended.

Example: *Code Value "3" not found for REF02 (D.E. 127) at col. 8 - possibly "03" was intended.*

Numeric code value 3 was used in the EDI data, but not found in the dictionary for that element. Analyzer does find 03 in the dictionary, however, and suggests that leading zeros might have been accidentally omitted in the data.

10607 Element or Composite *xxx* not found

All elements and composites must be in the standard's dictionary, if segments refer to them. This would probably indicate an error in a TIBCO Foresight supplied standard, or dictionary elements were accidentally deleted from a user guideline from within Standards Editor. It most likely is not an error in your data, but rather the standard or guideline.

10610 Excess Data Elements after <*element and column*>

Example*: "Excess Data Elements after NTE02 (D.E. 352) at col. 9"*

By looking at the element and sub-element delimiters, Analyzer concludes that there are too many elements in this segment's data. Check that the data itself does not contain the delimiter character, and check that a segment terminator has not been omitted.

10611 Excess trailing <*element or composite sub-element*> delimiters after *xxx* at col. *nn*

Example:

"Excess trailing delimiters after PO110 (D.E. 235) at col. 43"

"Excess Trailing Composite Sub-Element Delimiter(s) after UNH02-S00904 (D.E. 0051) at col. 26"

Explanation: Analyzer is going to ignore the rest of the segment or composite. Look at the location listed for extra delimiters or separators.

10612 Leading Blanks in *name*

Leading blanks are significant and allowed for alphanumeric elements, but Analyzer issues warning messages for them in R or N elements.

10613 Element *xxx* at col. *nn* is missing, though marked "*usage*"

Example: "*Element BGN05 (D.E. 623) at col. 34 is missing, though marked "Must be Used.""* This diagnostic means the 5th element (05) in the BGN, which is data element 623 (D.E. 623), contains no data although the industry standard or guideline says it must be used.

Explanation: According to the industry standard or guideline, the element must be used at this position in this segment in the transaction set or message, but no data is present for this element. For an explanation of the differences between Mandatory and Must be Used, please see "Overview of Requirement Designators and User Attributes" in the Standards Editor manual.

The "because" reason will appear in the message if the element must be used due to a business rule.

10614 Missing Mandatory *item* at *location*

Examples:

"Missing Mandatory Loop HL at 2-010"

"Missing mandatory LIN02 (D.E. 235) at col. 6"

According to the published standard, a mandatory loop, group, segment, or element should appear at the *location* shown (in the top example, it should be in Table 2, position 10; in the bottom example, it should be the second element in the LIN segment).

10617 possibly "*name*" was intended

This is not a stand-alone diagnostic, but appends to the end of another diagnostic such as #10600.

10618 Element *xxx* is present, though marked *usage*

Example: *BGN04 (D.E. 337) at col. 29 present, though marked "Not Used."* The 4th element in the BGN segment, which is data element 337, was marked as not used in the guideline. However, it is included in the data, and starts in column 29.

Explanation: According to the guideline or industry subset (VICS, UCS, etc.), this element is not to be included in the data. This may differ from the requirement in the underlying X12 standard. If you do not wish Analyzer to notify you about this condition, look under **Options | Analyzer**.

The "because" reason will appear in the message if the item was not used due to a business rule.

10621 Conditional Element Relation violated: *condition*

Example: *"Conditional Element Relation violated: Only one of MEA08 or MEA03 may be present."*

A conditional relation (syntax rule) has been violated.

From Standards Editor or Standards Reference, you can see a segment's syntax rules by highlighting the segment in the segment dictionary and then clicking the Edit Rules button in the details pane. If the syntax rule is on a composite, highlight the composite in the composite dictionary and then click the Edit Rules button.

10622 Segment *xxx* doesn't exist in standard *yyy*

Examples*:*

"Segment CUL doesn't exist in standard 003042." The two most likely causes are that CUL should not be in the data, or you are analyzing against the wrong standard.

"Segment UNG doesn't exist in standard (UnNamed)." This example resulted when the UNG was in the default enveloping standard UN1ICS, which Analyzer could not read because it had accidentally been turned into a word processor file.

Explanation: Is the segment tag correct? Are you analyzing against the correct standard? You can require Analyzer to ask you for a standard before each transaction set or message under **Options | Analyzer | Ask for Standard Name.**

10623 Sub-element separator seen in elementary data element at *location*. Excess ignored.

There should not be a sub-element separator at this point. It is being ignored. Check the number of sub-elements in this composite, and also be sure that the data itself does not contain the sub-element separator. Also check to be sure that you are checking against the appropriate standard.

10624 Suspicious data found in *element*, may be undisplayable or cause problems in transmission

EDI data element values should generally be printable data, except the ASC X12 ISA16 (the sub-element delimiter) or when the type is Binary.

10625 Warning - *element and location* doesn't conform to Level <*A or B*> character set

Example: *Warning - NAD02-C08201 (D.E. 3039) at col. 8 doesn't conform to Level A character set.*

The 2nd item in the NAD (NAD**02**) is composite C082. The first item in this composite (C082**01**) is data element 3039, which starts in column 8. It contains data that doesn't conform to UNOA. It may be lower case, for example.

10626 Syntax Error for *xxx* (*element number*) at col. *nn* "*value: reason*"

Example: *"Syntax Error for BGN04 (D.E. 337) at col. 29 "ABCDEFGHIJK": Time required."* This diagnostic warns that the contents of the 4th (04) element in this BGN (data element 337), contains the data "ABCDEFGHIJK." It should contain time data.

Explanation: The value at element *xxx* in this segment does not conform to the length or the type required for this element when used in this location. The erroneous value, which is shown in quotation marks at the end of the message, begins in column *nn*.

10627 Trailing Blanks in *element* and *location*

Example*: Trailing Blanks in PKG04 (D.E. 754) at col. 13*. The PKG segment's 4th element (PKG**04**), data element 754, ends with blanks.

10628 Application Value "*xxx*" not found in value list "*listname*" which is used for ...

Example: *"Application Value "MAGGIE MCGILL" not found in value list "cust1" which is used for N102 (D.E. 93) at col. 7 because Variable1 equals ABC"*

This message appeared because the 2nd element in the N1 at this location (N1**02**, data element 93) has a business rule that uses application values list cust1 when Variable equals ABC. In this transaction Variable is ABC, but that list does not include the value "MAGGIE MCGILL".

10629 Code Value *xxx* not <*used/found*> in the local code list which is used for *...*

The code value in this element violates the business rule given at the end of the error message. The business rule was attached to the guideline in Standards Editor.

10630 Value "*xxx*" is an invalid "*appl. value list desc. (listname)*" which is used for *location*.

Explanation: An application values list called ***listname*** has been attached to this element at this location, either directly or through a business rule. The value *xxx* does not appear on that list (application values are case sensitive). Application values are attached in Standards Editor and can be viewed in both Standards Reference and Editor. Please see "Application Value Lists" in the index of the SE manual. Checking application values is optional. Look under **Options | Analyzer Profile | Check App. Data Values** in the Analyzer menus.

Example: *Value "USA" is an invalid "3-digit Country Code" (COUNTRIES) for CUR02 (D.E. 100) at col. 8*

This message appeared because the 2nd element in the CUR02 at this location (data element 100) has an application values list attached, but that list does not include the value "USA". The value list may or may not be associated with a business rule at this location.

10631 Blank Value supplied for *element* and *location*

Example: *Blank Value supplied for CTA02-C05601 (D.E. 3413) at col. 8*. The 2nd item in the CTA (CTA02) is composite C056. The first item in this composite (C0506**01**) is data element 3413, which starts in column 8. This item contains only blanks.

Explanation: The value in *element* at *location* is completely blank, which is discouraged in EDI data unless needed to fill an element out to its minimum length (as in the case of the ASC X12 ISA). Completely blank element values may indicate a problem in the data. Why weren't they just omitted, if optional? And if mandatory, where is the data? Analyzer continues checking the element's syntax.

10806 Dependent *object* at *location* wasn't expected because ***business*** *rule*

This element, segment, loop, or group should not have been included because a business rule said it is not used.

Example: *Dependent Loop N1 (0100) "Name" at 1-140 wasn't expected because PaymentMethod'NE'BP*

Explanation: This N1 loop should not be included because the element identified by user-defined variable PaymentMethod did not contain a value of BP.

10807 Dependent *object* at *location* not found, but was expected because *business rule*

This element, segment, loop, or group should have been included because a business rule required it.

Example: *Dependent Loop N1 (0100) "Name" at 1-140 not found, but was expected because PaymentMethod'EQ'BP*

Explanation: This N1 loop should be present because the element identified by user-defined variable PaymentMethod contains a value of BP.

10808 <*group, loop, or segment*> xxx at *location* exceeded max use count of *n*

Example: *"Group RFF at 0-060 exceeded max use count of 1"*

This group appears too many times at this location, according to the guideline being used for comparison.

10808 <*segment, loop, or group*> exceeded max use count of *n*

Example: *"Segment PKG at 1-420 exceeded max use count of 2"*

This segment appears too many times at this location, according to the guideline being used for comparison.

10809 Incomplete <*Loop or Group*> *xxx* at location

Example: *"Incomplete 101 Loop PRV (2000) at 2-005"*

Analyzer has detected the end of the transaction set, but loop or group *xxx* is incomplete.

10810 Missing Mandatory <*Loop, Group, Element, or Segment*>

Analyzer has reached the transaction set or message trailer, or the end of a loop or group, without finding this item, which is mandatory, must be used, or recommended according to the standard or guideline. The item might have usage that depends on a business rule, syntax rule, or dependency note.

10811 Missing *xxx*, though marked *usage*

Example: *Missing Segment CTP at 2-040, though marked "Must be Used".* The CTP segment at Table 2, position 040, was marked as "must be used" in the guideline, but it was not included in the data.

Explanation: According to the guideline, segment *xxx* must be used or is recommended at this position in the transaction set or message, but the segment is not present in the data. For an explanation of the differences between mandatory and must be used, please see "Overview of Requirement Designators and User Attributes" in the Standards Editor manual.

10815 *xxx* at *location* present, but was marked "*usage*"

Example: *"Segment SN1 at 2-102 present, but was marked "Not Used".*

This segment should not be included in the data, according to the guideline being used for comparison.

10900 Functional Group Count *n* incorrect - should be *x*

Analyzer counted *x* functional groups in this interchange, but the IEA01 says the count is *n*. Check the GS and GE pairs between this IEA and its matching ISA.

10902 Application Segment total *n* Incorrect - should be *x*

X12: The SE01 (Number of Included Segments) has a value of *n* but Analyzer counts *x* segments in this transaction set, including the ST and SE.

EDIFACT: The first element in the UNT (the Number of Segments in a Message) has a value of *n* but Analyzer counts *x* segments in this message.

10903 Incorrect <*Transaction Set or Message*> Count *n* - should be *x*

Example: *Incorrect Transaction Set Count 21 - should be 1. End of Interchange, Ctl. No. 000000010, contains 1 Functional Group(s).*

X12: Analyzer counted *x* transaction sets in this functional group, but the GE's Number of Transaction Sets gives the count as *n*. Check the ST and SE pairs between this GE and its matching GS.

EDIFACT: Analyzer counted *x* messages in this functional group, but the UNE's Number of Messages gives a different count. Check the UNH and UNT pairs between this UNE and its matching UNB.

10904 Interchange Control Count *n* incorrect - should be *y* (no. of *xxx*)

The first element in the UNZ, the Interchange Control Count (data element 36), has a value of *n*. However, Analyzer counted *y* functional groups in this interchange. If no functional groups were included, Analyzer counted *y* messages.

10905 Message Count *x* incorrect - should be *y*

The value in the first element in the UNE (data element 60, Number of Messages) is *x*, but Analyzer counted *y* messages. Check all UNH and UNT pairs between this UNE and the matching UNG.

10907 Message Reference *x* incorrect - should be *xx*

The message trailer segment count is incorrect. Analyzer counts segments for the UNT01 and the SE01, and offers the correct number of segments.

10908 *and* 10917 Message segment total *nn* Incorrect - should be *xx*

The UNT01 indicates that the message should contain *nn* segments but Analyzer counted *xx* segments. Check for missing or extraneous segments, especially a UNH or UNT.

10909 Interchange Ctl No. *nnn* in EG doesn't match BG's *xxx*

Explanation UCS (TDCC/EDIA): Check the values in the BG07 and the EG01.

10910 Functional Group Reference *xx* in UNE doesn't match UNG

The Functional Group Reference (data element 0062) in the UNE's second element does not match that in the UNG's first element. Check for a missing UNG or UNE, and check the values in data element 0062.

10911 Functional Group Ctl No. *n* in GE doesn't match GS

If the Group Control Numbers in GS06 (data element 28) and GE02 don't match, something is wrong. Either some segments are missing or erroneously included, or the control numbers are in error.

10912 Interchange Ctl No. *nnn* in IEA doesn't match ISA's *xxx*

Example: *Interchange Ctl No. 000000013 in IEA doesn't match ISA's 000000010.* This IEA has interchange control number 13, but the matching ISA has interchange control number 10.

Explanation: Check the values in the ISA13 and IEA02.

10913 Transaction Set Ctl No. *n* in SE doesn't match ST

If the Transaction Set Control Number (data element 329) in ST02 does not match that in SE02, something is wrong. Is a ST or SE segment missing? Are one or both of the values wrong?

10914 Interchange Ctl Reference *x* in UNZ doesn't match UNB *y*

The interchange control reference (data element 0020) in the 5th element in the UNB doesn't match the one in the 2nd element in the UNZ. Check the numbers and pairing of UNZ and UNB. Also check delimiters in these segments.

10915 Message Reference *n* in UNT doesn't match UNH

If the Message Reference (data element 62) in the first element of the UNH does not match the one in the second element in the UNT, something is wrong. Is a UNH or UNT missing? Are the values wrong?

10916 Sub-Element separator same as Data Element separator

The ISA's last element is the sub-element separator. Analyzer can determine the element separator from the ISA, since each of its elements is fixed length. These two types of separator should not be the same. Otherwise, it won't be possible to properly analyze sub-elements in a composite. If it's true that sub-element separators and element separators are the same in the data, then Analyzer will interpret sub-elements as elements and you will get diagnostics when you reach any segment containing a composite.

10917 <*Transaction set or message*> segment total *nn* Incorrect - should be *xx*

Example: "*Transaction Set segment total 99 Incorrect - should be 100.*"

The SE01 or UNT01 indicates that the transaction set or message should contain *nn* segments but Analyzer counted *xx* segments. Check for missing or extraneous segments, especially an SE or ST for X12, or a UNH or UNT for EDIFACT.

11000 Inferring Message ID *xxx* from Functional Group ID

Analyzer could not find the message identifier in the UNH. Based on the contents of the Functional Group identification in the UNG, it is using message *xxx* for the comparison. Check the contents of the Message Identifier in the UNH.

11001 Loaded <*Transaction Set or Message*> *xxx* **from Standard** *yyy* **(***zzz***)**

Example: "*Loaded Message BANSTA from Standard UN-921 (92.1.)"*

Explanation: This message confirms that Analyzer found the transaction set or message in the standard that it is using for comparison. If you do not get this message, be sure that you are analyzing against the correct standard. If Analyzer did not ask you to choose a standard, choose **Options | Analyzer** and turn on **Ask for Standard Name**.

11002 Missing Message ID

The UNH had no data in the first element, so Analyzer cannot identify the message. If the data contains a UNG, then Analyzer will look there for the message ID. Otherwise, Analyzer cannot compare the data to a message, and will have to use dictionary segments for compliance checking.

11002 Missing Transaction Set ID

The ST had no data in the first element, so Analyzer cannot identify the transaction set. Analyzer cannot compare the data to a transaction set, and will have to use dictionary segments for compliance checking.

11003 *and* **11004 Can't find** <*Transaction Set or Message*> *xxx* in **Standard** *yyy* (*zzz*). **Only limited checking …**

Example: "Can't find Transaction Set 837 in Standard SMALL (003040UCS)." This message occurred when the user analyzed the data against guideline "SMALL." The data conformed to guideline "LARGE."

Analyzer does not recognize *xxx* (the contents of ST01 or the Message Identifier in the UNH) as a transaction set or message in guideline *yyy*, which is based on published standard *zzz*. Checking continues, but only from the guideline's dictionary. Did Analyzer use the correct standard or guideline for the analysis? You can have Analyzer prompt for a standard for each file if you place an X next to **Options | Analyzer | Ask for Standard Name**.

11100 Standard *name* cannot be loaded

The standard could not be loaded. Perhaps it is damaged. Contact TIBCO Foresight technical support.

11101 Can't open Control Standard *xxx*

Analyzer cannot open the standard containing the interchange enveloping segments. It will look for these segments in the following locations, in order: (1) Are the interchange and functional group envelopes within the standard itself? If so, they will be used for analyzing the control segments. (2) Does the standard's .STD file have a .CTL record that gives the name of a standard containing the interchange enveloping segments? If so, Analyzer tries to find and open it so that it can interpret the interchange and functional group enveloping segments. (3) If they are still not found, Analyzer will use X12ICS or UN1ICS for interchange and functional group control segments. If this fails, you receive the message above. It's time to contact TIBCO Foresight technical support.

11107 No EDI Standard specified in envelope header

Explanation: Analyzer is trying to determine which published standard to load. It looks in the functional group header for the formal name (example: 003030VICS or A2/7). For EDIFACT, this would be in the message version composite element (S008) of the UNG. For X12, this would be in the GS08 (element 480, the Version / Release / Industry Identifier Code). Check this data and the delimiters in the segment. If data appears to be present, the standard that contains the enveloping might be corrupted. By default, EDISIM uses files \Edisim50 \ Static \ Un1ics.std or X12ics.std for enveloping standards (please see "What to Do About Enveloping" in the Standards Editor manual or "Enveloping" in this document). Be sure this standard is still a plain text file, as it was when it was installed (this diagnostic message has been seen when UN1ICS.STD was accidentally turned into a word processor document).

After displaying this diagnostic, Analyzer will ask you to choose a standard to use for analysis of the functional group header and trailer and everything between. It will not be able to suggest the correct one, but will just select the first standard in its list.

11200 Empty Functional Group contained no <*Messages or Transaction Sets*>

This segment ends a functional group that appears to be empty. It may actually be empty, or the transaction set header (ST) or message header (UNH) may be missing or wrong. Check the pairing of the headers and trailers for the functional group and transaction set or message.

11201 Empty Interchange

This interchange control trailer appears to be terminating an empty interchange. Check the pairing of interchange segments (ISA and IEA or UNB and UNZ).

11202 Empty <*Transaction Set or Message*>, Ctl. No. *xxx*, contained no application Segments

Example: "*Empty Message, Ctl. No. BANSTA4, contained no application Segments."* Analyzer displayed this diagnostic when it found no segments between the UNH and the UNT.

Application segments make up the transaction sets or messages. The data contained transaction set or message header and trailer, but no segments between them.

11203 End of Functional Group, Ctl. No. *xxx*, contains *nnn* <*Transaction Set(s) or Message(s)*>

Examples:

"End of Functional Group, Ctl. No. 7, contains 1 Message(s)." In this informative diagnostic on an EDIFACT UNE, **7** is the Functional Group Reference Number from the UNG. It should match the one in the UNE. **1** is the number of messages that Analyzer counted in the group. It should match the UNE's Number of Messages.

"End of Functional Group, Ctl. No. 10, contains 1 Transaction Set(s)" This diagnostic is on an X12 GE. **10** is the Group Control Number from the GS. **1** is the number of transaction sets that Analyzer counted in the group. It should match the GE's Number of Transaction Sets Included.

Explanation: This informative diagnostics does not imply an error. *xxx* is the GS's Group Control Number (X12) or the UNG's Functional Group Reference Number (EDIFACT). It should match the value in the corresponding GE or UNE. It need not be numeric for EDIFACT. *nnn* is the number of transaction sets or messages in the group, as counted by Analyzer. This should match the GE's Number of Transaction Sets Included or the UNE's Number of Messages.

11204 End of <*Transaction Set or Message*>, Ctl. No. *xxx*, contains *nnn* Segments

Examples:

"End of Transaction Set, Ctl. No. 0015, contains 24 Segments."

The transaction set terminated by this SE contains 24 segments, counting the ST and SE. This should match the count in the SE. 0015 is the control number in the ST.

"End of Message, Ctl. No. BANSTA4, contains 19 Segments." The message terminated by this UNT contained 19 segments. This should match the count in the UNT. BANSTA4 is the control characters UNH and the first in the UNT.

Explanation: This is for information only, and does not imply an error. *xxx* is the Transaction Set Control Number or Message Reference Number. For EDIFACT, it need not be numeric. Analyzer read *xxx* from the SE or UNH, and it should match the one in the ST or UNT. *nnn* is the number of segments counted by Analyzer in the transaction set or message, including the ST-SE or UNH-UNT. *nnn* should match the SE's Number of Included Segments or the UNT's Number of Segments.

11205 End of Interchange, Ctl. No. *xxx*, contains *nnn* <*Functional Groups or Messages*>

This informative diagnostic does not mean that an error has occurred. *xxx* is the ISA's Interchange Control Number (X12) or the UNB's Interchange Control Reference (EDIFACT). *nnn* is the number of functional groups or messages counted by Analyzer in the interchange.

Examples:

"*End of Interchange, Ctl. No. 00000001, contains 1 Functional Group(s)."*

This diagnostic on an X12 IEA means that Analyzer has just checked the interchange with Interchange Control Number 1 (it got the number from the ISA). The IEA should also show Interchange Control Number 1. Analyzer counted 1 functional group. This should match the Number of Included Functional Groups in the IEA.

"End of Interchange, Ctl. No. 7, contains 1 Functional Group(s)."

This diagnostic on an EDIFACT UNZ means that Analyzer has just checked the interchange with Interchange Control Reference 7 (as shown in the UNB). The UNZ should also show Interchange Control Reference 7. Analyzer counted 1 message in the group. This should match the number in the Interchange Control Count in the UNZ.

"End of Interchange, Ctl. No. 7, contains 2 Message(s)."

This diagnostic on an EDIFACT UNZ shows 2 messages in the interchange. This happens when there are no UNG and UNE segments (EDIFACT does not require functional

groups, but X12 does). In this case, the UNZ Interchange Control Count should show the number **2 –** the number of messages rather than number of functional groups.

11206 Functional Group Ctl. No. *nnn* not properly terminated

Example: *"Functional Group Ctl. No. 10 not properly terminated."* This diagnostic was received on the IEA when the GE is missing.

This segment (IEA or UNZ) ends the interchange, but the GE or UNE is missing. EDIFACT messages do not need to have functional groups, but if the functional group header (UNG) is included, then the matching UNE must be included.

11207 Interchange Ctl. No. *nnn* not properly terminated

A new interchange header has been detected, but the trailer (IEA or UNZ) from the previous interchange has not been found. Analysis of the new interchange continues.

11208 Have not seen <*Transaction Set or Message*> Trailer

This GE or UNE ends a functional group, but the transaction set or message trailer (SE or UNT) is missing.

11209 Previous Functional Group, Ctl. No. *n* not properly terminated

This segment (GS or UNG) starts a new functional group, but the functional group trailer (GE or UNE) is missing from the previous group.

11300 Abandoning segment alignment with <*Message or Transaction Set*>

Analyzer has lost track of where segments should go. If this is an X12 document, check all LS-LE pairs, and, in particular, be sure all mandatory segments are within the loops that the LS-LE pairs enclose.

11301 Alternate Trigger at *yyy* ignored - use alternates of the <*segment, loop, or group*> instead

Analyzer has found a second instance of the trigger segment in the group. For example, the first (trigger) segment in the loop or group is RFF, but another RFF is in the same loop or group. Instead, Analyzer suggests creating a second RFF loop or group.

11302 Ambiguous <*loop, or group, or segment*> placement - using definition at *xxx*

There are consecutive loops, groups, or segments with the same loop tag or ID defined in your guideline. This would not happen in a base ASC X12 or EDIFACT standard. Analyzer is trying to determine which alternative definition to use. To do so, Analyzer uses the data in the first coded data element in the loop, group, or segment. It compares this code to the code values in the standard or guideline. If an exact match can't be found, Analyzer looks at other codes in the segment to differentiate between alternative definitions. If there is still no basis for differentiation, it generally uses the first definition.

11303 Assuming LS matches mandatory *segment*, though ID is not "*nnn*"

The current loop did not contain the loop ID (LS01) matching the LS's ID in the standard or guideline. Analyzer found the matching loop elsewhere and is assuming the LS refers to it. Example: LS*0100 is found in the data instead of LS*N1 in 875 of UCS 3030; since the LS*N1 is mandatory, it's assumed that the 0100 is incorrect rather than looking for LS*0100 elsewhere in the standard or guideline.

11304 Loop not terminated by LE at *location*.

The LS-LE are not paired correctly in the ***guideline*** (the problem is not in the data). Contact TIBCO Foresight technical support.

11305 Can't resynchronize - Compliance Check for *xxx* Terminated

Segments are out of order to the extent that Analyzer cannot effectively analyze the document. Check the LS and LE segments carefully and then check the segments within the loop that they enclose.

11319 Empty Bounded *xxx* at *yyy*

Example: *"Empty Bounded Loop NM1 (2310) at 2-250."*

Analyzer found an LS segment immediately followed by a LE at location *yyy*. The value in the LS-LE pair indicated that a loop *xxx* should be enclosed within the LS-LE. In the **example**, the LS and LE at table 2 position 250 included the value 2310, which means that a NM1 loop should be enclosed. You should not use the enveloping LS and LE segments if there are no repetitions of the enclosed loop.

11320 Incorrect loop ID at *position* - should be *xxx* instead of *yyy*

Example: *"Incorrect loop ID at 2-280 - should be 2310 instead of 2330"*

The value in LE01 does not match the type of loop it is terminating. Check the standard in Standards Reference and see the explanation for the diagnostic "**Missing Loop Trigger Segment for Loop... ."**

11321 LE wasn't used to terminate *loop and location*

Example: *"LE wasn't used to terminate Loop NM1 (2310) at 2-250"*

Explanation: This appears to be the end of the bounded (LS-LE) loop but the data does not include an LE.

11322 LS doesn't immediately precede a loop definition - error in <*segment and location*>

Example: "*LS doesn't immediately precede a loop definition - error in Segment REF at 1-015.*"

An extraneous LS or a missing loop caused this message. It is likely to be followed with one or more messages about which loop it expected to see based on the data in the LS. It will continue through the data, searching for an LE to terminate the loop.

11323 Segment *xxx* can't be aligned within <*loop, group, transaction set, or message*>

This segment, loop, or group appears to be out of place. If this is an X12 transaction, and if *xxx* is part of a loop bounded with LS and LE, be sure the value in the LS and LE corresponds to the first segment in the loop (the loop trigger) in transaction set or message *yyy*.

11324 Misplaced *xxx* within *yyy*

Example: *"Misplaced Loop NM1 (2210) at 2-095 within Loop PAT (2200) at 2-090."*

Segment, group, or loop *xxx* appears to be out of order in loop or group *yyy*.

11325 Missing Loop Trigger Segment for *xxx* (*yyy*) at *location*

Example: "*Missing Loop Trigger Segment for Loop NM1 (2310) at 2-250*." The LS and LE contain the value "2310," indicating that a NM1 loop follow. However, the next segment, at Table 2, position 250, was not a NM1.

This LS and LE give a value that corresponds to a *xxx* loop at the location shown, but the first segment in the loop is not *xxx*. Please check this transaction set in Standards Reference or Standards Editor to see the correct value for the LS and LE at this location. These segments are from X12-3040 transaction set 837:

Segment	Req max use	Description	Notes
240 QTY	O 10	Quantity	
241 HCP	O 1	Health Care Pricing	
245 LS	O 1	Loop Header	
2310 NM1	O 9	**Individual or Org Name**	*A bounded loop with ID 2310*
280 LE	O 1	Loop Trailer	
2400 LX	O 10000	**Assigned Number**	A *loop that is not bounded, with ID 2400*
455 LS	O 1	Loop Header	
2500 SBR	O 10	**Subscriber Information**	A *bounded loop with ID 2500*
520 LE	O 1	Loop Trailer	

The LS at 245 and the LE at 280 should contain the value "2310," the Loop ID of the NM1 loop that they enclose. The LS at 455 and the LE at 520 should contain the value "2500," the Loop ID of the SBR loop that they enclose.

11326 Floating NTEs may be defined only at Set Level

A floating NTE segment must be defined in an ASC X12 transaction set near the beginning, after the ST and the beginning segment (such as the BPS). There may be only one floating segment defined in a transaction set, and it should be NTE. It should not be within a loop. ASC X12 discourages its further use. Therefore, check to be sure that a floating NTE, if used, is in Table 1. If this is not the problem, go into Standards Editor and check for other segments that might have accidentally been marked as floating in the guideline.

11329 HL is incorrectly defined as a standalone segment - HLs must only be used as loop triggers

This HL segment doesn't appear to be the first segment in a loop – the only way that HL segments may be used.

11400 Assuming Parent ID (HL02) should be *xxx*

The parent ID in the HL02 does not make sense based on the nesting of the HLs in this document. Analyzer is guessing that it should be *xxx*.

11401 HL segment has different Level Code "*xxx*" from that of sibling ID *yyy* – should be *zzz*

HL segments with different Level Codes both claim to have the same parent.

11402 HL segment *n* marked as having children but in fact has none

Example: "*HL segment "2" marked as having children but in fact has none."* This HL's 4th element (HL04) contains a **1**, indicating that it has children. However, no other HL loop claims this loop as its parent (with a **2** in its 2nd element , which would have indicated that it is the child of HL 2).

The "parent" HL might look like this:

"I am HL loop 2"

↓

HL*2**0*1!

↑

"I have at least one child"

There should be another HL that says that HL2 is its parent:

HL*3*2*O*1!

↑

"My parent is HL loop 2"

However, Analyzer did not find such a child HL.

11403 HL loop *n* has child(ren) but is marked as having none in HL04

Example: *HL loop 2 has child(ren) but is marked as having none in HL04*. The value in HL04 indicates that this loop has no children, but another HL loop is indicating that this loop is its parent.

Explanation: The current loop has *n* in its HL02, indicating that it is the child of HL loop *n*. However, HL loop 2 has the value of 0 in HL04, indicating that it has no children. The following data would generate this type of diagnostic:

HL*2*1*O*0! HL01 contains 2, indicating that this is HL loop 2. HL04 has 0, indicating that this loop has no children.

HL*3*2*T*0! HL02 contains 2, indicating its parent is HL loop 2.

11404 HL loop *n* not subordinate to parent loop *x*

Example: *"HL loop "2" not subordinate to parent loop "9".”* The second element in this HL (HL02) claims that its parent is HL loop 9, a HL with hierarchical ID of 9 in its first element. This is not true, possibly because there is no HL with a 9 in its first element.

"I am HL loop 2"

↓

HL*2*9*O*1!

↑

"My parent is HL loop 9"

11609 Empty Segment - no data elements had values

Check the standard for mandatory elements and composites in this segment. Why include the segment at all if it has no data?

Diagnostic Explanations in Alphabetic Order

<group, loop, or segment> **xxx at** *location* **exceeded max use count of** *n*

Example: *"Group RFF at 0-060 exceeded max use count of 1"*

This group appears too many times at this location, according to the guideline being used for comparison.

<segment, loop, or group> **exceeded max use count of** *n*

Example: *"Segment PKG at 1-420 exceeded max use count of 2"*

This segment appears too many times at this location, according to the guideline being used for comparison.

<Transaction set or message> **segment total** *nn* **Incorrect - should be** *xx*

Example: "*Transaction Set segment total 99 Incorrect - should be 100."* The SE01 or UNT01 indicates that the transaction set or message should contain *nn* segments but Analyzer counted *xx* segments. Check for missing or extraneous segments, especially an SE or ST for X12, or a UNH or UNT for EDIFACT.

Abandoning segment alignment with *<Message or Transaction Set>*

Analyzer has lost track of where segments should go. If this is an X12 document, check all LS-LE pairs, and, in particular, be sure all mandatory segments are within the loops that the LS-LE pairs enclose.

Alternate Trigger at *yyy* ignored - use alternates of the <*segment, loop, or group*> instead

Analyzer has found a second instance of the trigger segment in the group. For example, the first (trigger) segment in the loop or group is RFF, but another RFF is in the same loop or group. Instead, Analyzer suggests creating a second RFF loop or group.

Ambiguous <*loop, or group, or segment*> placement - using definition at *xxx*

There are consecutive loops, groups, or segments with the same loop tag or ID defined in your guideline. This would not happen in a base ASC X12 or EDIFACT standard. Analyzer is trying to determine which alternative definition to use. To do so, Analyzer uses the data in the first coded data element in the loop, group, or segment. It compares this code to the code values in the standard or guideline. If an exact match can't be found, Analyzer looks at other codes in the segment to differentiate between alternative definitions. If there is still no basis for differentiation, it generally uses the first definition.

Application Segment total *n* Incorrect - should be *x*

X12: The SE01 (Number of Included Segments) has a value of *n* but Analyzer counts *x* segments in this transaction set, including the ST and SE.

EDIFACT: The first element in the UNT (the Number of Segments in a Message) has a value of *n* but Analyzer counts *x* segments in this message.

Application Value "*xxx*" not found in value list "*listname*" for <*element and location*>

Explanation: An application values list called *listname* has been attached to this element at this location. The value *xxx* does not appear on that list (application values are case sensitive so capitalization makes a difference).

Application values are attached in Standards Editor. Please see "Application Value Lists" in the SE index. Checking application values is optional. Look under **Options | Analyzer Profile | Check App. Data Values** in the Analyzer menus.

Example: *"Application Value "MAGGIE MCGILL" not found in value list "cust1" for N102 (D.E. 93) at col. 7."* This message appeared because the 2nd element in the N1 at this location (N1**02**, data element 93) has an application values list attached, but that list does not include the value "MAGGIE MCGILL".

Example involving Business Rule: "*Application Value "ARTHUR JONES" not found in value list "HOSPITAL*" which is used for N102 (D.E. 93) at col. 7 because DoctypeBEG02'EQ'BL" This message appeared because the N102, element 93, contains ARTHUR JONES. This value is not in the Application Value List HOSPITAL, which is attached when the element pointed to by variable DoctypeBEGO2contains a BL. A business rule attached by a Standards Editor user is enforcing this.

Application Value "*xxx*" not found in value list "*listname*" which is used for …

Example: *"Application Value "MAGGIE MCGILL" not found in value list "cust1" which is used for N102 (D.E. 93) at col. 7 because Variable1 equals ABC"*

This message appeared because the 2nd element in the N1 at this location (N1**02**, data element 93) has a business rule that uses application values list cust1 when Variable equals ABC. In this transaction Variable is ABC, but that list does not include the value "MAGGIE MCGILL".

Assuming LS matches mandatory *segment*, though ID is not "*nnn*"

The current loop did not contain the loop ID (LS01) matching the LS's ID in the standard or guideline. Analyzer found the matching loop elsewhere and is assuming the LS refers to it. Example: LS*0100 is found in the data instead of LS*N1 in 875 of UCS 3030; since the LS*N1 is mandatory, it's assumed that the 0100 is incorrect rather than looking for LS*0100 elsewhere in the standard or guideline.

Assuming Parent ID (HL02) should be *xxx*

The parent ID in the HL02 does not make sense based on the nesting of the HLs in this document. Analyzer is guessing that it should be *xxx*.

Assuming Sub-Element Delimiter is *x*, rather than *y*

Examples:

(EDIFACT): "*Assuming Sub-Element Delimiter is '@', rather than ':'."* The UNA gave the sub-element delimiter as ":". However, the UNB used @ to separate sub-elements. Analyzer is assuming that the rest of the file uses @.

(X12): *"Assuming Sub-Element Delimiter is ':', rather than '@'" The* last element in the ISA (just before the segment terminator) contains "@", but the current segment appears to be using the colon for a sub-element delimiter.

Explanation: Even though the ISA, UNA or appropriate defaults may have been used to supply the sub-element delimiter, it's likely that an altogether different character is actually being used to separate sub-elements within a composite data element.

For X12, check the ISA and then the segments that have composites.

For EDIFACT, check the UNA (if present), the UNB, and the segments that have sub-elements.

Binary segment length error, cannot continue

The BIN01 specifies the length in bytes for the binary data in the BIN02. However, end of file was reached before that number of bytes was supplied.

Blank Value supplied for *element* and *location*

Example: *Blank Value supplied for CTA02-C05601 (D.E. 3413) at col. 8*. The 2nd item in the CTA (CTA02) is composite C056. The first item in this composite (C0506**01**) is data element 3413, which starts in column 8. This item contains only blanks.

Explanation: The value in *element* at *location* is completely blank, which is discouraged in EDI data unless needed to fill an element out to its minimum length (as in the case of the ASC X12 ISA). Completely blank element values may indicate a problem in the data. Why weren't they just omitted, if optional? And if mandatory, where is the data? Analyzer continues checking the element's syntax.

Blank Value supplied for *name*, should be omitted

The optional element contains blanks, but it should be omitted entirely.

Can't find <*Transaction Set or Message*> *xxx* in Standard *yyy* (*zzz*). Only limited checking ...

Example: "Can't find Transaction Set 837 in Standard SMALL (003040UCS)." This message occurred when the user analyzed the data against guideline "SMALL." The data conformed to guideline "LARGE."

Analyzer does not recognize *xxx* (the contents of ST01 or the Message Identifier in the UNH) as a transaction set or message in guideline *yyy*, which is based on published standard *zzz*. Checking continues, but only from the guideline's dictionary. Did Analyzer use the correct standard or guideline for the analysis? You can have Analyzer prompt for a standard for each file if you place an X next to **Options | Analyzer | Ask for Standard Name**.

Can't Interpret Control Segment *xxx* - Missing from Standard *yyy*

This serious error occurs when the enveloping standard could be found, but its dictionary does not include the interchange or group enveloping segments, or when the application standard's dictionary does not include the document header (ST or UNH). Check that X12ICS.STD or UN1ICS.STD is in the Static folder and contains these segments in the dictionary, or contact TIBCO Foresight technical support.

Can't interpret segment without proper Envelopes

This segment is not enclosed in recognizable interchange, functional group, or transaction set/message enveloping segments. Check all envelope pairings. For EDIFACT, these must be interchange (UNB and UNZ) and message (UNH and UNT) headers and trailers. If included, UNG and UNE must be paired. For X12, these must be interchange (ISA and IEA), functional group (GS and GE), and transaction set (ST and SE) headers and trailers.

Can't open Control Standard *xxx*

Analyzer cannot open the standard containing the interchange enveloping segments. It will look for these segments in the following locations, in order: (1) Are the interchange and functional group envelopes within the standard itself? If so, they will be used for analyzing the control segments. (2) Does the standard's .STD file have a .CTL record that gives the name of a standard containing the interchange enveloping segments? If so, Analyzer tries to find and open it so that it can interpret the interchange and functional group enveloping segments. (3) If they are still not found, Analyzer will use X12ICS or UN1ICS for interchange and functional group control segments. If this fails, you receive the message above. It's time to contact TIBCO Foresight technical support.

Can't open transmission file *name*

The file may be in use by another application.

Can't resynchronize - Compliance Check for *xxx* Terminated

Segments are out of order to the extent that Analyzer cannot effectively analyze the document. Check the LS and LE segments carefully and then check the segments within the loop that they enclose.

Code Value xxx not <*used or found*> for <*element and location*>

Example: *"Code Value "850" not used for ST01 (D.E. 143) at col. 4."*

Code value 850 is marked as not used in the guideline or industry subset for the ST's 1st element (ST**01**, data element 143), although it may be a valid code for the underlying X12 standard.

If the diagnostic says *xxx* was not **used**, it is in the dictionary for this segment and element, but is marked as not used at the current location. If the diagnostic says *xxx* was not **found**, the code value was not in the dictionary for this segment and element. In the example, "Col. 4" means the code 850 starts in the fourth character position in the segment, after "ST*".

Code Value *xxx* not <*used/found*> in the local code list which is used for *...*

The code value in this element violates the business rule given at the end of the error message. The business rule was attached to the guideline in Standards Editor.

Code Value *xxx* not found for *element* - possibly *xxx* was intended.

Example: *Code Value "3" not found for REF02 (D.E. 127) at col. 8 - possibly "03" was intended.*

Numeric code value 3 was used in the EDI data, but not found in the dictionary for that element. Analyzer does find 03 in the dictionary, however, and suggests that leading zeros might have been accidentally omitted in the data.

Conditional Element Relation violated: *condition*

Example: *"Conditional Element Relation violated: Only one of MEA08 or MEA03 may be present."*

A conditional relation (syntax rule) has been violated. To see syntax rules, select the segment or composite in Standards Reference and look at the syntax area at the bottom left.

From Standards Editor, you can see a segment's syntax rules by highlighting the segment in the segment dictionary and then clicking the Edit Rules button in the bottom Detail pane. If the syntax rule is on a composite, highlight the composite in the composite dictionary and then click the Edit Rules button.

Could not interpret functional group header segment

Analyzer could not understand the GS or UNG in the data. Analyzer looks in one of two places for a GS or UNG definition: in the dictionary of the standard or guideline that contains the transaction set/message or, if it isn't there, in the dictionary of the interchange standard or guideline.

The interchange standard, by default, is either UN1ICS or X12ICS. It may be another standard or guideline if the user chose "Other Standard" when saving as… or changing the properties to the transaction(s)/message(s) in Standards Editor.

Could not interpret UNA - ignored

Analyzer needs to interpret the UN/EDIFACT Service String Advice (UNA) at least as far as the end of the release indicator field (see below). The last field of the UNA is reserved. When included, the UNA must appear immediately before the Interchange Header (UNB) segment and begin with the upper case characters UNA followed by six characters selected by the sender to indicate, in order:

Type Length	Req	Description	Notes
an 1	M	COMPONENT DATA ELEMENT SEPARATOR	
an 1	M	DATA ELEMENT SEPARATOR	
an 1	M	DECIMAL NOTATION	Comma or full stop
an 1	M	RELEASE INDICATOR	If not used, insert space character
an 1	M	Reserved for future	Insert space character
an 1	M	SEGMENT TERMINATOR	

Example: `UNA:+.? '`

Could not load standard *xxx* to interpret <*ISA or UNB*> interchange header segment

Examples:

X12: *" Could not load standard X12ICS to interpret ISA interchange header segment."* This message appeared on the ISA when the default enveloping standard X12ICS.STD was not available in the \Edisim<n.n>\Static folder.

EDIFACT: *"Could not load standard UN1ICS to interpret UNB interchange header segment."* This message appeared on the UNB when the default enveloping standard UN1ICS.STD was not available in the \Edisim50\Static folder.

Explanation: Analyzer cannot load the standard that contains the Interchange and Functional Group envelopes. When the guideline (or "local standard") was saved in Standards Editor, the user chooses where Analyzer should look for its enveloping (via Standard Editor's **File | Save As | Properties**). If the user chose "Default," then Analyzer looks in standard UN1ICS (EDIFACT) or X12ICS (X12) for the enveloping segments. These standards must be in the \Edisim50\Static folder. If **Other Standard** was selected in Standards Editor, the user specifically selected a standard where Analyzer should look for enveloping. This standard must be available as an .STD file in EDISIM's Static or User Files… directories and must be readable (plain text) by Analyzer. If Analyzer could not load the standard, then it could not interpret either the interchange envelope header or the functional group header. For more information, look under "Enveloping" in the index of the Standards Editor or Analyzer manuals.

Couldn't read XrefSeg at *xxx*

xxx is a hex address. An essential resource is unavailable. Is the disk drive full?

CTT01 must contain *nnn* – the count of *xxx* segments

The value in the CTT01 is incorrect. It should be counting the segment mentioned. A business rule in Standards Editor has requested CTT01 checking.

CTT02 must be used here

The guideline specifies that this element must be used, but no data is included.

CTT02 must contain *nnn* – the sum of *xxx* element

The value in the CTT02 must sum the values in element *xxx*. A business rule in Standards Editor has requested CTT02 checking.

CTT02 should not be used here

The guideline specifies that this element should not be used, but data is included.

Default Level A Data Element Separator Used - Assuming UNOA

UNOB syntax level was specified in the UNB, but the default data element separator for the UNOA syntax level was used in column 4 of the UNB Interchange Header, which leads Analyzer to believe that UNOB was specified in error.

Dependent *object* at *location* not found, but was expected because *business rule*

This element, segment, loop, or group should have been included because a business rule required it.

Example: *Dependent Loop N1 (0100) "Name" at 1-140 not found, but was expected because PaymentMethod'EQ'BP*

Explanation: This N1 loop should be present because the element identified by user-defined variable PaymentMethod contains a value of BP.

Dependent *object* at *location* wasn't expected because ***business*** *rule*

This element, segment, loop, or group should not have been included because a business rule said it is not used.

Example: *Dependent Loop N1 (0100) "Name" at 1-140 wasn't expected because PaymentMethod'NE'BP*

Explanation: This N1 loop should not be included because the element identified by user-defined variable PaymentMethod did not contain a value of BP.

Didn't pass any of the following specs: *x, y* Probably the length ...

The check digit parameter specified in Standards Editor requires that this value conform to *x* and *y* formats. The problem can be too many leading zeroes, but is usually the length. See the **check digit** section of the Standards Editor manual for details.

Element or Composite *xxx* not found

All elements and composites must be in the standard's dictionary, if segments refer to them. This would probably indicate an error in a TIBCO Foresight supplied standard, or dictionary elements were accidentally deleted from a user guideline from within Standards Editor. It most likely is not an error in your data, but rather the standard or guideline.

Element *xxx* at col. *nn* is missing, though marked "*usage*"

Example: "*Element BGN05 (D.E. 623) at col. 34 is missing, though marked "Must be Used.""* This diagnostic means the 5th element (05) in the BGN, which is data element 623 (D.E. 623), contains no data although the industry standard or guideline says it must be used.

Explanation: According to the industry standard or guideline, the element must be used at this position in this segment in the transaction set or message, but no data is present for this element. For an explanation of the differences between Mandatory and Must be Used, please see "Overview of Requirement Designators and User Attributes" in the Standards Editor manual.

The "because" reason will appear in the message if the element must be used due to a business rule.

Element *xxx* is present, though marked *usage*

Example: *BGN04 (D.E. 337) at col. 29 present, though marked "Not Used."* The 4th element in the BGN segment, which is data element 337, was marked as not used in the guideline. However, it is included in the data, and starts in column 29.

Explanation: According to the guideline or industry subset (VICS, UCS, etc.), this element is not to be included in the data. This may differ from the requirement in the underlying X12 standard. If you do not wish Analyzer to notify you about this condition, look under **Options | Analyzer**.

The "because" reason will appear in the message if the item was not used due to a business rule.

Empty <*Transaction Set or Message*>, Ctl. No. *xxx*, contained no application Segments

Example: "*Empty Message, Ctl. No. BANSTA4, contained no application Segments."* Analyzer displayed this diagnostic when it found no segments between the UNH and the UNT.

Application segments make up the transaction sets or messages. The data contained transaction set or message header and trailer, but no segments between them.

Empty Bounded *xxx* at *yyy*

Example: *"Empty Bounded Loop NM1 (2310) at 2-250."*

Analyzer found an LS segment immediately followed by a LE at location *yyy*. The value in the LS-LE pair indicated that a loop *xxx* should be enclosed within the LS-LE. In the **example**, the LS and LE at table 2 position 250 included the value 2310, which means that a NM1 loop should be enclosed. You should not use the enveloping LS and LE segments if there are no repetitions of the enclosed loop.

Empty Functional Group contained no <*Messages or Transaction Sets*>

This segment ends a functional group that appears to be empty. It may actually be empty, or the transaction set header (ST) or message header (UNH) may be missing or wrong. Check the pairing of the headers and trailers for the functional group and transaction set or message.

Empty Interchange

This interchange control trailer appears to be terminating an empty interchange. Check the pairing of interchange segments (ISA and IEA or UNB and UNZ).

Empty Segment - no data elements had values

Check the standard for mandatory elements and composites in this segment. Why include the segment at all if it has no data?

Encountered Functional Group Header outside of Interchange

This functional group envelope header (GS or UNG) does not appear to be within an interchange (ISA-IEA or UNB-UNZ). At present, groups can only be included within interchanges. This is because Analyzer uses the interchange envelope to verify delimiters and separators, as explained below. For details on how Analyzer determines where to look for enveloping, please see "enveloping segments" in the index of the Standards Editor or Analyzer manuals.

X12: The functional group header segment GS points to the application standard under which the GS itself should be interpreted. But Analyzer cannot break apart and interpret the GS in order to extract GS08, the Version-Release-Industry Code, without knowing its structure. It needs to have a definition of the GS in the interchange standard (either in the one selected by the user, or in the default X12ICS interchange standard). Once Analyzer finds the GS08, it uses the standard listed there, which describes the transaction sets or messages. Newer application standards have a GS in the segment dictionary, and Analyzer uses it to actually validate the GS. Otherwise, the interchange standard is used to validate the GS.

EDIFACT: Analyzer uses a similar process to the X12 explanation above. If it cannot find the UNG in the dictionary of the standard that contains the transaction set/message, Analyzer uses the interchange standard (UN1ICS or the one chosen during Standards Editor's **Save As | Properties**) to check the syntax of the UNG.

End of <*Transaction Set or Message*>, Ctl. No. *xxx*, contains *nnn* Segments

Examples:

"End of Transaction Set, Ctl. No. 0015, contains 24 Segments."

The transaction set terminated by this SE contains 24 segments, counting the ST and SE. This should match the count in the SE. 0015 is the control number in the ST.

"End of Message, Ctl. No. BANSTA4, contains 19 Segments." The message terminated by this UNT contained 19 segments. This should match the count in the UNT. BANSTA4 is the control characters UNH and the first in the UNT.

Explanation: This is for information only, and does not imply an error. *xxx* is the Transaction Set Control Number or Message Reference Number. For EDIFACT, it need not be numeric. Analyzer read *xxx* from the SE or UNH, and it should match the one in the ST or UNT. *nnn* is the number of segments counted by Analyzer in the transaction set or message, including the ST-SE or UNH-UNT. *nnn* should match the SE's Number of Included Segments or the UNT's Number of Segments.

End of Functional Group, Ctl. No. *xxx*, contains *nnn* <*Transaction Set(s) or Message(s)*>

Examples:

"End of Functional Group, Ctl. No. 7, contains 1 Message(s)." In this informative diagnostic on an EDIFACT UNE, **7** is the Functional Group Reference Number from the UNG. It should match the one in the UNE. **1** is the number of messages that Analyzer counted in the group. It should match the UNE's Number of Messages.

"End of Functional Group, Ctl. No. 10, contains 1 Transaction Set(s)" This diagnostic is on an X12 GE. **10** is the Group Control Number from the GS. **1** is the number of transaction sets that Analyzer counted in the group. It should match the GE's Number of Transaction Sets Included.

Explanation: This informative diagnostics does not imply an error. *xxx* is the GS's Group Control Number (X12) or the UNG's Functional Group Reference Number (EDIFACT). It should match the value in the corresponding GE or UNE. It need not be numeric for EDIFACT. *nnn* is the number of transaction sets or messages in the group, as counted by Analyzer. This should match the GE's Number of Transaction Sets Included or the UNE's Number of Messages.

End of Interchange, Ctl. No. *xxx*, contains *nnn* <*Functional Groups or Messages*>

This informative diagnostic does not mean that an error has occurred. *xxx* is the ISA's Interchange Control Number (X12) or the UNB's Interchange Control Reference (EDIFACT). *nnn* is the number of functional groups or messages counted by Analyzer in the interchange.

Examples:

"*End of Interchange, Ctl. No. 00000001, contains 1 Functional Group(s).*" This diagnostic on an X12 IEA means that Analyzer has just checked the interchange with Interchange Control Number 1 (it got the number from the ISA). The IEA should also show Interchange Control Number 1. Analyzer counted 1 functional group. This should match the Number of Included Functional Groups in the IEA.

"End of Interchange, Ctl. No. 7, contains 1 Functional Group(s)." This diagnostic on an EDIFACT UNZ means that Analyzer has just checked the interchange with Interchange Control Reference 7 (as shown in the UNB). The UNZ should also show Interchange Control Reference 7. Analyzer counted 1 message in the group. This should match the number in the Interchange Control Count in the UNZ.

"End of Interchange, Ctl. No. 7, contains 2 Message(s)." This diagnostic on an EDIFACT UNZ shows 2 messages in the interchange. This happens when there are no UNG and UNE segments (EDIFACT does not require functional groups, but X12 does). In this case, the

UNZ Interchange Control Count should show the number **2 –** the number of messages rather than number of functional groups.

Exceeds maximum length of *n*

The check digit parameter specified in Standards Editor requires that this value be no longer than *n* characters long, including the check digit (last digit) itself.

Excess Data Elements after <*element and column*>

Example*: "Excess Data Elements after NTE02 (D.E. 352) at col. 9"*

By looking at the element and sub-element delimiters, Analyzer concludes that there are too many elements in this segment's data. Check that the data itself does not contain the delimiter character, and check that a segment terminator has not been omitted.

Excess trailing <*element or composite sub-element*> delimiters after *xxx* at col. *nn*

Example:

"Excess trailing delimiters after PO110 (D.E. 235) at col. 43"

"Excess Trailing Composite Sub-Element Delimiter(s) after UNH02-S00904 (D.E. 0051) at col. 26"

Explanation: Analyzer is going to ignore the rest of the segment or composite. Look at the location listed for extra delimiters or separators.

Expected sequence number *n* (repetitions of *yyy*) at *location*

Example: *"Expected sequence number 1 (repetitions of DNA) at DNA01 (D.E. SEQA) at col. 5"*

TRADACOMS only. Repeating segments, or segments within loops, will have sequence numbers at the front of the segment explicitly giving the repetition occurrence. For example, a single repeating DNA segment in the Header section of Product File will have a single SEQA in position 1, which will have the repeat count of the segment. Likewise, the loop trigger of the NOI loop in the detail section will have a single SEQA giving the repeat count of the NOI loop. Nested loops and segments immediately contained within the NOI loop will have both SEQA and SEQB - SEQA is the same as SEQA in the NOI segment, and SEQB is the occurrence of the loop or segment, respectively. This pattern of sequence

numbers is repeated up to 4 times (SEQA through SEQD), accommodating that much nesting.

Floating NTEs may be defined only at Set Level

A floating NTE segment must be defined in an ASC X12 transaction set near the beginning, after the ST and the beginning segment (such as the BPS). There may be only one floating segment defined in a transaction set, and it should be NTE. It should not be within a loop. ASC X12 discourages its further use. Therefore, check to be sure that a floating NTE, if used, is in Table 1. If this is not the problem, go into Standards Editor and check for other segments that might have accidentally been marked as floating in the guideline.

Functional Group Count *n* incorrect - should be *x*

Analyzer counted *x* functional groups in this interchange, but the IEA01 says the count is *n*. Check the GS and GE pairs between this IEA and its matching ISA.

Functional Group Ctl No. *n* in GE doesn't match GS

If the Group Control Numbers in GS06 (data element 28) and GE02 don't match, something is wrong. Either some segments are missing or erroneously included, or the control numbers are in error.

Functional Group Ctl. No. *nnn* not properly terminated

Example: *"Functional Group Ctl. No. 10 not properly terminated."* This diagnostic was received on the IEA when the GE is missing.

This segment (IEA or UNZ) ends the interchange, but the GE or UNE is missing. EDIFACT messages do not need to have functional groups, but if the functional group header (UNG) is included, then the matching UNE must be included.

Functional Group ID "*xx*" mismatch - <*Transaction set or Message*> is type "*yy*"

Examples*:*

"Functional Group ID "BANSTz" mismatch - Message is type "BANSTA." " The message identifier in the UNG erroneously contained "BANSTz."

"Functional Group ID "PQ" mismatch - Transaction Set is type "PO."" The GS01 erroneously contained "PQ" but the ST01 contained "850."

EDIFACT: The contents of the UNG Message Type Identifier and the UNH Message Type Identifier are different. Example:

```
UNG+BANSTz+APPL SENDER ID:12+.. etc.
UNH+BANSTA1+BANSTA:92:1:UN:+COMM ACCESS REF… etc.
```

If no UNG-UNE segments are used, then different EDIFACT message types may be in a single interchange.

X12: The contents of GS01 (Functional Identifier Code) do not correspond to the ST01 (Transaction Set Identifier Code) as required by ASC X12. An exception is the 980 Functional Group Totals, which assumes the functional group ID of the containing functional group. In this case, TDG's FuncID function will return "" and Analyzer will not display a message about the difference. This message also appears if the GS is missing from both the application standard (the one that contains the transaction set) or in the interchange standard (X12ICS or the one chosen in Standards Editor).

Functional Group Reference *xx* in UNE doesn't match UNG

The Functional Group Reference (data element 0062) in the UNE's second element does not match that in the UNG's first element. Check for a missing UNG or UNE, and check the values in data element 0062.

Functional Trailer Group ignored - No corresponding func grp exists

This GE or UNE does not appear to have a corresponding GS or UNG.

Have not seen <*Transaction Set or Message*> Trailer

This GE or UNE ends a functional group, but the transaction set or message trailer (SE or UNT) is missing.

HL is incorrectly defined as a standalone segment - HLs must only be used as loop triggers

This HL segment doesn't appear to be the first segment in a loop – the only way that HL segments may be used.

HL loop *n* has child(ren) but is marked as having none in HL04

Example: *HL loop 2 has child(ren) but is marked as having none in HL04.* The value in HL04 indicates that this loop has no children, but another HL loop is indicating that this loop is its parent.

Explanation: The current loop has *n* in its HL02, indicating that it is the child of HL loop *n*. However, HL loop 2 has the value of 0 in HL04, indicating that it has no children. The following data would generate this type of diagnostic:

`HL*2*1*O*0!`	*HL01 contains 2, indicating that this is HL loop 2. HL04 has 0, indicating that this loop has no children.*
`HL*3*2*T*0!`	*HL02 contains 2, indicating its parent is HL loop 2.*

HL loop *n* not subordinate to parent loop *x*

Example: *"HL loop "2" not subordinate to parent loop "9".*" The second element in this HL (HL02) claims that its parent is HL loop 9, a HL with hierarchical ID of 9 in its first element. This is not true, possibly because there is no HL with a 9 in its first element.

"I am HL loop 2"

↓

`HL*2*9*O*1!`

↑

"My parent is HL loop 9"

HL segment has different Level Code "*xxx*" from that of sibling ID *yyy* – should be *zzz*

HL segments with different Level Codes both claim to have the same parent.

HL segment *n* marked as having children but in fact has none

Example: "*HL segment "2" marked as having children but in fact has none."* This HL's 4th element (HL04) contains a **1**, indicating that it has children. However, no other HL loop claims this loop as its parent (with a **2** in its 2nd element , which would have indicated that it is the child of HL 2).

The "parent" HL might look like this:

"I am HL loop 2"
↓
HL*2**0*1!
↑
"I have at least one child"

There should be another HL that says that HL2 is its parent:

HL*3*2*O*1!
↑
"My parent is HL loop 2"

However, Analyzer did not find such a child HL.

ICS must be exactly 67 bytes long, excl. segment terminator

The ICS is a fixed length record, which doesn't use the element delimiter to separate elements.

Incomplete <*Loop or Group*> *xxx* at location

Example: *"Incomplete 101 Loop PRV (2000) at 2-005"*

Analyzer has detected the end of the transaction set, but loop or group *xxx* is incomplete.

Incorrect <*Transaction Set or Message*> **Count** *n* **- should be** *x*

Example: *Incorrect Transaction Set Count 21 - should be 1. End of Interchange, Ctl. No. 000000010, contains 1 Functional Group(s).*

X12: Analyzer counted *x* transaction sets in this functional group, but the GE's Number of Transaction Sets gives the count as *n*. Check the ST and SE pairs between this GE and its matching GS.

EDIFACT: Analyzer counted *x* messages in this functional group, but the UNE's Number of Messages gives a different count. Check the UNH and UNT pairs between this UNE and its matching UNB.

Incorrect loop ID at *position* **- should be** *xxx* **instead of** *yyy*

Example: *"Incorrect loop ID at 2-280 - should be 2310 instead of 2330"*

The value in LE01 does not match the type of loop it is terminating. Check the standard in Standards Reference and see the explanation for the diagnostic "**Missing Loop Trigger Segment for Loop... ."**

Inferring Message ID *xxx* **from Functional Group ID**

Analyzer could not find the message identifier in the UNH. Based on the contents of the Functional Group identification in the UNG, it is using message *xxx* for the comparison. Check the contents of the Message Identifier in the UNH.

Interchange Control Count *n* **incorrect - should be** *y* **(no. of** *xxx***)**

The first element in the UNZ, the Interchange Control Count (data element 36), has a value of *n*. However, Analyzer counted *y* functional groups in this interchange. If no functional groups were included, Analyzer counted *y* messages.

Interchange Ctl No. *nnn* **in EG doesn't match BG's** *xxx*

Explanation UCS (TDCC/EDIA): Check the values in the BG07 and the EG01.

Interchange Ctl No. *nnn* in IEA doesn't match ISA's *xxx*

Example: *Interchange Ctl No. 000000013 in IEA doesn't match ISA's 000000010.* This IEA has interchange control number 13, but the matching ISA has interchange control number 10.

Explanation: Check the values in the ISA13 and IEA02.

Interchange Ctl Reference *x* in UNZ doesn't match UNB *y*

The interchange control reference (data element 0020) in the 5th element in the UNB doesn't match the one in the 2nd element in the UNZ. Check the numbers and pairing of UNZ and UNB. Also check delimiters in these segments.

Interchange Ctl. No. *nnn* not properly terminated

A new interchange header has been detected, but the trailer (IEA or UNZ) from the previous interchange has not been found. Analysis of the new interchange continues.

Invalid Check Digit: Non-Numeric Data

The last character in this value must be numeric so that it can be used as a check digit. A business rule in Standards Editor requested check digit validation.

Invalid Check Digit: Should be *nnn*

The last character in this value must be *nnn* to be a valid check digit. A business rule in Standards Editor requested check digit validation.

Invalid data type, this value has been ignored in the calculation of CTT02

Elements containing values to be totaled for the CTT02 hash total must be numeric data types. A business rule in Standards Editor has requested CTT02 checking.

Invalid Date time format

X12: This 1250-1251 element pair has the DateTime business rule validation attached in Standards Editor. The date does not match the format specified in the qualifier.

EDIFACT: Analyzer checks the date/time format for each 2380-2379 subelement pair in a C507 composite. The date value should be an integer if the qualifier's code value is greater than 800. If this location has DateTime business rule validation attached in Standards Editor, then Analyzer will further examine the date to see if it matches its qualifier.

Invalid date time format – *xxx*

The date in this element does not conform to the format required by its qualifier, *xxx*. A business rule in Standards Editor has requested date-time checking.

Invalid Syntax identifier - UNO*x*, where *x* is alpha, expected

To determine the allowable character sets, Analyzer looks at the UNB01-01 (Syntax Level). It sets the syntax level to the last alpha character in the 4-character code … usually A, B, or C. This in effect determines the default separators and delimiters. If a UNA segment precedes this UNB, then the delimiters set there will be used instead of whatever is specified in the UNB.

LE wasn't used to terminate *loop and location*

Example: *"LE wasn't used to terminate Loop NM1 (2310) at 2-250"*

Explanation: This appears to be the end of the bounded (LS-LE) loop but the data does not include an LE.

Leading Blanks in *name*

Leading blanks are significant and allowed for alphanumeric elements, but Analyzer issues warning messages for them in R or N elements.

Loaded <*Transaction Set or Message*> *xxx* **from Standard** *yyy* **(***zzz***)**

Example: "*Loaded Message BANSTA from Standard UN-921 (92.1.)"*

Explanation: This message confirms that Analyzer found the transaction set or message in the standard that it is using for comparison. If you do not get this message, be sure that you are analyzing against the correct standard. If Analyzer did not ask you to choose a standard, choose **Options | Analyzer** and turn on **Ask for Standard Name**.

Loop not terminated by LE at *location.*

The LS-LE are not paired correctly in the ***guideline*** (the problem is not in the data). Contact TIBCO Foresight technical support.

LS doesn't immediately precede a loop definition - error in <*segment and location*>

Example: "*LS doesn't immediately precede a loop definition - error in Segment REF at 1-015.*"

An extraneous LS or a missing loop caused this message. It is likely to be followed with one or more messages about which loop it expected to see based on the data in the LS. It will continue through the data, searching for an LE to terminate the loop.

Message Count *x* **incorrect - should be** *y*

The value in the first element in the UNE (data element 60, Number of Messages) is *x*, but Analyzer counted *y* messages. Check all UNH and UNT pairs between this UNE and the matching UNG.

Message Reference *n* **in UNT doesn't match UNH**

If the Message Reference (data element 62) in the first element of the UNH does not match the one in the second element in the UNT, something is wrong. Is a UNH or UNT missing? Are the values wrong?

Message Reference *x* incorrect - should be *xx*

The message trailer segment count is incorrect. Analyzer counts segments for the UNT01 and the SE01, and offers the correct number of segments.

Message segment total *nn* Incorrect - should be *xx*

The UNT01 indicates that the message should contain *nn* segments but Analyzer counted *xx* segments. Check for missing or extraneous segments, especially a UNH or UNT.

Misplaced *xxx* within *yyy*

Example: *"Misplaced Loop NM1 (2210) at 2-095 within Loop PAT (2200) at 2-090."* Segment, group, or loop *xxx* appears to be out of order in loop or group *yyy*.

Missing Functional Group Header

A GS may be missing or out of order. EDIFACT doesn't require functional group envelopes, since you can deduce the application standard from the UNH segment. X12 cannot tolerate missing functional groups, since the VRI (Version-Release- Industry Code) only exists in the GS segment.

Missing Loop Trigger Segment for *xxx* (*yyy*) at *location*

Example: "*Missing Loop Trigger Segment for Loop NM1 (2310) at 2-250.*" The LS and LE contain the value "2310," indicating that a NM1 loop follow. However, the next segment, at Table 2, position 250, was not a NM1.

This LS and LE give a value that corresponds to a *xxx* loop at the location shown, but the first segment in the loop is not *xxx*. Please check this transaction set in Standards Reference or Standards Editor to see the correct value for the LS and LE at this location. These segments are from X12-3040 transaction set 837:

Segment	Req max use	Description	Notes
240 QTY	O 10	Quantity	
241 HCP	O 1	Health Care Pricing	

245 LS	O 1	Loop Header	
2310 NM1	O 9	**Individual or Org Name**	*A bounded loop with ID 2310*
280 LE	O 1	Loop Trailer	
2400 LX	O 10000	**Assigned Number**	A *loop that is not bounded, with ID 2400*
455 LS	O 1	Loop Header	
2500 SBR	O 10	**Subscriber Information**	A *bounded loop with ID 2500*
520 LE	O 1	Loop Trailer	

The LS at 245 and the LE at 280 should contain the value "2310," the Loop ID of the NM1 loop that they enclose. The LS at 455 and the LE at 520 should contain the value "2500," the Loop ID of the SBR loop that they enclose.

Missing Mandatory <*Loop, Group, Element, or Segment*>

Analyzer has reached the transaction set or message trailer, or the end of a loop or group, without finding this item, which is mandatory, must be used, or recommended according to the standard or guideline. The item might have usage that depends on a business rule, syntax rule, or dependency note.

Missing Mandatory *item* **at** *location*

Examples:

"Missing Mandatory Loop HL at 2-010"

"Missing mandatory LIN02 (D.E. 235) at col. 6"

According to the published standard, a mandatory loop, group, segment, or element should appear at the *location* shown (in the top example, it should be in Table 2, position 10; in the bottom example, it should be the second element in the LIN segment).

Missing Message ID

The UNH had no data in the first element, so Analyzer cannot identify the message. If the data contains a UNG, then Analyzer will look there for the message ID. Otherwise, Analyzer cannot compare the data to a message, and will have to use dictionary segments for compliance checking.

Missing or Too Long Segment Tag

The segment tag (the segment's ID, which is the first item in the segment: ST, UNH, etc.) is incorrect. Check the tag and delimiters.

Missing Transaction Set ID

The ST had no data in the first element, so Analyzer cannot identify the transaction set. Analyzer cannot compare the data to a transaction set, and will have to use dictionary segments for compliance checking.

Missing *xxx*, though marked *usage*

Example: *Missing Segment CTP at 2-040, though marked "Must be Used"*. The CTP segment at Table 2, position 040, was marked as "must be used" in the guideline, but it was not included in the data.

Explanation: According to the guideline, segment *xxx* must be used or is recommended at this position in the transaction set or message, but the segment is not present in the data. For an explanation of the differences between mandatory and must be used, please see "Overview of Requirement Designators and User Attributes" in the Standards Editor manual.

No corresponding Header for *<Transaction Set or Message>* Trailer

This document trailer does not appear to have a corresponding document header. There are either too few document headers or too many document trailers.

No EDI Standard specified in envelope header

Explanation: Analyzer is trying to determine which published standard to load. It looks in the functional group header for the formal name (example: 003030VICS or A2/7). For EDIFACT, this would be in the message version composite element (S008) of the UNG. For X12, this would be in the GS08 (element 480, the Version / Release / Industry Identifier Code). Check this data and the delimiters in the segment. If data appears to be present, the standard that contains the enveloping might be corrupted. By default, EDISIM uses files \Edisim40\Static\Un1ics.std or X12ics.std for enveloping standards (please see "What to Do About Enveloping" in the Standards Editor manual or Enveloping Methods). Be sure this standard is still a plain text file, as it was when it was installed (this diagnostic message has been seen when UN1ICS.STD was accidentally turned into a word processor document).

After displaying this diagnostic, Analyzer will ask you to choose a standard to use for analysis of the functional group header and trailer and everything between. It will not be able to suggest the correct one, but will just select the first standard in its list.

Padding, spaces or Control characters after segment terminators are ignored

Look after the segment for blanks, tabs, or other extraneous characters.

Possibly "*name*" was intended

This is not a stand-alone diagnostic, but appends to the end of another diagnostic such as #10600.

Previous Functional Group, Ctl. No. *n* not properly terminated

This segment (GS or UNG) starts a new functional group, but the functional group trailer (GE or UNE) is missing from the previous group.

Printable delimiters used, though UNOA syntax level not used

This UNA specified binary delimiters. This is supposed to override delimiters specified in the UNB. However, the UNB uses printable ones like : + ' so Analyzer will assume that these are going to be used in the rest of the interchange. This error can happen if the data

includes multiple interchanges in the same file, but include only one UNA. A UNA only applies to the UNB that immediately follows it. Example UNA and UNB:

```
UNA:+.? '
UNB+UNOB:1+012345678100001+987654321399999+950322:1431+0+++++TYPE TEST 202R-01+1'
```

Segment Tag is not alphanumeric

Segment tags (the first characters in each segment: ST, BEG, UNH, etc.) must be alphanumeric and start with a letter. Also check nearby segment delimiters.

Segment terminator is suspicious - shouldn't be alphanumeric

For X12 data, Analyzer looks at the last character in the ISA to identify the segment terminator. For EDIFACT data, it looks in the UNA. The segment terminator should be something that does not appear in the actual data.

Segment *xxx* can't be aligned within <*loop, group, transaction set, or message*>

This segment, loop, or group appears to be out of place. If this is an X12 transaction, and if *xxx* is part of a loop bounded with LS and LE, be sure the value in the LS and LE corresponds to the first segment in the loop (the loop trigger) in transaction set or message *yyy*.

Segment *xxx* doesn't exist in standard *yyy*

Examples*:*

"Segment CUL doesn't exist in standard 003042." The two most likely causes are that CUL should not be in the data, or you are analyzing against the wrong standard.

"Segment UNG doesn't exist in standard (UnNamed)." This example resulted when the UNG was in the default enveloping standard UN1ICS, which Analyzer could not read because it had accidentally been turned into a word processor file.

Explanation: Is the segment tag correct? Are you analyzing against the correct standard? You can require Analyzer to ask you for a standard before each transaction set or message under **Options | Analyzer | Ask for Standard Name.**

Segment *xxx* terminated by control character

Is there a segment terminator and is it correct?

Shorter than minimum length of *n*

The check digit parameter specified in Standards Editor requires that this value be at least *n* characters long, including the check digit (last digit) itself.

Skipping for the start of interchange

Are all enveloping segments present and correct? This diagnostic can appear if Analyzer becomes lost during an analysis and cannot realign itself. It skips the rest of the interchange in which it became lost.

Standard *name* cannot be loaded

The standard could not be loaded. Perhaps it is damaged. Contact TIBCO Foresight technical support.

Sub-Element separator same as Data Element separator

The ISA's last element is the sub-element separator. Analyzer can determine the element separator from the ISA, since each of its elements is fixed length. These two types of separator should not be the same. Otherwise, it won't be possible to properly analyze sub-elements in a composite. If it's true that sub-element separators and element separators are the same in the data, then Analyzer will interpret sub-elements as elements and you will get diagnostics when you reach any segment containing a composite.

Sub-element separator seen in elementary data element at *location*. Excess ignored.

There should not be a sub-element separator at this point. It is being ignored. Check the number of sub-elements in this composite, and also be sure that the data itself does not contain the sub-element separator. Also check to be sure that you are checking against the appropriate standard.

Superfluous Segment Delimiter Ignored

Does the file have two consecutive segment terminators?

Suspicious data found in *element*, may be undisplayable or cause problems in transmission

EDI data element values should generally be printable data, except the ASC X12 ISA16 (the sub-element delimiter) or when the type is Binary.

Syntax Error for *xxx* (*element number*) at col. *nn* "*value: reason*"

Example: *"Syntax Error for BGN04 (D.E. 337) at col. 29 "ABCDEFGHIJK": Time required."* This diagnostic warns that the contents of the 4th (04) element in this BGN (data element 337), contains the data "ABCDEFGHIJK." It should contain time data.

Explanation: The contents of the element at *xxx* in this segment does not conform to the length or the type required for this element when used in this location. The erroneous value, which is shown in quotation marks at the end of the message, begins in column *nn*.

Too many leading zeros. Max is *n*

The check digit parameter specified in Standards Editor requires that this value have no more than *n* leading zeros.

Trailing Blanks in *element* and *location*

Example*: Trailing Blanks in PKG04 (D.E. 754) at col. 13*. The PKG segment's 4th element (PKG**04**), data element 754, ends with blanks.

Transaction Set Ctl No. *n* in SE doesn't match ST

If the Transaction Set Control Number (data element 329) in ST02 does not match that in SE02, something is wrong. Is a ST or SE segment missing? Are one or both of the values wrong?

UNA conflicts with delimiters used in UNB

The delimiters in the UNA do not match what is actually being used in the UNB and with the contents of the first element in the UNB. Since they don't match, Analyzer is going to use whatever is in the 4th character position of the UNB for the segment delimiter and whatever is at the end of the UNB for the segment terminator. The sub-element separator that is being used will be shown in another diagnostic message.

UNA has no meaning except immediately before UNB

The UNA cannot appear *within* an interchange.

UNA is too short

The UN/EDIFACT Service String Advice (UNA) is a fixed field record of length 8 plus the segment terminator. It may appear before a UNB to override the default delimiters given in the first element of the UNB. A UNA only has meaning right before an interchange header, and it only affects the interchange that immediately follows it. If the transmission contains more than one interchange, a single UNA will only override delimiters for one of them. Without the UNA, default delimiters come from the first element in the UNB, or from the delimiters that are actually being used in the UNB.

Example of a UNA: **`UNA:+.? '`**

For a description of the UNA layout, see the message **10218 Could not interpret UNA - ignored**.

UNA overrides previous UNA

The data contains two consecutive UNA segments. Each UNA segment must be immediately followed by a UNB and will affect only that interchange, even if there are other interchanges in the same file.

Unexpected end-of-file

Analyzer stopped because it became lost when processing a segment.

Unrecognized Segment Tag, cannot continue

The segment tag (characters before the first delimiter in the segment) may be numeric, too long, or empty. Processing stops.

Unsupported Interchange Type

Analyzer does not recognize the ID of the interchange header segment, which determines which standard to use. Analyzer supports ISA, ICS, UNB, and BG.

Value "*xxx*" is an invalid "*appl. value list desc. (listname)*" which is used for *location*.

Explanation: An application values list called ***listname*** has been attached to this element at this location, either directly or through a business rule. The value *xxx* does not appear on that list (application values are case sensitive). Application values are attached in Standards Editor and can be viewed in Standards Reference. Please see "Application Value Lists" in the index of the SE manual. Checking application values is optional. Look under **Options | Analyzer Profile | Check App. Data Values** in the Analyzer menus.

Example: *Value "USA" is an invalid "3-digit Country Code" (COUNTRIES) for CUR02 (D.E. 100) at col. 8*

This message appeared because the 2nd element in the CUR02 at this location (data element 100) has an application values list attached, but that list does not include the value "USA". The value list may or may not be associated with a business rule at this location.

Value "*xxx*" is an invalid '*name*' for *name*

This diagnostic is often paired with #10617.

The data is invalid for this kind of field. For example, the value may contain leading zeros. Analyzer will suggest a possible value.

Warning - *element and location* doesn't conform to Level <*A or B*> character set

Example: *Warning - NAD02-C08201 (D.E. 3039) at col. 8 doesn't conform to Level A character set.*

The 2nd item in the NAD (NAD**02**) is composite C082. The first item in this composite (C082**01**) is data element 3039, which starts in column 8. It contains data that doesn't conform to UNOA. It may be lower case, for example.

xxx at *location* present, but was marked "*usage*"

Example: *"Segment SN1 at 2-102 present, but was marked "Not Used".*

This segment should not be included in the data, according to the guideline being used for comparison.

xxx HL is required in the previous HL order

A business rule attached in Standards Editor requires a HL loop with Hierarchical Level Code (HL03) *xxx*.

xxx HL loop can't be inside another HL order.

xxx is the Hierarchical Level Code (HL03) for the current HL loop. It must be the outer loop, according to a business rule attached to it in Standards Editor.

xxx HL loop must be inside *yyy* HL order.

xxx is the Hierarchical Level Code (HL03) for the current loop. It must be inside another HL loop with a Hierarchical Level Code of *yyy*, according to a business rule attached to it in Standards Editor.

7 Index

TIBCO Documentation and Support Services

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the TIBCO Product Documentation website, mainly in HTML and PDF formats.

The TIBCO Product Documentation website is updated frequently and is more current than any other documentation included with the product. To access the latest documentation, visit https://docs.tibco.com.

Product-Specific Documentation

Documentation for TIBCO® Foresight® EDISIM® is available on the TIBCO Foresight® EDISIM® Documentation page.

The following documents for this product can be found on the TIBCO Documentation site:

- *TIBCO Foresight® EDISIM® Release Notes*
- *TIBCO Foresight® EDISIM® Data Types*
- *TIBCO Foresight® EDISIM® Documentation and Demo Data Index*
- *TIBCO Foresight® EDISIM® Supported File Formats*
- *TIBCO Foresight® EDISIM® Installation Guide*
- *TIBCO Foresight® EDISIM® Introduction to EDISIM®*
- *TIBCO Foresight® EDISIM® DocStarter: Creating a Guideline from EDI Data*
- *TIBCO Foresight® EDISIM® Guideline Merge*
- *TIBCO Foresight® EDISIM® Document Builder User's Guide*
- *TIBCO Foresight® EDISIM® Error Message Numbers, Editing, and Management*
- *TIBCO Foresight® EDISIM® Validator User's Guide*
- *TIBCO Foresight® EDISIM® Using Flat Files*
- *TIBCO Foresight® EDISIM® Library User's Guide*
- *TIBCO Foresight® EDISIM® Validation Profile Files (APF)*
- *TIBCO Foresight® EDISIM® Using XML*
- *TIBCO Foresight® EDISIM® Comparator User's Guide*
- *TIBCO Foresight® EDISIM® Analyzer User's Guide*
- *TIBCO Foresight® EDISIM® Standards and Guidelines Reference Manual*
- *TIBCO Foresight® EDISIM® Test Data Generator User's Guide*
- *TIBCO Foresight® EDISIM® Self-Paced Tutorial: Introduction to EDISIM® (X12 Standards)*

- *TIBCO Foresight® EDISIM® Self-Paced Tutorial: Introduction to EDISIM® EDIFACT D99A Orders*
- *TIBCO Foresight® EDISIM® Standards Editor User's Guide*
- *TIBCO Foresight® EDISIM® Business Rules*

How to Contact TIBCO Support

You can contact TIBCO Support in the following ways:

- For an overview of TIBCO Support, visit http://www.tibco.com/services/support.
- For accessing the Support Knowledge Base and getting personalized content about products you are interested in, visit the TIBCO Support portal at https://support.tibco.com.
- For creating a Support case, you must have a valid maintenance or support contract with TIBCO. You also need a user name and password to log in to https://support.tibco.com. If you do not have a user name, you can request one by clicking Register on the website.

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the TIBCO Ideas Portal. For a free registration, go to https://community.tibco.com

Legal and Third-Party Notices