

TIBCO Foresight® EDISIM®

Test Data Generator User's Guide

Version 6.20.0

May 2021



Contents

1	Introducing Test Data Generator.....	1
	Where Test Data Generator Fits.....	1
	Outbound Flow	2
	Inbound Flow	2
2	TDG Tutorial	3
	Introduction.....	3
	Useful Terms	3
	Starting TDG	3
	Watching the Titles	4
	Steps in Creating Test Data	4
	Step 1: Building a New Message/Set Model	5
	Step 2: Building the Enveloping Model	14
	Step 3: Generating Data	16
	Next Time	18
	Backing Up Your Work	19
	Additional Features.....	19
	Print	19
	TDG Utilities.....	20
	Thank You for joining us in this Introduction.....	20
3	Basics	21
	Location of Files	21
	Model Names.....	22
	Data Generated	22
	Nuts and Bolts of Navigating	23
	The Hierarchy	23
	Menu	23
	Guide to Keystrokes	23
	Guide to Colors and Symbols	24
	The Toolbar	26
	Navigating a List	27
4	TDG Steps: Overview	29
	Steps in Using TDG to Create Test Data	29
	Overview of Selecting or Building a Message/Set Model	30
	Overview of Selecting or Building an Enveloping Model.....	31
	Overview of Assembling the Script.....	32
	Overview of Creating the Test Data.....	33
5	TDG Steps: Details	35
	Step-By-Step Details of TDG Use.....	35
	Message/Set Models	37
	To Create or to Reuse a Message/Set Model?.....	37

	Getting to the Message/Set Model Screen	38
	Top Pane: the Top of the Message/Set Model Screen	39
	Detail Pane: the Middle of the Message/Set Model Screen.....	47
	The Two Data Panes: The Bottom of the Message/Set Model Screen	47
	Deleting a Message/Set Model	54
	Exporting a Message/Set Model	54
	Saving a Message/Set Model	54
	Exiting the Transaction Set Model Screen	55
	What if I Later Change the Guideline or MIG in Standards Editor?	55
	Enveloping Models.....	55
	To Create or to Reuse an Enveloping Model?	55
	Getting to the Enveloping Model Screen.....	56
	Customizing your Enveloping Model.....	56
	Deleting an Enveloping Model	63
	Exporting an Enveloping Model	64
	Saving your Enveloping Model	64
	Exiting the Enveloping Model Screen	64
	Assembling the Script	65
	To Reuse or To Create a New Script?	65
	Getting to the Script Window.....	65
	Customizing your Script.....	66
	Deleting a Script.....	67
	Exporting a Script.....	68
	Generating Data	68
	Generating Data from a Script	68
	Generating Data from a Message/Set Model.....	69
	While TDG Creates your Data	69
6	Common Commands	73
	Save or Save As	73
	Copying a Model.....	74
	Export and Import.....	74
	Exporting and Importing from within TDG	74
	Exporting and Importing from the Command Line	75
	Export File Compatibility	76
	Print.....	76
7	Functions.....	79
	What are Functions?	79
	Complete List of Functions.....	80
	LOOPID Example	83
	Forward Referencing.....	83
	Functions in Default Values	84
	Envelope segments	84
	CTT Segment and Functions	85
	HL Segment and Functions	86
8	Advanced Topics	89
	External Models	89
	Creating an External Model	90
	Using an External Model.....	90

	Editing an External Model	91
	Detaching an External Model (Embed)	91
	Replacing an External Model with Something Else.....	91
	Deleting an External Model	92
	How Values Lists Cycle	92
	Segment Repeated by Segment Repeat Count.....	92
	Segment Repeated on Different Lines within a Message/Set Model	93
	Segment Repeated by Loop Repeat Count within a Message/Set Model.....	94
	Backing Up and Restoring Your Work	98
	If Your PC Becomes Locked Up	99
	Incomplete Saves	99
	Wrapping and Folding Data with EDITWRAP	99
	Wrapped and Unwrapped Data	100
	Unwrapping, Wrapping, and Folding Data with EDITWRAP	100
9	Samples	105
	Sample Scripts	105
	Sample Message/Set Models	105
	Sample Enveloping Models	105
10	Appendix A: TDG Sample HIPAA Transaction Models	107
	Introduction.....	107
	Sample HIPAA Models based on X12-4010	108
	Sample HIPAA Models based on X12-5010 Errata	110
11	Index	113
	TIBCO Documentation and Support Services	117
	How to Access TIBCO Documentation.....	117
	Product-Specific Documentation	117
	How to Contact TIBCO Support.....	118
	How to Join TIBCO Community	118
	Legal and Third-Party Notices	119

1 Introducing Test Data Generator

Where Test Data Generator Fits

TIBCO Foresight® EDISIM® Test Data Generator (TDG) lets you create incoming EDI data similar to what you will receive from trading partners or outgoing EDI data similar to what will come out of your translator. It can produce test data independently of those systems.

TDG can make data based on X12, EDIFACT, industry subsets like HIPAA, VICS, and UCS, and your own guidelines and MIGs created with Standards Editor.

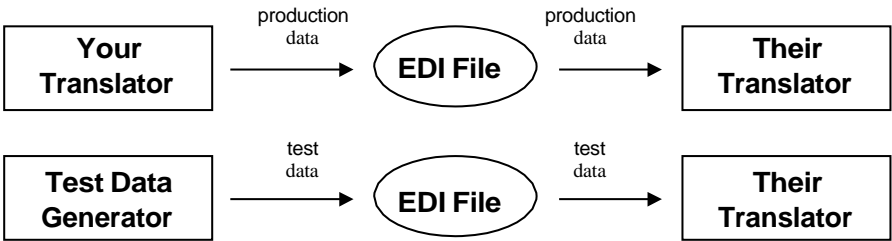
TDG therefore provides you with test data:

- When the translator or trading partner cannot provide it.
- When systems are not ready yet.
- When the people who know how to create test data cannot afford the time.
- When generating the data would make updates to production files.
- When you want to simulate error conditions and verify that they will be properly recognized and handled.
- When you are performing high volume (stress) testing and performance testing.

Finally, TDG serves as an excellent organizational tool. Through its use of scripts, you can organize thorough regression tests for use whenever there is a major change such as a new translator or hardware platform.

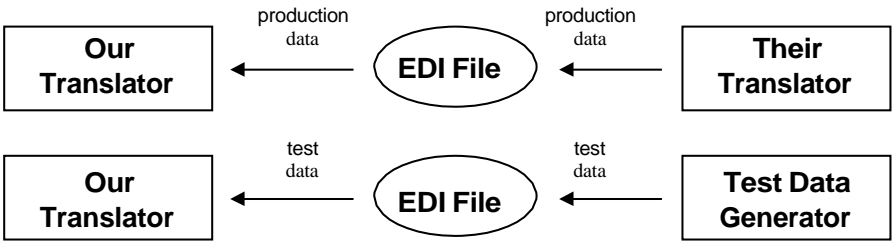
Outbound Flow

TDG simulates outbound EDI data as your translator will eventually create it. The top figure shows the fully developed process, and the next figure shows how TDG simulates the EDI file.



Inbound Flow

TDG simulates inbound EDI data that will be sent from your trading partners to your translator. The top figure shows the fully developed process. The bottom figure shows how TDG simulates EDI files that will be sent to you by your partners.



2 TDG Tutorial

Introduction

After completing this tutorial, you will have used the main TDG features and generated test data. This exercise is based on X12-4010.

Useful Terms

Besides the usual EDI terms, TDG uses the following:

Message/Set Model	A customized version of a transaction set or message.
Enveloping Model	A customized version of enveloping.
Script	Enveloping models linked to message/set models, representing one EDI data file

Starting TDG

Select **Start | Programs | <TIBCO_HOME> | EDISIM | Test Data Generator**, or use file explorer to double-click on **Fstdg.exe** in EDISIM's **Bin** folder.

If TDG does not fill up the entire screen, maximize it by double clicking on the title bar.

Watching the Titles

Keep your eye on the title to be sure of what you are editing:



Steps in Creating Test Data

To set up the data in the transaction (segments ST through SE):

1.	Build or select a message/set model	This describes X12's ST through SE segments, or EDIFACT's UNH through the UNT segments.	See Step 1: Building a New Message/Set Model on page 5
2.	Build or select an enveloping model	This describes the enveloping segments that go with the message/set model.	See Step 2: Building the Enveloping Model on page 14
3.	Generate the test data file	This creates an EDI file.	See Step 3: Generating Data on page 16.

Step 1: Building a New Message/Set Model

We'll choose a standard and a transaction set from your EDISIM® database. These include TIBCO Foresight-supplied standards plus any guidelines or MIGs you have created with Standards Editor.

Then we'll work with it until it describes the transaction set that we're going to use for a particular test with our imaginary trading partner Rusty Fishing Outfitters.

Although we'll be working with an X12 standard for this example, our activities would be similar if we were to use EDIFACT, UCS, VICS, HIPAA, etc. Terminology will vary, however:

X12	EDIFACT
Guideline	MIG
Transaction set	Message
Loop	Group

We are going to use a published X12 standard for these exercises.

Your action From within TDG, choose **File | New | New Message/Set Model**.

This is a directory of all standards, guidelines, and MIGs known to EDISIM. You can use the scroll bar on the right edge of the screen to navigate more quickly.

We're going to create a new model of an X12-4010 purchase order.

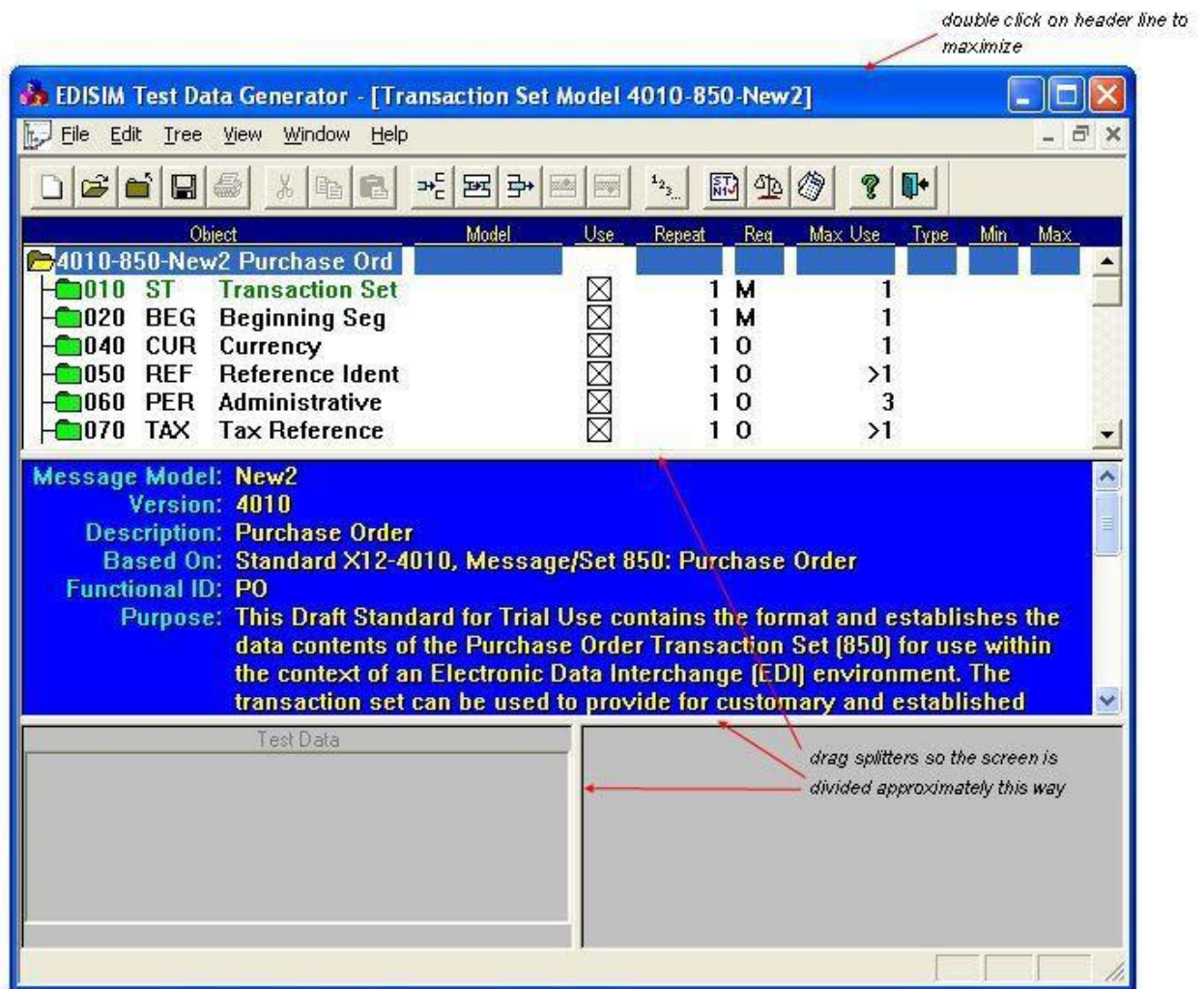
Your action In the top pane, choose **X12-4010**.

In the bottom pane, choose **850 Purchase Order** (click in the bottom area and type 8 to get to the transaction sets that start with 8).

Click **OK**.

Adjusting the Screen

We want to adjust the screen so that it looks like this:



Your action Double-click on the **Transaction Set Model** header to maximize the model's window.

Pass the mouse over each splitter bar until it turns into a double-headed arrow. Now you can drag the bar to adjust the spacing in the panes.

Viewing Information about the Objects on the Screen

The top pane shows all loops and segments in X12-4010's 850. The columns show:

Object The location, ID, and name of each object

Editable in TDG:

Model	External model being used (if any). This is a reusable, customized version of a loop, group, segment, composite, or element that has been saved with File Save Externally . See External Models on page 89.
Use	Shows whether the object will be included in the data at all. You can toggle the X on or off with a mouse click, with the space bar, or with <i>Ctrl+G</i> .
Repeat	The number of times the segment, loop, or group will appear at this location in the test data. To change it, click on the number and type a new one over it, or use the plus and minus keys on your numeric keypad.

Not editable in TDG:

Req	Requirement (optional, mandatory, conditional, etc.) as specified by the underlying standard. If you had based your model on a guideline or MIG instead of a published standard, look in the blue detail pane to see the company requirement or status for the currently-highlighted item.
Max Use	The maximum number of times the item can be used here.
Type	The data type: ID, AN, DT, R, etc. (element only).
Min	The minimum length of the data (element only).
Max	The maximum length of the data (element only).

Your action Use the scroll bar on the right side of the top pane to scroll down until you see the **SAC** loop at position 120. Notice its icon: a green folder with a looping arrow inside.

Click on the **SAC** loop and look in the blue detail pane to see information about it. Use the scroll bar, if necessary, to see the rest of the information in the detail pane.

Marking Segments and Loops as Unused

We are going to omit some segments from the test data by toggling the Use column off:



This element is to be included in the test data.



This element is to be excluded from the test data.

Your action Highlight the **CUR** segment at position 040 and click on its **Use** column or press the space bar. The X disappears from the check box, and its text becomes

faded. This indicates that it will be excluded from the data.

Now, mark each of these as unused also:

PER	TAX	FOB	CTP	PAM	CSH	
SAC loop		ITD	DIS	INC	LDT	SI
MEA	PWK	PKG	TD1	TD5	TD3	TD4
MAN	PCT	CTB				
N9 <i>loop</i>						


















We want to exclude the **AMT** segment too, but we cannot see it.


Your action Double-click on the **CTT** loop near the bottom and clear the **X** from its Use column.

If you try to mark a mandatory object as unused, TDG warns you and gives you the option of canceling. However, TDG will allow you to mark a mandatory object as unused so that you can test that condition.

OOPS! If you mark the wrong segment as unused, you can toggle its use back on with another click.

Your message/set model should now resemble the following:

	Object	Model	Use
	250 TD3 Carrier Details [<input type="checkbox"/>
	260 TD4 Carrier Details [<input type="checkbox"/>
	270 MAN Marks and Num		<input type="checkbox"/>
	276 PCT Percent Amount		<input type="checkbox"/>
	280 CTB Restrictions/Co		<input type="checkbox"/>
	285 TXI Tax Information		<input checked="" type="checkbox"/>
	287 AMT Monetary Amou		<input checked="" type="checkbox"/>
	295 N9 Reference Ident		<input type="checkbox"/>
	310 N1 Name		<input checked="" type="checkbox"/>
	430 LM Code Source Inf		<input checked="" type="checkbox"/>
	450 SPI Specification Id		<input checked="" type="checkbox"/>
	610 ADV Advertising De		<input checked="" type="checkbox"/>
	010 PO1 Baseline Item D		<input checked="" type="checkbox"/>
	010 CTT Transaction Tot		<input checked="" type="checkbox"/>
	010 CTT Transaction T		<input checked="" type="checkbox"/>
	020 AMT Monetary Am		<input type="checkbox"/>
	030 SE Transaction Set		<input checked="" type="checkbox"/>

Each loop is preceded with a dark green folder with a looping arrow:  This alerts you that it is a loop and not the segment with the same name.

Let's customize the PO1 loop by excluding unwanted segments and subloops in it.

Your action Double-click on the **PO1** loop. Mark all its segments and loops in it as unused except the PO1 segment, the PID loop, and the REF segment. The PO1 loop goes all the way down to the CTT loop.

Scroll up to the PO1 loop and double click to stop displaying its subordinates.

Making an External Loop Model

Since we may need this particular configuration of the PO1 loop again, we decide to give it a name and make it globally available. These external models give you consistency and save time.

Your action Highlight the PO1 loop header and choose **File | Save Externally**.


At the Save as External Loop Model box, name it **SMALLPO1**. Do not press *Enter*.

For the Description, type **Small PO1 loop with PID and REF**.

Choose **OK**.

We made an external loop model, SMALLPO1, but we haven't used it anywhere yet.

Your action With the PO1 loop highlighted, choose **Edit | Replace** or click on the replace button:



In the Select Item box, choose **Types | Loop Models for PO1 | PO1-SMALLPO1 | OK**.

Click elsewhere so that you can see the shading behind the PO1 loop. This is a visual clue that an external model is being used.

The shading, and the SMALLPO1 in the Model column, show that this PO1 loop is using an external model. You may use SMALLPO1 anywhere you see a PO1 loop. For details, see [External Models](#) on page 89.

Saving

As with all PC software, it is wise to save your work frequently when using TDG.

Your action Select **File** | **Save As**.


Enter **BASICPO** for the name, press *Tab* to get to the Description area, and type **Basic X12-4010 Purchase Order**. Click **OK**.

Now the title bar shows this name.

Inserting Segments and Loops

You can insert segments to intentionally create errors or to restore segments you accidentally delete. If you had based this model on a guideline or MIG rather than a published standard, this is the way to include unused segments, loops, and groups.

Your action Highlight **150 DTM**.
Choose **Edit** | **Insert** or use the Insert toolbar button



Click on **Types** and look at all the types of object you can insert here.

Your choices for objects to insert include:

- **Set 850 Purchase Order:** All segments in the 850 transaction set, in the same order in which they appear.
- **Segment Dictionary:** All segments in the X12-4010 dictionary, in alphabetical order.
- **Segment Models:** All external segment models known to TDG.
- **Loop Models:** All external loop models known to TDG.

We want to insert two segments right from the dictionary.

Your action Choose **Segment Dictionary**.
Click on the **ITD** segment and *Ctrl*+click on the **TAX** segment.
With the **Above** button selected, click **OK**.

The ITD and TAX segments now appear above the DTM at position 150. They have no position numbers.

Repeat Counts

The **Repeat** column shows how many times a segment will appear in your test data at the current position. It is initially set to 1, but can be changed.

Your action Click on the Repeat column for **150 DTM**.
Change it to **2**.

Press the **Save** button on the toolbar: 

Two DTM segments will be created at this location in your test data.

Data Elements

Now let's view data elements within a segment.

Your action	Double-click on the inserted ITD segment to see its elements.
--------------------	--

Let's mark some elements as unused.

Your action	For element 446 at ITD06 , click on the Use box to toggle the X off. Turn off the Use for the elements at positions 07-15 in the same manner.
--------------------	--

These elements now have faded text.

Viewing and Changing Values

Although each element comes preloaded with at least one default value, you can change the values. Depending on the type of element, values can be:

- Literal values
- Code values
- Functions
- Application values (if you based your model on a guideline or MIG instead of a published standard).

When you highlight an element, the Test Data pane at the bottom left shows the list of data that will actually be used in the test data file. You can change this list by moving values up or down, by deleting them, by adding to the list, or by modifying an existing value.

To modify an existing value, double-click on it.

One way to add to the list is to drag values over from the data values directory pane at the bottom right. The values directory contains folders that will display their contents if you double click on them.

You can then either double-click on a value to move it to the Test Data list, or you can select several with *Ctrl*+click then drag them over to the Test Data list.

Default Values

To see the default values for the highlighted element, double-click on the Default Values folder in the bottom right pane.

Your action Click on element **333** at position ITD02. The Test Data pane is preloaded with default values 1, 8, and 3. Double click on the **Default Values** folder. There are three default values.

Since every element has default values, you could generate your test data without modifying any element's value list.

Cycling through Test Data Lists

To generate the test data, TDG cycles repeatedly through the Test Data list. The first time element 333 is generated its value will be 1. The second time, it will be 8. The third time, it will be 3. The fourth time, it will be 1 again.

For details on value lists and cycling, please see [How Values Lists Cycle](#) on page 92.


Inserting a Literal Value

You can change the Test Data list, which initially has only the default values.

Your action Choose element **338**.

The value list is displayed. The top pane shows this information about 338:

- Minimum Length: 1
- Maximum Length: 6
- Type: R (numeric with decimal)

Your action Click on the top value (.0035) in the Test Data values pane at the bottom left. Select **Edit | Insert** or click on the Insert toolbar button: 

In the value editing box, type **.065**.

Press **Add**.

Since we are finished, press **Cancel**.


Inserting a Code Value

Element 333-Terms Basis Date Code has an ID Type. This means this element has code values maintained by ASC X12. You can see these official values and choose from among them.

Your action Highlight element **333**.

Double-click on the **Code Values** folder at the bottom right.

Use the scroll bar to go down to code **ZZ** and double-click on it. Code ZZ has now been inserted at the end of the value list.

Move **ZZ** to the top of the Test Data list by highlighting it and then clicking three times on the **Move Row Up** toolbar button: 

Using a Function

Another way to supply a value for an element's value list is to enter a function. We will insert the date function into an element.

Your action Highlight element **370** at position ITD04 and look at the {DATE} function in the Test Data pane.

To see all functions, double-click on the **Functions** folder at the bottom right.

If we wanted today's date, we would leave this as is, but we want to use 10 days ago. Functions with a plus-minus symbol can be used in computation.



To edit something in the Test Data list, double click on it.


Your action Double-click on {DATE} in the Test Data list (on the left) and change it to read: {DATE-10}, then click **OK**.

Ignore the date required warning, since our function will yield a date.

Errors

You can enter a value that violates the conditions of an element, but TDG will warn you. For example, let's enter a value that is the wrong type into element 370's value list.

Your action Click on the {DATE-10} function in the Test Data

list and select **Edit | Insert**, or use 

Type **9374** and click **Add**.

Click **OK** to clear the warning. 9374 remains in the value list.

Click **Cancel** to close up the Data Value box.

TDG lets you leave the erroneous value, in case you want to force an error into your test data. If the error is unintentional, you can delete the value.

Your action Highlight **9374** and press the *Delete* key on your keyboard.

Close the model with **File | Close** or with the closed file folder toolbar button: 

Say **Yes** when asked if you want to Save.

You have now finished the most time-consuming part of using TDG: creating a message/set model. In the next section, you will learn how to build the enveloping.

Step 2: Building the Enveloping Model

The second step in building test data is to set up enveloping. Once defined, this can be used over and over. In this section, we'll set up outbound enveloping for our test data files.

Your action Select **File | New | New Enveloping Model**.

We want to have Y2K enveloping, with the 8-byte date field in the GS segment.

Your action Choose Envelope Type **X12 (Y2K) (ASC X12 Standards 3072 and later - ISA, IEA)**.

We want as few interchanges and groups as possible in each file. This only makes a difference when you have more than one transaction set or message in the data file. This is explained more fully in [Enveloping Method](#) on page 59.

Your action Change Envelope Method to **Simple**.

Check the values in the ISA and GS.

Your action Click the three ellipses button to the right of the **ISA - Interchange Control Header** field.

Change these values:

ISA05=**ZZ**

ISA06=**RUSTYHOOK** (pad with 6 trailing spaces)

Save and close the ISA.

Now edit the GS and change the GS07 to always be **X**.

Save and close the GS.

It's time to save the enveloping model.

Your action Choose **File | Save As**.

Name the copied enveloping model **RUSTYOUT**.

Tab to Description, and type **Rusty Hook Corp. in Oregon - outbound enveloping**.

Choose **OK**.

Setting the Counters

You can set the starting number for the counters that generate the control numbers in envelope segments.


Your action Change the counters to **4** for interchange, group, and transaction.

Setting Delimiters

The lower right area lets you change delimiters. These will be placed at the end of every segment, element, and subelements in the test data.

You can represent your delimiters by the actual keyboard character or by a hexadecimal value.

Since they are appropriate for us, let's leave them as they are. They are pre-set to the most commonly used delimiters.

Your action Close the enveloping model with **File | Close** or with 
Save when asked.

You have now defined your enveloping for this partner. If we put this together with the customized transaction, we will have all the information we need to create test data. Let's do that now.

Step 3: Generating Data


We have defined enveloping and a message/set model, so we have everything we need to create data.

Generating Data from a Message/Set Model

If we want only one transaction in the test data file, we can create data directly from a message/set model.

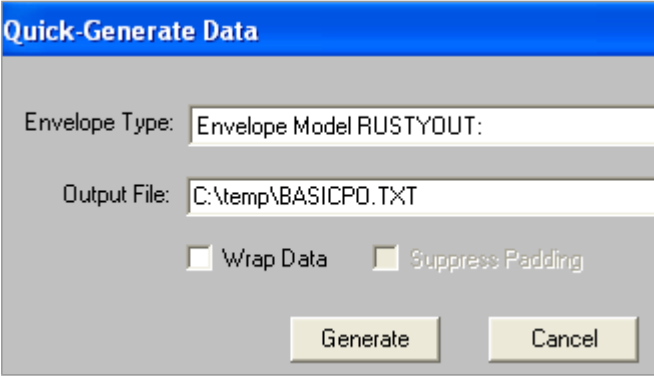
Your action **File | Open | Object Types Message/Set Models**
 | BASICPO | Open.

From here, we can generate a file containing enveloping and a transaction set.

Your action Click the  button on the toolbar. (Generating data is as easy as 1-2-3.)

For Envelope Type, choose **RUSTYOUT**.

For Output File, change the path to something that you can find easily.



The image shows a dialog box titled "Quick-Generate Data". It has a blue header bar. Below the header, there are two text input fields. The first is labeled "Envelope Type:" and contains the text "Envelope Model RUSTYOUT:". The second is labeled "Output File:" and contains the text "C:\temp\BASICPO.TXT". Below these fields are two checkboxes: "Wrap Data" (unchecked) and "Suppress Padding" (checked). At the bottom of the dialog are two buttons: "Generate" and "Cancel".

Click **Generate**.

The screen shows the progress of your request. When finished, a beep and the notice Test Data Generation Complete will alert you.

You can now look at the new data.

Your action Note the location of the file, then click **OK** to close the Test Data Generation Progress window.

From Windows Explorer, go to the folder shown during generation and double-click on **BASICPO.TXT**. This should open the file in a text

editor.

Your text editor may display segments as wrapped on your screen.

Your data should have segments terminated with ! and elements separated with *.

The DTM segment was repeated twice when you changed the use count to 2. Notice how TDG cycled through the values of elements within the DTM.

Generating Multiple Transaction Sets


To get more than one transaction in the file, we create data from a script. As with transaction or enveloping models, you can modify an existing script, create a new script, or use **File | Save As** with an existing script to copy it to a new name.

Your action Close your EDI data file.

Return to TDG and choose **File | New | New Script**.

You are looking at an empty Script Window.

Our first job is to add a message/set model to this script.

Your action Choose **Edit | Insert** or use 

Click the three dots at the end of the message/set model line. Choose **BASICPO** and press **Select**.

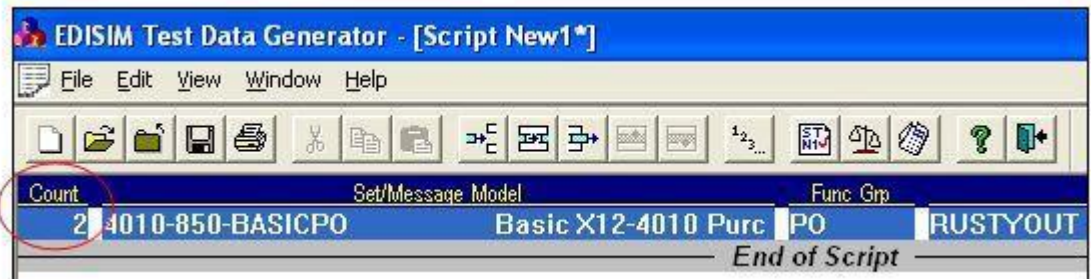
You need to choose an existing enveloping model to go with the message/set model.

Your action Click on the three dots to the right of the enveloping model line. Choose **RUSTYOUT** and click **Select**.

In the Script Line box, choose **OK** to return to the main script window. Change the Count column (at the far left) to 2.

The script screen is divided into two parts: a section showing which message/set model to use and another section showing which enveloping to use with it.

Your action Change the Count (at the far left) to **2**, because we want two transaction sets in the file.



You could add more lines to the script. A script makes one data file, regardless of how many lines it contains.


It's time to save the script.

Your action Choose **File | Save As**.

Type **BASIC850** for Name and *Tab* to the description field, where you should type **Two small 850s**.

Click **OK**.

Now that we have saved, we can generate data.

Your action Click the  toolbar button.

Specify an output file.

Click **Generate**.

When it finishes, open the output file and look for two ST segments.

Next Time

You can open a message/set model, enveloping model, or script.

Your action Use **File | Open**.

Select the appropriate Object Type.

Select the model or script from the list.

Backing Up Your Work

It's very important to protect your work by exporting scripts frequently and backing up the database files frequently. Let's back up our script BASIC850 to an external file.

Do this from within TDG.

Your action	Choose File Open Object Types Scripts BASIC850 Open . Choose File Export . Notice the folder and filename. Choose Save .
--------------------	--

You could import BASIC850.EXP into another PC's TDG, or re-import it into this one (see [Export and Import](#) on page 74).

You can also back up your entire database to protect against catastrophes such as hard disk failure.

Your action	Exit TDG. Use Windows Explorer to create a new folder called BACKUP1. Copy the contents of EDISIM's MODELS folder to it.
--------------------	--

Additional Features

This exercise has been a brief introduction into the capabilities of TDG. There are many other features that you will find useful.

Print

Print produces a text printout of a model that you may find useful to document your testing program. Other printing available in EDISIM:

- For professional-quality printed guidelines and MIGs, use EDISIM's Doc Builder.
- For text printouts of guidelines and MIGs, use Standards Editor or Standards Reference.
- For reports of differences between standards, guidelines, or MIGs, use Comparator.
- For reports of errors in EDI data, use Analyzer.

TDG Utilities

- **EDITWRAP** - Compress or wrap the test data in various ways (string segments end to end). See [Wrapping and Folding Data with EDITWRAP](#) on page 99.
- **Fsdosutl** - Mass import and export. See [Exporting and Importing from the Command Line](#) on page 75.

Thank You for joining us in this Introduction

Once you have created the core models and scripts that are key to your business, you will need to make only minor changes to create additional test data.

TDG is anchored in the EDI standards. The data you generate is in parallel with EDI data, without manually entering the data or referencing a standards manual.

If you have questions, contact TIBCO Foresight Customer Support.

3 Basics

Location of Files

TDG determines which folder to use when reading or writing a file as follows:

- The path that you specify when creating test data, for example, will be retained for future sessions. (This is stored in a file called Fstdg.ini in your workstation's folder under EDISIM's Users files folder.)
- The initial path is controlled by Edisim5.BaseSettings.ini in EDISIM's Bin folder on your local PC. Type or print this file, look at the list of folders, then read the description of what goes in each folder.

```
[Directories]
BASEROOT      = "C:\EDISIM50"

:* FORESIGHT Supplied .STD Standard Files
FSFACTORY     = "C:\EDISIM50\STATIC"

:* User-defined .STD Standard Files
ALLUSERSHARED = "C:\EDISIM50\User Files\Public Guidelines"

:* where the per-user (user_system) folders are stored
PERUSERBASE   = "C:\EDISIM50\User Files\Personal Folders"

:* Model Export directory
TDG exports   → = "C:\EDISIM50\User Export"

:* writable (per-user) Model Database Directory
TDG database  → = "C:\EDISIM50\Models"

:* Model Database Filename Prefix
ModPfx       = MODEL

:* Model Database ODBC Data Set Name
Modelsdsn    = TDG-Models50

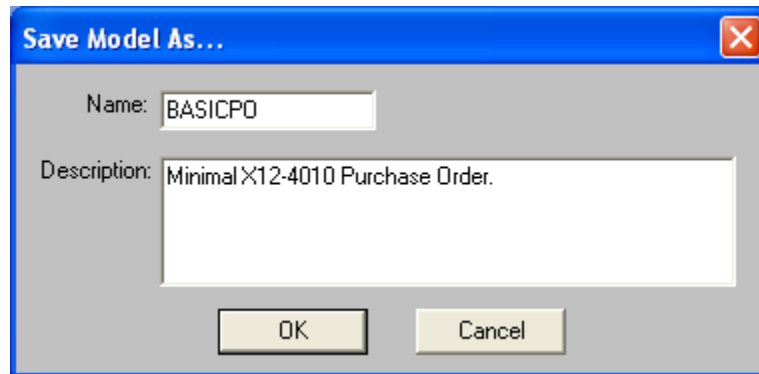
Used during installation → ult exported models imported during install
ps → = "C:\EDISIM50\TDGdefaultModels"

TDG-created .EDI files
- under DATABASE; Data Generator directory
<USER> is the workstation name → = "<USER>\TDG"

:* FORESIGHT Supplied .SEF files
Addlstds     = "C:\EDISIM50\Addlstds"
```

Model Names

When you name a model for the first time, you will see a box where you will enter a name and description.



Names that you give to your message/set models, enveloping models, scripts, and external models can be up to 8 characters long and include:

- Letters
- Numbers
- Underscores
- Dashes

Avoid:

- Other special characters
- Spaces
- Names which begin with '\$', since this indicates a TIBCO Foresight-supplied model that is likely to be overwritten when you upgrade EDISIM.

If the name exists, you will be asked to confirm.

It is important to customize the description, which will show up in selection lists and remind you about the model's purpose.

Data Generated

TDG will generate test data from the original guideline and populate every used element with data.

It does not take into consideration syntax rules or conditional relationships between data elements since it cannot determine which of the elements you would like to use.

You will need to tweak the data and ungenerate any elements that you do not wish to put in the test data.

Nuts and Bolts of Navigating

This section shows how to use your keyboard or mouse to get around TDG's menus and screens.

For details about particular screens, see:

Message/Set Model screen	page 39
Enveloping Model screen	page 56
Script screen	page 65

The Hierarchy

To see the subordinate parts of an EDI object in the top pane, double-click on it or highlight it and press *Enter*. To conceal the subordinates, double-click or use *Enter* again.

Menu

The TDG menu across the top shows the main actions you can take from your current location. There are two ways to select from the menu:

- **Keyboard:** Hold down the *Alt* key and press the highlighted letter in the command that you want. Example: press *Alt* + *F* to open the File menu, then press **S** to save.
- **Mouse:** Point your mouse at the desired menu choice then click.

Guide to Keystrokes

Your Action	Result
Cursor keys	Scroll one line or column at a time.
<i>Alt</i> + key	Chooses something from the menu. Hold down <i>Alt</i> and press the key that is underlined in your choice.
<i>Ctrl</i> + key	Executes a menu choice. These are listed on menus, next to each selection. Example: Click on the File menu and look next to New . The hot key is <i>Ctrl+N</i> . You can use it without opening the menu.




<i>Enter</i>	Displays (or conceal) subordinates of the currently highlighted entry, pick a selection, or end something you are typing.
<i>PgUp</i> and <i>PgDn</i>	Scrolls one screen at a time.
<i>Tab</i>	Moves to next field in a multi-field window such as the enveloping model screen.
<i>Del</i> key on keyboard	Deletes the currently highlighted value in the Test Data pane. In user-enterable fields, the <i>Del</i> key deletes the currently highlighted character.
<i>Space Bar</i>	Toggles (on-off) Use for a loop, segment, composite, or element.
<i>F1</i>	Opens Help.
<i>F6</i>	Moves to another pane.
When entering information into a field:	
<i>Delete</i>	Deletes current character.
<i>Home</i>	Moves to the beginning of the line.
<i>End</i>	Moves to the end of the line.
<i>Alt + Delete</i>	Deletes to the end of the line.

Guide to Colors and Symbols

Colors

 yellow	transaction set, message
 bright green	segment
 dark green	loop or group
 blue-green	composite
 blue	element
 red	value

Symbols in Top Pane

	Closed folder. Item has subordinates that can be displayed by double-clicking or pressing <i>Enter</i> on it.
	Open folder. Item has subordinates that are currently displayed. To conceal its subordinates, double click or press <i>Enter</i> on it.
	Dark green folder with a circular arrow. A loop (X12) or group (EDIFACT). To toggle display of its segments and subloops, double click or press <i>Enter</i> on it.



Blue-green folder, yellow C. A composite. To toggle display of subelements, double click or press *Enter* on it.



Blue circle. An element. If the blue circle contains a dot, this means someone has modified the Test Data list, and so default values are not being used.



First character is \$ in user-supplied part of name for TIBCO Foresight-supplied models.

Symbols in Data Values Directory



Red folder (with plus sign if closed). A folder for code values, default values, application values, or functions. To toggle display of the values in the folder, double click or press *Enter* on it.



Red page. A dictionary code value.



Red page, yellow F. A function.



Red page, yellow L. A local code value, added to the guideline or MIG by a Standards Editor user.



Red page, yellow A. An application value. The name of the application value list will be shown next to the application value folder at the top.



Red page, yellow D. A default value.

Symbols in Multiple Panes

Light gray background

(Top or Test Data panes) **An external model.** Look in Model column of the top pane.

Faded text

(Top or data values directory) **Unused segment or loop**, if Use column has no X, or unused code value.

{ }

(Either value pane) The value inside the curly braces is a **function**, not a literal value.

The Toolbar

To see what a toolbar button does, rest your mouse cursor on it. For more information, hold the button down and read the description on the status bar at the bottom. If you don't want to execute the button, drag the mouse cursor off before releasing.



File | New. Create a new message/set model, enveloping model, or script.



File | Open. Open an existing message/set model, enveloping model, or script.



File | Close. Close the active window (script or any type of model) prompting for unsaved changes if necessary.



File | Save. Save the model or script in the active window. If it has never been saved before, this will act as a File | Save As.



File | Print. Print the information in the active window.



Edit | Cut. Cuts highlighted values from the Test Data pane. Used with Paste or Copy, this can help you rearrange the list of values.



Edit | Copy. Copies highlighted values from the Test Data pane, so that you can paste them in another element's value list.



Edit | Paste. Pastes values copied with Edit | Copy into a values list.



Edit | Insert. Insert something before or after the currently selected item.



Edit | Replace. Replace the currently selected item.



Edit | Delete. Delete the currently selected object.



Edit | Move Up. Move up the currently selected value in the Test Data pane.



Edit | Move Down. Move down the currently selected value in the Test Data pane.



File | Generate Test Data. Generates test data from the current

script. Available in Script window.



Opens Analyzer.



Opens Comparator.



Opens Doc Builder.



Help | Index. Opens Help.



File | Exit. Prompts about saving changes and then closes TDG.

Navigating a List

Many panes will have lists of items. This section will give you tips on how to use them.

Seeing the Bottom of a List

Part of the list may extend below the bottom of the screen. The cursor arrow keys, *PgDn* key, or scroll bar can take you to the bottom. The location of the box on the scroll bar can give you a visual clue about the length of the list.

Selecting from a List

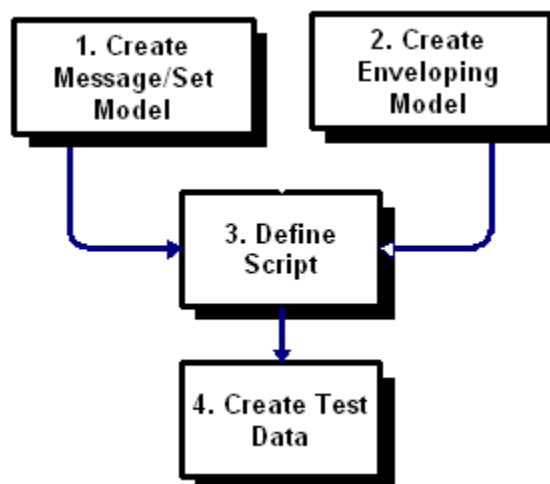
To act on a single entry in a list, click on it and issue the appropriate command.

In some lists, you can act on multiple items by selecting them with the *Ctrl*+click or *Shift*+click. Selected entries are in a different color on your screen.

4 TDG Steps: Overview

Steps in Using TDG to Create Test Data

1. Create a message/set model, which customizes a message or transaction set.
2. Create an enveloping model, which customizes control envelopes.
3. Define the script by linking message/set models with enveloping models.
4. Create test data.



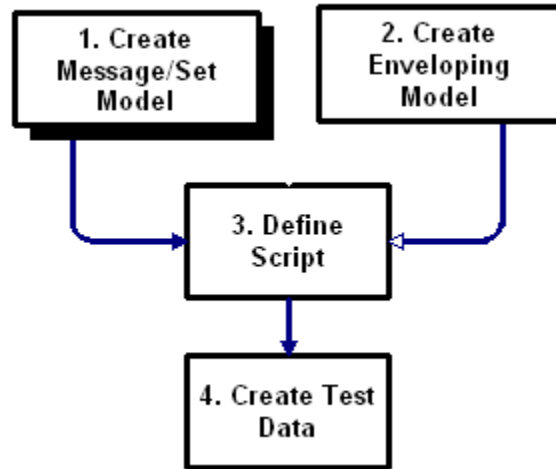
The rest of this chapter is a conceptual overview of this process. The next chapter will give a detailed description of each of these steps.

For more overview information, see:

- [Overview of Selecting or Building a Message/Set Model](#)
- [Overview of Selecting or Building an Enveloping Model](#)

- [Overview of Assembling the Script](#)
- [Overview of Creating the Test Data](#)

Overview of Selecting or Building a Message/Set Model



A message/set model is a unique subset of an EDI message or transaction set.

To create a new message/set model, use **File | New | New Message/Set Model**.

To edit an existing message/set model, use **File | Open | Object Types | Message/Set Models**.

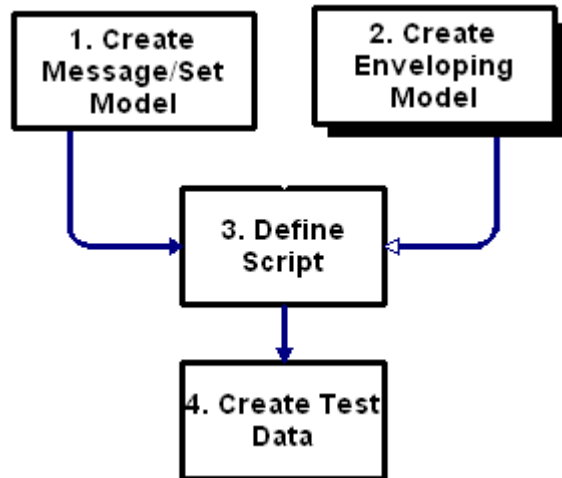
The models you create should contain only segments and elements involved in your test cases, or that you expect to send to or receive from a trading partner.

Example: Acme Co. receives purchase orders in several forms. It receives a subset of the ANSI X12 PO version 4020 from most of its customers. One customer, BIGCO Inc., uses different segments, which are defined by a separate message/set model. Still other customers send the older version 2040. Acme gives unique model names to message/set models of the same version and transaction, and creates the following models for all functional testing:

Ver	Trans	Model	Description
4020	850	REGPO	Model of usual X12 PO
4020	850	ODDUSE	Model sent only by BIGCO, Inc.
2040	850	REGPO	Older version sent by OLDCO
U3/3	875	ALLUCS	Sent by all UCS customers
4020	810	REGINV	Sent by most vendors
3070	997	ALLX12	All X12 acknowledgments

This process is explained fully in [Message/Set Models](#) on page 37.

Overview of Selecting or Building an Enveloping Model



An enveloping model contains the envelope (control) segments and enveloping method to be used with a group of one or more trading partners with similar characteristics (e.g., use of control numbers, acknowledgments). Example:

Acme has several customers that never increment the control number sent in their envelopes. Acme calls this group NONUM. Acme has created these distinct enveloping models:

Enveloping Model	Description
X12_GRP1	Most X12 customers using Y2K enveloping
X12_GRP2	Most X12 customers using non-Y2K enveloping
NONUM	Customers who don't use control numbers
BIGCO	BIGCO, Inc. only
OLDCO	OLDCO, Inc. and several others
UCS_GRP1	UCS customers
VENDOR_1	Vendors doing EDI

You create new enveloping models under **File | New | New Enveloping Model**.

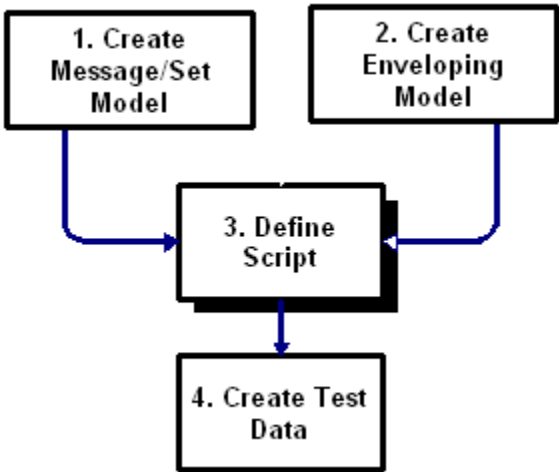
You edit existing message/set models under **File | Open | Object Types | Enveloping Models**.

The enveloping model describes delimiters, control numbers, whether there are one or more transactions or messages per functional group envelope (GS/GE or UNG/UNE) and whether there are one or more functional groups per interchange. You can also skip the functional group envelope altogether (as in some EDIFACT messages), or even skip the interchange envelope (as in some older TDCC transaction sets).

It does not matter whether you create the enveloping model or the message/set model first. You can re-use existing enveloping or message/set model, either as-is or by changing them and then saving them to a new name with **File | Save As**.

This process is explained fully in [Enveloping Models](#) on page 55.

Overview of Assembling the Script



A script links message/set models with enveloping models. Together they make an entire EDI file. Acme matches up their message/set models and enveloping models to form these pairs:

Message/set models

Envelope models

4020	850	REGPO
4020	850	REGPO
4020	850	ODDUSE
2040	850	REGPO
U3/3	875	ALLUCS
4020	810	REGINV
3070	997	ALLX12
U3/3	999	ALLUCS

Acme might put the first four pairs above into one script called **ORDERS** since they are all the same type of transaction (purchase orders).

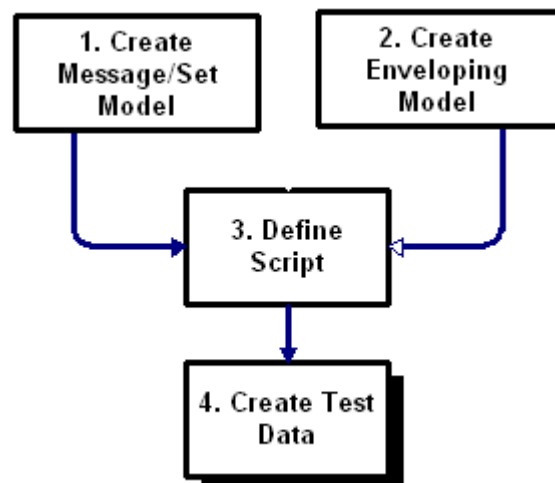
They might put the invoice (810) and the grocery products purchase order (875) into separate scripts, and the last two in a script called **ACKS**.


TDG does not restrict the way in which you group these pairs into scripts. Acme could group its pairs by version. Organize the scripts in the way that you find most useful. Each script makes one data file.

By default, each pair is included once when TDG creates data, but you can specify another quantity for any pair in the script. This is particularly useful in performance and capacity tests where you wish to simulate a representative mix of transactions.

This process is explained fully in [Assembling the Script](#) on page 65.

Overview of Creating the Test Data



To create test data, open the script, click in the top pane, and choose **File | Generate Test Data** or use the toolbar  button.

You have your choice of filename and folder for the test data file.

This process is explained in detail in [Generating Data from a Script](#) on page 68.

5 TDG Steps: Details

Step-By-Step Details of TDG Use

This section gives you step-by-step details of how to use TDG. It is assumed that you have done TDG in an Hour in the manual.

These are the steps you will take to create test data:

1. Building or selecting a message/set model.

In this step, you define what data goes into the segments from ST-SE or UNH-UNT. This is described in [Message/Set Models](#) on page 37.

2. Building or selecting an enveloping model.

In this step, you define data for interchange and group enveloping. This is described in [Enveloping Models](#) on page 55.

3. Defining the script.

In this step, you pair message/set models with enveloping models. This is described in [Assembling the Script](#) on page 65.

4. Creating data.

In this step, you actually generate the data. This is described in [Generating Data from a Script](#) on page 68.

In steps 1-3, you can either:

- Use an old model or script as is.
- Open an old model or script and change it.
- Open an old model or script, save it to a new name, and change it.

- Create a new model or script.

The steps are identical for all standards (X12, UCS, MOTOR, WINS, TRADACOMS, and EDIFACT). Where important differences exist, the differences are noted.

Message/Set Models

A message/set model is a customized version of a transaction set or message. It can be straight from the published standard or (more likely) based on a guideline or MIG that has been customized in Standards Editor.

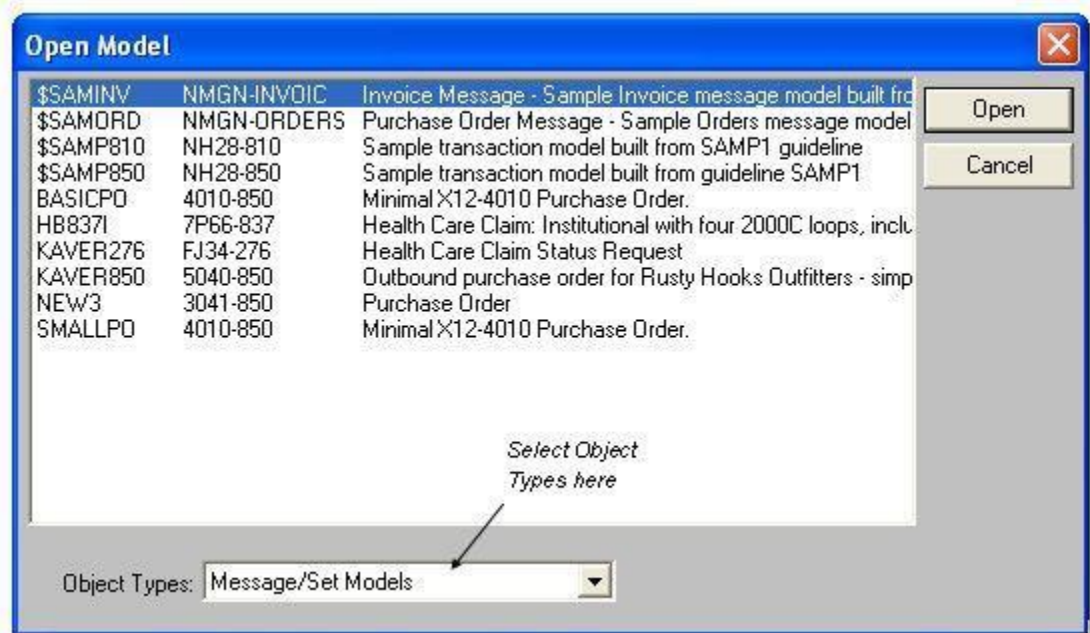
You can set it up to create the data that you have agreed on with your trading partner(s), or it can incorporate errors or variations that are likely to occur.

To Create or to Reuse a Message/Set Model?

If you **do not** have a similar message/set model, you will create a new one.

If you already have a similar message/set model based on that standard, you can reuse it as is or by saving it to a new name. This would be a good choice if you customized the model a lot, and you want to use those changes. By saving a message/set model under a new name, you can create your new one without affecting the original. This is explained in more detail in [Save or Save As](#) on page 73.

To decide which approach is best, choose **File | Open | Object Types | Message/Set Models**. You will see the names of all message/set models.



Which ones are similar to what you want? The names and descriptions are one clue. If you suspect that one might be close to what you need, highlight it and click **Open**.

Does it seem like a close match? If so:

1. Choose **File | Save As** and save it to a new name (up to 8 characters).
2. Customize the description so that, later, you will know the purpose of this model and how it differs from similar ones.
3. Choose **OK**.

If you look at the title, you'll see that you are now editing the copy. You can customize it without affecting the original.

If no existing model is close, create a new model from the appropriate EDI standard, guideline, or MIG. Choose **File | New | New Message/Set Model**, choose the standard, guideline or MIG, then the transaction set or message, then click **OK**. Choose **File | Save As** or use the diskette button on the toolbar, and fill out the name and description.

Regardless of whether you create a new model or copy an existing one, you will customize it by:

1. Disabling objects that should not be included in the data.
2. Changing the repeat count for objects.
3. Inserting, replacing, and moving objects.
4. Customizing values.

Getting to the Message/Set Model Screen

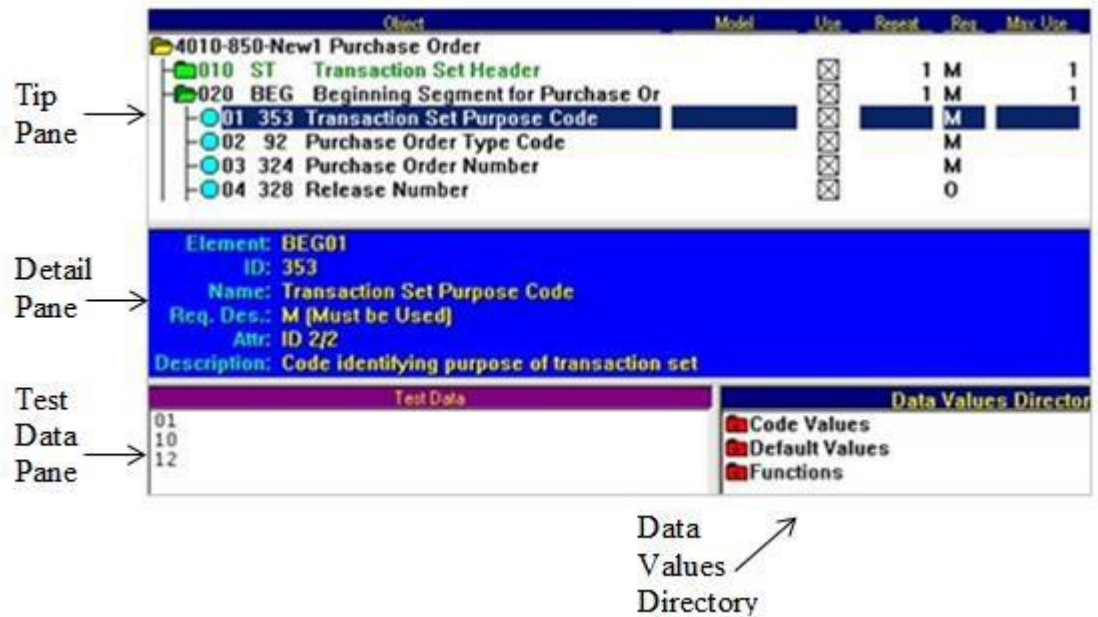
To open an existing message/set model:

File | Open | Object Types | Message/Set Models |
choose model | **Open**

To build a new one:

File | New | New Message/Set Model | *choose standard, guideline, or MIG*
| *choose tran. set or message* | **OK**.

You will see the four-pane message/set model screen, where you can customize the transaction set or message.



Top Pane: the Top of the Message/Set Model Screen

The top pane is the navigator. It shows all the objects in the standard, guideline, or MIG, in order, from ST to SE (for X12) or UNH to UNT (for EDIFACT). When you highlight an object in the top pane, the other panes update to show information about it.

Expanding and Collapsing Objects in the Top Pane

When you first enter the message/set screen, you will see a list of segments and loops or groups. No subordinate information is displayed. You can expand the display by one level or all the way down. A closed folder precedes expandable objects.

To expand one level, use any of these methods:

- Highlight the object and press *Enter*.
- Double-click on the object.
- Highlight the object that you want to expand and choose **Tree | Expand One Level**. On a loop, this would display all segments in the loop. On a segment, this would display all elements in the segment.

To expand a branch (all levels of subordinates):

- Highlight the object and press *Shift+Enter*.
- Hold down the Shift key and double-click on the object.
- Or, highlight the object and choose **Tree | Expand Branch**.

On a loop, this would display all segments and elements in the loop. On the top line, this would expand the entire transaction set or message.

To collapse an object's subordinates:

- Highlight the object and press *Enter*.
- Double-click on the object.
- By menu, highlight the object and choose **Tree | Collapse Branch**.

Information in the Top Pane

This information initially is inherited from the standard, guideline, or MIG. You can override the Use and Repeat settings. The columns are:

Object

The position number, ID, and name of the transaction set, message, loop, group, segment, composite, or element. For details about the icons that precede the objects, see [Your Complete Guide to Colors and Symbols](#) on page 24.

Model

The name of the external model, if one is attached. These are customized versions of the segment, loop or group, composite, or element. They have been saved with their own name, so that they can be used in multiple places. See [External Models](#) on page 89 for more information. This column is blank unless an external model is being used.

Use (can be edited right on the screen)

Determines whether data should be generated for this object. You can select or clear the check box with a mouse click or with the space bar. If the box is empty, no data will appear in the file for this object. See [Changing whether an Object is Used](#).

Repeat (can be edited right on the screen)

The actual number of times this segment or loop will appear in the data file at this position. See [Changing the Repeat Count](#) below.

Req

The underlying standard's requirement for this object (mandatory, optional, conditional, etc.). To see the user-defined requirement, look in the blue box. It will be in parentheses after the Req Des, if it exists.

Max Use

The maximum number of times this segment, loop, or group may appear here, according to the underlying standard. >1 means the segment, loop, or group may be repeated any number of times.

Type

(elements only) The data type for this element, according to the standard. Please see [DataTypes.pdf](#) in EDISIM's Documentation directory for details.

ForX12, UCS/WINS, TDCC:

AN Alphanumeric

N#	Implied decimal (# places right of implied decimal point)
R	Decimal (explicit decimal point required)
ID	Code value from list
TM	Time
DT	Date
B	Binary

For EDIFACT:

A	Alphabetic characters
N	Numeric characters (can include decimal point)
AN	Alphanumeric characters
ID	Code value from list

Min

(elements only) The minimum length for the element's data.

Max

(elements only) The maximum length for the element's data.

You can change Model, Use, and Repeat from within the top pane. To change the contents of other columns, use Standards Editor to change the guideline or MIG on which this model is based.

Rep

(composites and elements only) The maximum number of times the element or composite can repeat at this location. Repeating elements are available in X12-4030 and later. For an example, see set 810, DMG segment, C056 composite. EDIFACT versions D99A and later also allow repeating elements, although none actually appear in these versions.

The data will include the full number of repetitions. For example, if the guideline, MIG, or standard has a repeat count of 10 for a composite, TDG will create 10 repetitions in the data. The element repeat count cannot be changed in TDG. To create data with fewer repetitions, edit the guideline or MIG in Standards Editor and change the repeat count. Then re-open TDG and regenerate the data from the existing model.

What Does Shading and Fading Mean?

Objects that have a shaded background are part of an external model. You should see its name in the Model column at the top of the shaded area. This is fully explained under [External Models](#) on page 89.

Objects that have faded text are specifically set to unused, or are part of something that is unused. For example, a segment that is part of an unused loop will have faded text. If, in addition, it has no X in the Use column, it has specifically been marked as not used.

Whether there is an X or not, an object that is faded will not contain EDI data in the file created by TDG. See [Changing whether an Object is used](#) below.

These three shaded lines are all part of an external model named PACKAGE.

Elements 347 and 01 are faded because they are unused. Look at their Use column.

Object	Model	Use	R
0200 BSN Beginning Segmen		<input checked="" type="checkbox"/>	
0400 DTM Date/Time Referen		<input checked="" type="checkbox"/>	
0100 HL Hierarchical Level	PACKAGE	<input checked="" type="checkbox"/>	
0100 HL Hierarchical Lev		<input checked="" type="checkbox"/>	
1900 MAN Marks and Numb		<input checked="" type="checkbox"/>	
0100 CTT Transaction Totals		<input checked="" type="checkbox"/>	
01 354 Number of Line Item		<input type="checkbox"/>	
02 347 Hash Total		<input type="checkbox"/>	
03 81 Weight		<input checked="" type="checkbox"/>	
04 355 Unit or Basis for Me		<input checked="" type="checkbox"/>	

Changing whether an Object is used

If your model is based on a guideline or MIG rather than a published standard, some optional or conditional objects may have been marked as not used in Standards Editor. Such objects will appear faded in the TDG top pane, and their Use column check box will be cleared. This means that the test data will contain no data for this object.

To toggle an object's Use off or on, click on the check box in the Use column in the top pane, or highlight the object and press the *Space Bar*. This also sets its Repeat column to zero. Likewise, changing an object's repeat to zero automatically marks it as unused (see [Changing the Repeat Count](#) below).

You can clear the **Use** check box for an object that is mandatory, but TDG warns you.

Be aware of syntax rules or dependency notes between elements. These appear in the blue detail pane.

Changing the Repeat Count

The Repeat column in the top pane shows how many times the object will appear at this location in the data. (Max Use is the maximum number of times it is *allowed* to appear there.) You can change repeat count for segments, loops (X12), or groups (EDIFACT). Composites and elements do not have repeat counts. Click on the number in the Repeat column, type the new repeat count, and press *Enter*.

Easy keyboard tricks: When highlighting a segment, loop, or group in the top pane, you can press the keyboard's plus (+) key to increase the repeat count by 1. The minus key decreases it by 1. If you decrease it to zero, the object automatically becomes unused.

Element Repeat Counts

EDIFACT D99A and later can contain elements and composites that repeat more than once. This information appears in the **Rep** column at the far right in the top pane. It can be changed in Standards Editor but not in TDG.


Object	Model	Use	Repeat	Req	Max Use	Type	Min	Max	Rep
220 CTA Group 5: Contact		<input checked="" type="checkbox"/>	1	C	5				
0230 CTA Contact Inform		<input checked="" type="checkbox"/>	1	M	1				
0240 COM Communicatio		<input checked="" type="checkbox"/>	1	C	5				
01 C076 Communicatio		<input checked="" type="checkbox"/>		M					3
01 3148 Communicat		<input checked="" type="checkbox"/>		M		AN	1	512	
02 3155 Communicat		<input checked="" type="checkbox"/>		M		AN	1	3	
0 TAX Group 6: Duty/Tax		<input checked="" type="checkbox"/>	1	C	5				

The data generated by each repetition of the loop, group, segment, composite, or element may not be identical, since TDG will cycle through the value lists.

Inserting a Segment, Loop, or Group

You can insert segments, loops, or groups into your message/set model.

1. In the top pane of the message/set model, highlight the line above or below where you want something inserted.

2. Choose **Edit | Insert** or click the Insert button on the toolbar: .

3. In the Select item box, click on the **Types** line and choose the type of object you want to insert:

Set xxx or **Message xxx** lists all segments in the transaction set or message.

Segment Dictionary lists all segments in the dictionary.

Segment Models lists external segment models for all segments (not just the current one) known to TDG. See [External Models](#) on page 89.

Loop Models or **Group Models** lists external loop models for all loops (not just the current one) known to TDG. See [External Models](#) on page 89.


4. Highlight the object(s) that you would like to insert. To select multiple objects, use *Ctrl*+click.
5. Click the **Above** or **Below** radio button to choose whether insertion goes above or below the line highlighted in the top pane.
6. Click **OK**.

You cannot insert an element or composite into a segment. If you really need to do this, create a new segment in the Standards Editor dictionary for the guideline or MIG, then return to TDG and replace the segment with the new one from the segment dictionary.

Moving a Segment, Loop, or Group

Moving a line in the top pane consists of inserting it in the new location and then deleting it from the old location.

1. In the top pane, note the position number of the object that you wish to move.


2. Highlight the line above or below where you want the object to appear (while inserting, you have the choice to insert above or below the current line).
3. Choose **Edit | Insert** or press the insert button: .
4. Choose **Types | Message** (if you are using EDIFACT) or **Types | Set** (if you are using X12).
5. Click on the segment, loop, or group that you want to move.
6. Select **Above** or **Below**.
7. Click **OK**.

Next, delete the object from its previous location as described in [Deleting a Segment, Loop, or Group](#) below.

Elements and composites cannot be moved in the top pane.

Deleting a Segment, Loop, or Group

As an alternative to changing the Use column, you can delete a segment, group, or loop entirely.

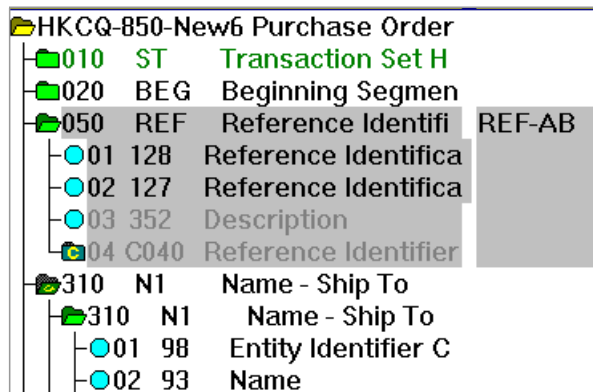
1. Highlight the segment, loop header, or group header. If you would like to delete multiple objects, use *Ctrl*+click to select them all.
2. Choose **Edit | Delete** or press the Delete button: .

TDG will confirm before deleting.

The following objects cannot be deleted directly from the top pane:

Objects that are part of an external model will have a gray background. You cannot delete *part* of an external model. You can, however, delete the *entire* model from the top pane by highlighting the segment, loop, or group that has the model name in its Model column, then choosing **Edit | Delete**. This only deletes the model from the *current location*. The model continues to exist and perhaps to be used elsewhere. To modify an external model, see [External Models](#) on page 89.

Elements and composites cannot be deleted. They must remain part of the segment because they act as placeholders, even if they are not used. To omit them from the test data file, click on their Use box to remove the X. For details, see [Changing whether an Object is Used](#) on page 42.



In the picture above:

The REF segment can be deleted. The entire segment, which is an external model, would be removed from the transaction set.

Element 128 cannot be deleted for two reasons: it is an element, and it is part of an external model.

The N1 loop can be deleted. The entire N1 loop would be deleted.

The N1 segment can be deleted. The only purpose to such a deletion would be to test an error condition, since this N1 segment is the loop trigger for the N1 loop.

Loops and Groups: Deleting the loop header will delete the entire group. You cannot select the header and a few segments for deletion.


To delete a loop, but keep one or more of its segments:

1. Delete the loop.
2. Use **Edit | Insert | Types | Set** or **Message** to insert the segments where you want them to go. See [Inserting a Segment, Loop, or Group](#) on page 43.

Replacing an Object

You can replace the currently-highlighted loop, group, segment, composite, or element in the top pane with an alternative definition of this object. Example: if you are on a NAD group header, you can choose from a list of other NAD groups known to TDG. If you are highlighting an N2 segment, you can choose from a list of other N2 segments known to TDG. This process is similar to inserting an object, except that you can only choose from similar objects.

1. In the top pane of the message/set model, highlight the object that you want to replace.

2. Choose **Edit | Replace** or click the replace button: 

3. In the Select item box, click on **Types** and choose the type of object that will replace the current one:

Set xxx or Message xxx lists all instances of this object in the transaction set or message.

xxx **Models** lists all external models for the current object.

4. Highlight the object that you would like to use and click **OK**.

Objects with gray backgrounds are part of an external model. To replace an object that is part of an external model, you have to edit the model itself. However, you can replace the *entire* external model by highlighting the line with the model's name in the Model column and then choosing **Edit | Replace**. See [External Models](#) on page 89.

A Word about Loops and Groups

You can recognize a loop (X12) or group (EDIFACT) by its dark green folder icon with the circular yellow arrow.

Since a loop or group is simply a collection of segments and perhaps other loops or groups, you customize it the same way you do any other object. These activities are described in the previous sections.

If you clear the Use check box for the header line of a loop or group, its segments and any nested loops are not included in the data. They all become faded.

If you leave the Use check box selected for a loop or group header line, you can set the Use check box for each of its segments and elements individually.

If you delete a loop or group header line, the entire loop or group will be deleted. For details, see [Deleting a Segment, Loop, or Group](#) on page 44.

A Word about Composites

Composites are groups of elements that are used together. You can recognize a composite by its blue-green folder icon containing a yellow **C**. Composite IDs start with **C** or **S**.

A composite has no Type, Min, or Max, since values are assigned to the subelements themselves, not directly to the composite.

EDIFACT standards contain many composites, and composites also appear in X12 versions beginning with 3030 (look in the MEA segment for an example). To expand a composite, double-click on it.

An element in a composite is called a subelement, and it has a position number within the composite.

If you clear the Use column check box for the composite itself, its subelements will be faded and omitted from the test data. Their Use check boxes will not change, however. If Use is selected for the composite itself, TDG looks at each of its subelements to see if they are to be used.

Detail Pane: the Middle of the Message/Set Model Screen

The middle (usually blue) pane shows all details about what you have selected in the top pane, with a scroll bar on the right if information is off the screen.

The Two Data Panes: The Bottom of the Message/Set Model Screen

The two panes at the bottom are active when an element is selected in the top pane.

The Test Data Pane at the bottom left contains values that will actually be used for this element in the test data. You can change this list of values. TDG steps through the list, using the next value each time the element is used. If it runs out of values when generating data, it returns to the top of the list. If you want the element to always have the same value, include only one value in the Test Data pane. For more detail, see [How Values Lists Cycle](#) on page 92. If this pane is empty, or becomes empty, TDG will use default values, which you can see by double-clicking on the Default Values folder to the right.

The Data Values Directory at the bottom right provides a convenient list of values that you might want to consider using for this element.



You can copy default values, functions, code values, and application values from the data values directory to the Test Data pane, as follows:

1. In the data values directory, double-click on a folder to expose its contents.
2. Use a mouse click to select the value that you want to use. For multiple values, use *Ctrl*+click.

3. Drag the folder or one of the highlighted values over to the Test Data pane. All selected values will go.

Shortcut: to move one value from the data values directory to the Test Data pane, double-click on it.

If you drag a folder itself, and if none of its contents are highlighted, everything in the folder goes to the Test Data pane.

Default Values

TDG has preloaded default data for this element. It can be functions, code values, application values, or other values. Default data conforms to the length and type needed for this element.

Use defaults whenever you *do not care* what values the element takes in the actual test data produced. For other elements, you can specify the values.

TDG gives you a visual cue when an element's values have been changed from the defaults: in the top pane, the element's blue circle icon will contain a black dot.

To restore the default values, delete the contents of the Test Data pane.

Functions Folder

The Functions folder lists every function in TDG. A function lets you automatically supply a value or part of a value. It is a powerful tool for those who need it.

You can expose the list of functions by double-clicking on the Functions folder then dragging one to the Test Data pane. While generating the test data, TDG interprets functions and uses the values they yield.

Functions are enclosed in braces { }. Examples:

{DATE+1}	Tomorrow's date (format: <i>yymmdd</i> or <i>yyyymmdd</i>).
{VALUE (ISA06) }	Sender ID from the ISA segment.
{SEGCOUNT (PO1) }	Number of PO1 segments in this message/set model up to this point.

When building data, TDG warns you if you have entered a function with the wrong syntax.

TDG does not actually compute the function until generation time, so be sure the resulting value will not violate the length or type of the element. Typing a specific value is generally preferred if your intention is to force an error.

Instead of dragging a function, you can type it just like a literal. Example:

{DATE}

puts the *yymmdd* or *yyyymmdd* system date (depending on element maximum length) into the test data for this element.

PRODNUM{RAND(10000,99999)}

concatenates the literal PRODNUM with a random number in the range shown:
PRODNUMxxxxxx where xxxxxx is a random number between 10000 and 99999.

A typical function, shown in the figure below, will generate random dollar amounts between 1.00 and 10.00 (always with zero cents). Notice that the value list also contains a literal of **100.00**, which will alternate with the random value.

02	187	Weight Qualifier
03	60	Freight Rate
04	122	Rate/Value Qual
Element: L303		
ID: 60		
Name: Freight Rate		
Test Data		
100.00		
{RAND(1,10)}.00		

For a list of functions, see [Complete List of Functions](#) on page 80.

Code Values

The Code Values folder contains the official list of acceptable values for this element in this segment. They come from the underlying standard, but it is possible for a Standards Editor user to modify the list. For X12, elements with code values usually have **ID** for their type.

Local codes, which are not in the underlying standard, were added by a Standards Editor user when developing a guideline or MIG. They contain a yellow **L** in the red page icon.

Faded codes have been marked as unused for this location (this is done in Standards Editor or by industry groups such as UCS and VICS).

Other codes are valid at this location, according to the guideline, MIG, or industry subset.

You can drag one or more codes to the Test Data pane, even if they are faded. As an alternative, you can click in the Test Data list, then use **Edit | Insert** and type the code value in capital letters.

To see the description for a code in the Test Data list, highlight it and look at the status bar on the bottom of the pane.

Application Values

(Guidelines and MIGs only) This is a list of values that a Standards Editor user has specified as valid for this element at this location. These values are not part of the underlying standard. You can drag one or more application values from their folder to the Test Data pane.

Typing Values

In addition to dragging values from the data values directory to the Test Data pane, you can simply type them into the Test Data pane.

For example, you might place these literal values in element 330-Quantity Ordered:

3.4

3.6

2.0

You might give these literal values to element 93-Name:

Rita O'Neill

Edward Wilson

Daniel Brown


Generally, literal values can include any characters on your keyboard except for braces { }, which are reserved for functions, and special control keys like *Enter* and *Esc*.

If you type leading or trailing blanks, they are retained. For fixed-length fields (for example, in the ISA segment), you can enter trailing blanks to the desired length by watching the Length indicator.

To enter a literal value, click in the Test Data pane and choose **Edit | Insert** to get to the Data Value box. TDG will report any violations of element type or length. Correct any reported violations unless you wish to force an error. For details, see [Using the Data Value Box](#) on page 52.

Copying Values from the Clipboard

To use values that are already stored elsewhere on your computer:

1. From the software that currently contains the data: copy the data into the clipboard. In most products, you can do this with **Edit | Copy** or *Ctrl+C*.
2. From within TDG, open the model and highlight the element that is to use the values.
3. Click in the Test Data pane and use **Edit | Paste** or the toolbar button that looks like a clipboard: 

The values are inserted above the current line.

Null Values

To enter a null value, use the **{EMPTY}** function. You can either:

- Type **{EMPTY}** as a literal, as described in [Typing Values](#) above.
- Drag the **{EMPTY}** function from the Functions folder in the data values directory to the Test Data list.

Dragging Data to the Test Data List

You can drag individual values or groups of values to the Test Data pane so that they will be used instead of TDG's default data.

Drag all code values

You can copy the entire contents of the Code Values folder to the Test Data list. If the folder is expanded (its contents are showing), be sure that no individual values within the folder are selected. Click on the folder and drag it to the Test Data pane. Each code value will appear in the Test Data pane.

Drag one value

Double-click on the folder that contains the values, so that its contents appear. Click and drag one value to the Test Data pane.

Drag selected values

Double-click on the folder that contains the values, so that its contents appear.

Ctrl+click to select individual codes.

Shift + click to select a range of codes.

When all codes that you want are highlighted, drag one of the highlighted codes to the Test Data pane.

When dragging, the mouse cursor includes an icon that looks like a piece of paper. This icon will have a circle with a line through it when you are pointing to a place where you cannot drop the values.

You can drop values:



You cannot drop values:



If you decide in mid-drag that you don't want to use the codes, just drag to where the paper icon has the circle and then drop.

If the Test Data pane already contains values, any values that you drag to it will go to the end of the list. You can then reorganize them as detailed in [Inserting, Deleting, or Moving Test Data Value](#).

Editing Values in the Test Data List

You may need to change an entry in the Test Data list. For example, you will need to include the min and max if you use the RAND function, or you may wish to do calculations that involve a function. To edit something in this list:

- Double-click on it
- Or highlight it and press *Enter*

to reach a Data Value box where you can make changes. This box is explained in [Using the Data Value Box](#) on page 52.

Inserting, Deleting, or Moving Test Data Values

You can use the Edit menu, or toolbar buttons to help you edit the Test Data list. These are active when you click in the Test Data pane.



Edit | Insert to insert a value before the currently-selected one



Edit | Replace or double-click on the object to replace the currently selected value.



Edit | Delete or press *Delete* key to delete the current values. Before deleting, you can select multiple values using *Ctrl+click*, *Shift+click*, *Shift+End*, or *Shift+Home*.



Edit | Move Up to move the currently selected value up.



Edit | Move Down to move the currently selected value down.




Edit | Paste or *Ctrl+V* to paste text (after using copy or *Ctrl+C* from another Windows application or another Test Data list)

Using the Data Value Box


You will see the Data Value box if you insert or replace an object in the Test Data list, or if you double click on a value in the Test Data list.

The Data Value dialog box has a blue title bar with the text 'Data Value' and a close button. It contains several fields and buttons. The 'Element' field shows '01003 127' and 'Reference Identification'. The 'Type' field shows 'AN', 'Minimum' shows '1', and 'Maximum' shows '30'. Below these is a row of checkboxes labeled '20', '30', '40', and '50'. A text input field contains the text 'DU'. At the bottom, it says 'Current Column: 15' and 'Length: 2'. On the right side, there are three buttons: 'OK', 'Cancel', and 'Functions »'.

Inserting a value using the Data Value box:

1. Click on a value in the Test Data list. The insertion will go above this line.
2. Press the Insert toolbar button , or choose **Edit | Insert**.
3. In the Data Value box, type the literal value that you would like to insert into the list of test data values. The *Home* and *End* keyboard keys move to the beginning or end of the value.

4. Click **Add** or press *Enter* to insert the value. If the value violates the element's type or length, you will see a warning. However, TDG lets you add an invalid value so that you can test errors.
5. Continue typing values and clicking Add until you have inserted all the values that you want.
6. To close the box, click **Cancel**.

To move the newly inserted value up or down, use **Edit | Move Up** or **Edit | Move Down**, or the corresponding toolbar buttons: 

Replacing a value using the Data Value box:

1. Double-click on the value that you wish to replace in the Test Data list. (The toolbar Replace button or **Edit | Replace** will also bring up this box if you have already highlighted the value).
2. In the Data Value box, type the literal value that you would like to insert into the list of test data values. The *Home* and *End* keyboard keys move to the beginning or end of the value.
3. Click **OK** or press *Enter*. TDG warns you if the value violates the element's type or length.

To enter a function using this box:

1. Click on a value in the Test Data list. The function will be inserted above this item.
2. Press the Insert toolbar button, or choose **Edit | Insert**.
3. In the Data Value box, press the **Functions** button.
4. Click on a function to read about it.
5. Double-click on a function to copy it to the editing line.
6. If necessary, make editing changes to the function. Functions with lowercase text must be edited. For instance, if you choose the **{RAND(*min*,*max*)}** function, replace *min* and *max* with numbers.
7. Choose **Add** or press *Enter*.
8. Choose **Cancel** to close the box.

Please see [Complete List of Functions](#) on page 80 for details about each function.

Length and position indicators in the Data Value box:

- Ruler.
- *Current Column* at the bottom left shows the current location of your text editing cursor.

- *Length* at the bottom right shows the current length of your value.

If you cannot Make Changes in the Data Panes

If you cannot change the Test Data pane, the element highlighted in the top pane is part of an external model. Both the data values directory and the Test Data pane will have shaded backgrounds if this is the case. To make changes in such a situation, please see [Editing an External Model](#) on page 91.

Deleting a Message/Set Model


1. With the model closed, choose **File | Delete Models**.
2. Select the Type from the bottom of the box.
3. Highlight the one that you wish to delete. To delete multiple models, use *Ctrl*+click.
4. Click **Delete**.

Exporting a Message/Set Model

1. Open the message/set model.
2. Save any changes.
3. Choose **File | Export**.
4. In the file box, enter the filename and path.
5. Choose **OK**.


Saving a Message/Set Model

When finished making changes to your message/set model, save your changes, using either:

File | Save or the  button on the toolbar: saves the model to the current name. If this is a new model, you will see the Save Model As. box just as if you used File | Save As.

File | Save As if it is a new model, or a previously-saved model that you wish to save to a different name. You will see the Save Model As... box. Enter a name (up to 8 characters) and description.

Exiting the Transaction Set Model Screen

Choose **File | Close** or click the  button on the toolbar.

TDG prompts if you have unsaved changes.

If you have an enveloping model that you want to use with this transaction set or message, you can now create a script and generate test data.

What if I Later Change the Guideline or MIG in Standards Editor?

If you change the usage of an object in Standards Editor, will its models reflect the change?

- | | |
|------------|--------------------------|
| No | Loop/Group, segment |
| Yes | Composite, element, code |

Enveloping Models

If the message/set model answers the question of *what* to simulate, you could say the enveloping model provides the *who*. It defines the interchange and group level envelopes, along with information on how enveloping and control numbers and delimiters are to be used. In the enveloping model, you:

- Choose the type of enveloping (EDIFACT, X12, UCS, etc.).
- View or change the control numbers (or counters).
- See or change control segments.
- Set the enveloping method (full, partial, simple).
- Set the delimiters.

Each enveloping model should describe a unique test case, representing one or more trading partners.

To Create or to Reuse an Enveloping Model?

If a suitable enveloping model exists, simply insert it into your script as described in [Adding Rows to your Script](#) on page 66.

If no suitable enveloping model exists, it is easiest to open and then Save As an existing one that is similar to the one you need.

If no similar one exists, create a new one.

To decide which approach is best, look at your existing enveloping models by choosing **File | Open | Object Types | Enveloping Models**.

Do you see one that might be similar? If so, double-click on it and view its structure. If it is a close match, you can use **File | Save As** and save it to a new name. Look at the title to confirm what you are now editing. You can then make any changes that you'd like without affecting the original.

If you have to create a new enveloping model instead, select **File | New | New Enveloping Model**. You then automatically go to the Envelope Model screen.

Getting to the Enveloping Model Screen

You have two ways to visit the Envelope Model screen:

- **File | New | New Enveloping Model** brings up the Envelope Model screen, preloaded with defaults for everything except Description.
- **File | Open | Object Types | Enveloping Models** lists all existing enveloping models. Highlight the one that you want and choose **Open**.

You will see the Enveloping Model screen. This is described in [Customizing your Enveloping Model](#) below.

Customizing your Enveloping Model

If you are creating a new enveloping model, the Envelope Model screen will be preloaded with defaults for everything except Description.

Starting at the top, work your way down all the fields, checking the settings and changing as necessary.

Description:	
Envelope Type: X12 (ASC X12 Draft Standards - ISA, IEA)	
Enveloping Method: Full (Each Set/Msg has own Func Group, Intchg Envelopes)	
Control Segments	
Interchange Control	Header: ISA - Interchange Control Header
	Functional Group Header: GS - Functional Group Header
	Functional Group Trailer: GE - Functional Group Trailer
	Trailer: IEA - Interchange Control Trailer
Counters	
Interchange:	1
Functional Group:	1
Transaction:	1
Delimiters	
Segment Terminator:	!
Element Separator:	*
Sub-Element Separator:	:
Decimal:	.
Release:	?
Rep Elem:	+

Description Field for Enveloping Model

Type in a description summarizing this enveloping model. The first 52 characters of the description will show up in the Open Model box whenever this model appears, so the description is an important way to distinguish this enveloping model from others.

Envelope Type

You can change this field only if you are creating a *new* enveloping model.

Click on this line to drop down a list of choices, then click on the one that you want to use:

X12 (ASC X12 Draft Standards - ISA, IEA)

Use for X12 data based on versions *earlier than* 3072.

Use for WINS and TDCC standards.

GS04 (element 373) length is 6 characters for date in YYMMDD format.

ISA11 is element I10 Interchange Control Standards Identifier.

To see the structure of the ISA, GS, GE, and IEA, look in the dictionary of any X12 standard or guideline based on versions 3040-3071.

X12 (Y2K) (ASC X12 Standards 3072 and later - ISA, IEA)

Can be used for X12 data based on version 3072 and later.

GS04 (element 373) length is 8 characters for date in CCYYMMDD format.

ISA11 is element I10 Interchange Control Standards Identifier.

To see the structure of the ISA, GS, GE, and IEA, look in the dictionary of any X12 standard or guideline based on version 3072 or later.

X12 (Rep Elm) (ASC X12 Standards 4020 and later - ISA, IEA)

Can be used for X12 data based on version 4020 and later.

GS04 (element 373) length is 8 characters for date in CCYYMMDD format.

ISA11 is element I65 Repetition Separator, which defines the separator character for repeating elements.

To see the structure of the ISA, GS, GE, and IEA, look in the dictionary of any X12 standard or guideline based on version 4020 or later.

EDIFACT (UN/EDIFACT - UNB, UNZ)

Can be used for all EDIFACT data *except* where the UNB complies with ISO 9735 Version 4.

To see the structure of the UNB, UNG, UNE, and UNZ, look in the dictionary of any EDIFACT standard.

EDIFACT4 (ISO 9735 Version 4 - UNB, UNZ)

Can be used for EDIFACT data based on version D99A and later, where the UNB complies with ISO 9735 Version 4.

To see the structure of the UNB, UNG, UNE, and UNZ, open UN4ICS and view CTLSET.

UCS (UCS/WINS - BG, EG)

Used for UCS retail and WINS warehousing guidelines.

To see the structure of the BG and EG, look in the segment dictionary of recent UCS and WINS standards.

TRADACOMS (TRADACOMS 93 - STX, END).

Used for guidelines based on TRAD93 standards.

To see the structure of the STX and END segments, open TRAD93 and look in the segment dictionary.

GENCOD (GENCOD - CNUT)

Used for guidelines based on the GENCOD standard.

When you choose the envelope type, TDG automatically updates the rest of the window accordingly.

For an overview, please see [Customizing your Enveloping Model](#).

Enveloping Method

Enveloping Method determines whether transaction sets or messages are to be sent as individually addressed units, or bundled in a group envelope or interchange envelope.

Click on the **Enveloping Method** line and select one of the three enveloping methods.

Full

Each transaction set or message will have its own interchange and group envelopes.

Simple

Each transaction set or message will share a common group and interchange envelope, until: (1) a subsequent enveloping model in the same script specifies *Full* or *Partial*; or (2) the functional ID of the matched transaction set or message in the script requires a new group envelope to comply with the EDI standard.

Partial

Each transaction set or message will have its own group envelope, but will share a common interchange envelope (until a subsequent enveloping model specifies *Full*).

Example Comparison of Enveloping

Assume we have to transmit two purchase orders and one invoice, using X12 enveloping.

Full	Simple	Partial
ISA	ISA	ISA
GS	GS	GS
ST	ST	ST
1st PO	1st PO	1st PO
SE	SE	SE
GE	ST	GE
IEA	2nd PO	GS
	SE	ST
(repeat all for 2nd PO)	GE	2nd PO
	GS	SE
(repeat all for Invoice)	ST	GE
	invoice	GS
	SE	ST
	GE	invoice
	IEA	SE
		GE

Control Segments

Choose which control segments to put in your enveloping. By default, these are:

	X12	EDIFACT	UCS	TRADACOMS	GENCOD
Interchange Start	ISA	UNB	BG	STX	CNUT-CNUT
Functional Group Start	GS	UNG	GS		
Functional Group End	GE	UNE	GE		
Interchange End	IEA	UNZ	EG	END	

Replacing a control segment:

Click on the arrow at the right end of the line and select the desired segment from the pull-down list.

Deleting a control segment:

To omit one of these from the test data file, click on the segment's pull-down arrow then select **None** from the top of the drop-down list. For instance, if you wish to generate data without a group envelope, set the Functional Group Header and Functional Group Trailer segments to None.

Example 1 To create a TDCC Motor transaction without an interchange envelope, set the functional group segments to GS/GE, and set the interchange segments to None.

Example 2 To create an EDIFACT message without a group envelope, use the interchange segments UNB/UNZ, and set the functional group slots to None.

EDIFACT users and UNA segment:

You cannot insert a UNA segment yourself. TDG will create one when building the data if you are using an EDIFACT envelope type and if you change delimiters from the EDIFACT defaults of:

' Seg Term
+ Elm Sep
: Sub-Elm Sep

Examining or editing an enveloping segment:

To change an enveloping segment, click on the three dots at the end of its line. You will drop down into a top pane containing the segment. Editing these segments is similar to editing a transaction set or message segment. In some control segments (notably the ISA), all elements are mandatory. In this case, there may not be much need to clear the Use check box for the elements, and most of the customization will be in the values. If


left unmodified, TDG will use default values for all generated elements, including valid control numbers.

Since elements in some interchange control segments are fixed length, default values are already blank-filled or zero-filled as appropriate to make them valid. Also, functions used to generate control numbers, such as {CTLINT}, automatically fill to the correct length.

Use defaults whenever you *do not care* what values the element takes in the actual test data produced.

For other elements, you can specify the values just as you would when customizing a transaction set or message: by using literals, functions, code values, or a combination. When you override these defaults, TDG will warn you if your entry is too short, too long, or the wrong type.

Consider using fictitious values for the sender IDs within the group and interchange envelopes so your simulations will not affect production data.

To close the enveloping segment and return to the main enveloping screen, choose **File** | **Close** or click on the  button on the toolbar.

External models of control segments:

These are included in the list that appears when you single click on the arrow to the right of an enveloping segment. Please see [External Models](#) on page 89 for details.

Counters

This area lets you manually set the counters that appear in the enveloping segments. Counters generate the control numbers that monitor successful transmissions. Each interchange, group, and transaction/message has its own set of counters.

You can leave the counters alone, since they automatically increment each time you generate data. To override or reset the counters, click on the desired counter and type the new number. TDG will provide leading zeros, if required to fill out a minimum length (as in the ISA and ST segments).

Where are counters used?

These counters automatically provide the values for the following elements through functions in their values lists:

Level	Element	Default
Interchange	ISA13, BG07, UNB05	{CTLINT}
Group	GS06, UNG05	{CTLGRP}
Trans Set	ST02, UNH01	{CTLSET}

The CTLxxx functions provide the current value of the respective counter. If you change the counters or the values lists of these elements, the test data will reflect those changes.

When are they updated?

Counters automatically increment each time you generate data. If you use the envelope a lot, the counters may someday contain values than you consider too large. You can change them in the Envelope Model screen.

For an overview, please see [Customizing your Enveloping Model](#).

Control numbers are automatically updated each time you generate data. *Example:* Imagine that the counters in this envelope are currently: Interchange 0, Group 0, Transaction 0. You now use this envelope to generate test data with this structure:

Segment	Which Counter?	Current Value in Counter
ISA	interchange	0
GS	group	0
ST	transaction	0
SE	transaction	0
ST	transaction	1
SE	transaction	1
GE	group	0
GS	group	1
ST	transaction	2
SE	transaction	2
GE	group	1
IEA	interchange	0

After generating the file, the counters increment to:

Interchange 1
Group 2
Transaction 3

Next week, you use this same envelope to generate more data. The counters will start at 1, 2, and 3. If you use the envelope a lot, the counters may someday contain values than you consider too large. You can change them in the Envelope Model screen.

Delimiters

Delimiters are the characters that you and your partner have agreed will be placed between subelements, between elements, and between segments.

Any changes you make here will override the data element separator, segment terminator, subelement separator, and repeating element separator specified in the ISA, ICS, or UNA control segments. The delimiters shown here are the ones that actually will appear in the test data.

Therefore, ISA element 16 should agree with the subelement separator to avoid error messages if you use the Analyzer with this data. ISA element 16 is the subelement delimiter. By default, its value is the function {SEDELIM}, which reads whatever you used in the main envelope window for subelement separator. However, if you replace {SEDELIM} with some other value, the subelement delimiter will not update to match changes in the main enveloping window.

Example: you leave the subelement separator at “:” on the main enveloping screen. You go to ISA element 16 and replace {SEDELIM} with a minus sign. When the test data is created, sub-elements will be delimited with “:”.

If the type of control segments you use do not provide for modifying the delimiters (BG/EG, for example), make sure the values set here correspond to industry-approved standards.

Defaults are the most commonly used delimiters:

	X12 UCS	EDIFACT
Segment	!	' (single quote)
Element	*	+
Subelement	:	:
Repeating element (EDIFACT D99A and later; X12-4030 and later - see set 810 DMG05-C056)	+	*

If you wish to change a delimiter, click on it in the main Envelope Model screen. You have two ways to change a delimiter:

- Press the key for the character to be used,
- Or, type the hex value of the delimiter. By default, these are

x1C Segment Terminator

x1D Element Separator

x1F Subelement Separator

EDIFACT Users: You can specify a different Decimal and Release character. These are not currently used by TDG, but you can enter them now so that they would be in place for the future.

Deleting an Enveloping Model

1. With the model closed, choose **File | Delete Models**.

2. Select **Types | Envelope Models** from the bottom of the box.
3. Highlight the one that you wish to delete. To delete multiple models, use *Ctrl*+click.
4. Click **Delete**.

If the deleted enveloping is used in a script, you will receive an error message when trying to use the script. You will then need to select a different enveloping model from within the script.


Exporting an Enveloping Model

1. Open the model as explained in [Getting to the Enveloping Model Screen](#) on page 56.
2. Save any changes.
3. Choose **File | Export**.
4. In the file box, select the path and filename for the export file.
5. Choose **OK**.

This export is a good backup, and can be imported by another TDG (see [Export and Import](#) on page 74).


Saving your Enveloping Model

When finished making changes to your enveloping model, save your changes, using either:

- **File | Save** or the  button on the toolbar: Saves the enveloping model to the current name. If this is a new model, you will see the Save Model As... box just as if you used **File | Save As**.
- **File | Save As** if it is a new model, or a previously-saved model that you wish to save to a different name. You will see the **Save Model As...** box. Enter a name and description.

If you exit the enveloping model when you have unsaved changes, TDG prompts about saving changes.

Exiting the Enveloping Model Screen

Choose **File | Close** or click on the  toolbar button.

You will be prompted about saving if you have unsaved changes. If you have a message/set model that you want to use with this enveloping, you can now create a script and link the two together.

Assembling the Script

Once you have a message/set model and an enveloping model, most of the work is over!

Your next step is to assemble a script, which consists of pairing up a message/set model with the enveloping you'd like to use with it. You can create one pair in a script, or many pairs. Each script will generate one test data file, regardless of the number of pairs it includes.


To Reuse or To Create a New Script?

You can create a new script, copy a script, or change an existing script. To decide which is best, first look at the existing scripts. Choose **File | Open | Object Types | Scripts**.

Do you see a script that might be similar? If so, double-click on it.


If the script is close to what you want, you can either:

- Make the changes directly to the script, then use **File | Save** to save it to the same name.
- Use **File | Save As** and save it to a new name. You can then make changes without affecting the original. Look at the title to confirm what you are now editing.

If no similar script exists, that's all right – creating a new script is easy! Use **File | New | New Script** or use the  toolbar button (see [Customizing your Script](#) on page 66).

Getting to the Script Window

You have two ways to get to the Script window:


- **File | New | New Script** brings up an empty Script window. The  toolbar button does the same thing.
- **File | Open | Object Types | Scripts** lists all existing scripts. Highlight the one that you want and choose **Open**.

Either way, you will see the Script window, and can use it as described in [Customizing your Script](#) below.

Customizing your Script

A script is a list of message/set models, each paired with an enveloping model. The entire script will generate one test data file.

Adding Rows to your Script


1. The insertion will go above the currently-highlighted line. To add to the end of the script, highlight the *End of Script* line.
2. Use **Edit | Insert** or click on the  toolbar button.
3. In the Script Line box, select a message/set model by clicking on the three dots to the right of that line.
4. Likewise, choose a corresponding enveloping model.
5. For number of repetitions, enter the number of consecutive times you would like this pair included in the file. You can change this on the main script screen also (see [Assembling the Script](#)).
6. You can type in the new functional group if you do not want to use the one that corresponds to the transaction set or message. See [Assembling the Script](#) on page 66.
7. Choose **OK**.

The screen will look like this:



You can insert as many pairs as you'd like. Insertions go above the highlighted line.

Deleting Rows in the Script

To delete a row in a script, click on the row, then choose **Edit | Delete** or click on the  button.

To delete multiple rows, use **Ctrl+click** to select them before deleting.

Replacing a Message/Set or Enveloping Model in the Script

Click on the unwanted message/set or enveloping model. Choose **Edit | Replace** or click on the  button.

You will go to the Script Line box, where you can select a different message/set model and/or a different enveloping model.

Changing the Functional Group for a Line in a Script

Why would you want to change the **Func Grp** column? The most common reason is to test your translator. What would it do in such a case?

You might want to test putting an 824 in the same functional group envelope as an 850, so you would change the functional group of the 824 to be **PO**. Transactions or messages with the same functional group, same version, and simple enveloping can go in the same functional group envelope.

To change the functional group on the main script screen, click on the one you want to change, then type the new functional group and either press *Enter* or click elsewhere.

Changing the Repeat Count for a Line in a Script

In the script screen, each line in your script starts with a column showing the repeat count. This is the number of times this pair will appear at this location in the script. To change the count, either:

- Click on the current count and type the correct number. Then click elsewhere or press *Enter*.
- Or, highlight the row and press + or - (the plus or minus keyboard keys).

The number can be between 0 and 9999999. To skip a line, set its Count to **0**.

The repeat count duplicates the entire pair, including all enveloping, in the current location before the next line starts. The test data generated for each repetition will probably be different since the value lists for each element continue to cycle.

If you want to repeat the whole script, not just one pair, use the Number of Repetitions setting in the Test Data Generation box instead (see [Generating Data from a Script](#) on page 68). The repetition count cycles through the entire script multiple times, including the interchange control header and trailer.

Saving a Script

Choose **File | Save** from the menu to save your script.

Deleting a Script

To delete a script:

1. With the script closed, choose **File | Delete Models**.
2. Select **Types | Script** from the bottom of the box.
3. Highlight the one that you wish to delete. To delete multiple scripts, use *Ctrl*+click.
4. Click **Delete**.

Exporting a Script

To export a script:

1. Open it with **File | Open | Object Types | Scripts | *<select script>* | Open**.
2. Once it is open, choose **File | Export**.
3. From the file box, enter a filename and path.
4. Choose **Save**.


Your exported file is a good backup, and can be imported into other copies of TDG. It includes all enveloping models, message/set models, and external models that are in the script.

Generating Data

Generating Data from a Script

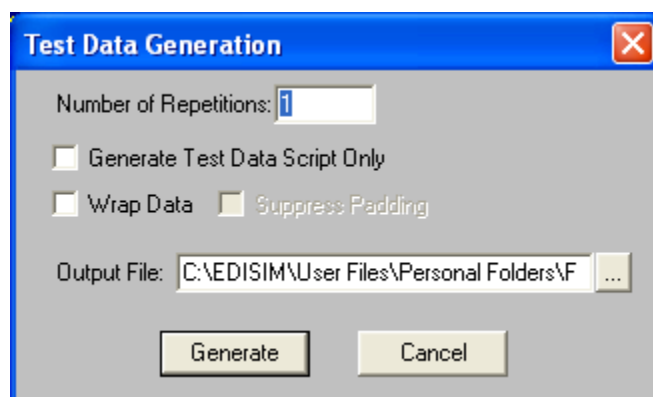
Use this method when you want to put more than one transaction set, group, or interchange in a single file.

To generate a file containing test data:

1. Open the script with **File | Open | Object Types | Scripts | *<choose the script>* | Open**.
2. Choose **File | Generate Test Data** or click on the  toolbar button.

This takes you to the Test Data Generation box, which is described below.

Using the Test Data Generation Box



Wrap Data creates test data with no CR/LF characters. It will all be one continuous line. For related information, please see [Wrapping and Folding Data with EDITWRAP](#) on page 99.

Suppress Padding is active only when Wrap Data is selected. If it is not selected, the file is padded out to 80 characters.

Output File is the path and filename for the test data. You can type a new one or click on the small dotted box at the end of the line. You will see a typical Windows file box where you can change the filename, drive, and folder.

By default, the file is called *script.TXT* (where *script* is the name of your script). You can use long file names as allowed by your operating system.

The default path is Database\<workstation>\TDG folder, where <workstation> is the name of your PC.

Be sure that you have write access to the folder, and that the drive has enough disk space for the file you will be generating.

If a file with that name exists, TDG asks if you want to proceed.

Closing the Test Data Generation Box


When you click **Generate** in the Test Data Generation box, TDG begins processing the contents of the script.

For information about what happens after you click Generate, see [While TDG Creates your Data](#) on page 69.

For details about how to quickly make data from the message/set model screen, see [Generating Data from a Message/Set Model](#) on page 69.

Generating Data from a Message/Set Model

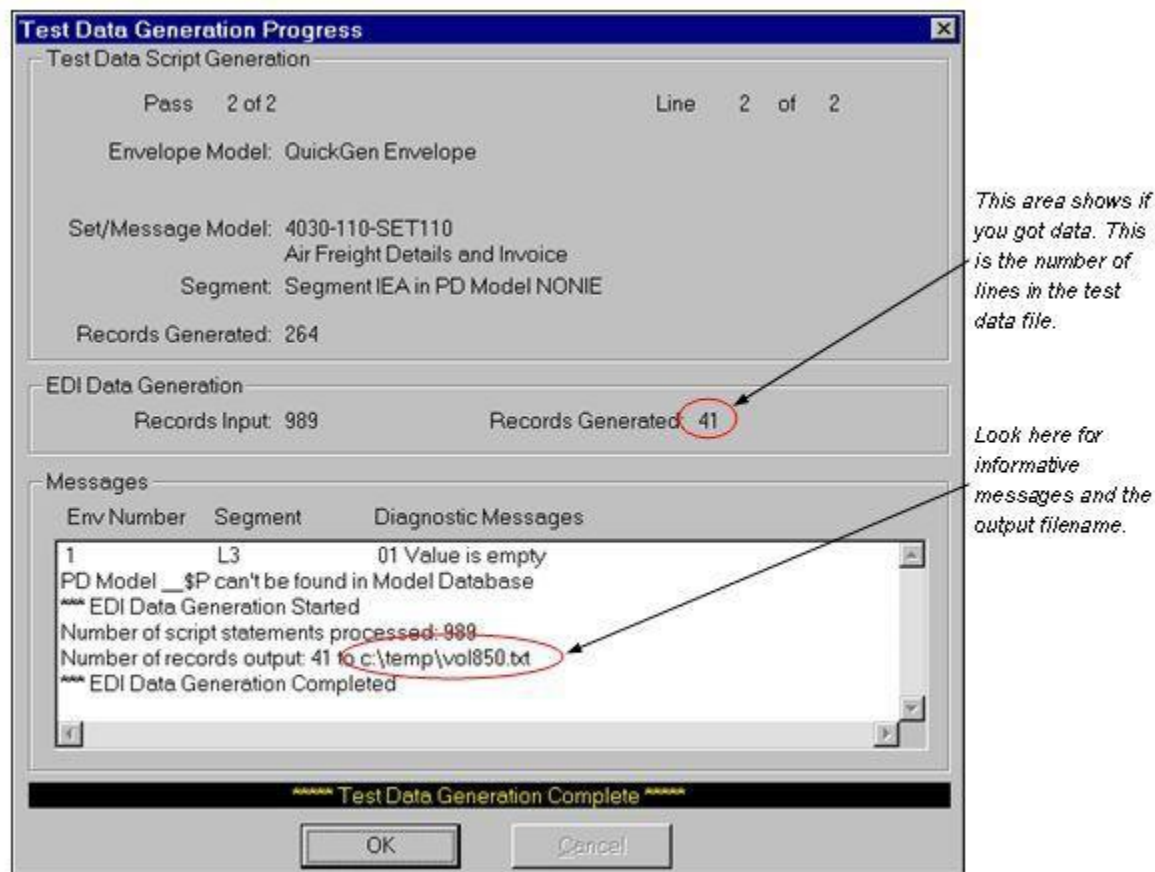
Use this method when you want to include only one transaction set in a file.

1. Open the message/set model.
2. Press the  toolbar button or choose **File | Generate Test Data**.
3. Select an enveloping model and output file name.
4. Select **Generate**.

While TDG Creates your Data

After choosing Generate in the Test Data Generation box, the screen shows progress of the script or data creation. The top shows what is currently being processed; the

Messages area shows informative messages and errors, and the very bottom line reports when the job is completed.



If you get the warning, Couldn't find EDI standard for Model xxx-xxx-xxxx, this means the underlying standard for the model is missing. If it is a published standard, you can install it from the EDISIM installation CD. If it is a guideline or MIG, you will need to restore it from a backup or export it from Standards Editor on another PC, then import it into your Standards Editor.

Diagnostics

The Diagnostic Messages list provides information about TDG's ability to make the test data. It does not check the data itself for compliance to the underlying standard (use Analyzer for that).

After viewing the diagnostics, you can decide if the errors must be corrected. Some errors prevent data from being created, and therefore have to be fixed. Look at the Records Generated line in the middle of the Test Data Generation Progress box to see if TDG created data. If this line gives a non-zero number, then data was created.

If you get the diagnostic Referenced element not checked, this is just informing you that the data in this element is being read from elsewhere (with a function) and TDG did not check it for length and type.

Once TDG has gone through the script without any unacceptable diagnostics, it saves the data to the filename shown in the Messages area. You can type or edit it, if you want to see it. There are many ways to do this, but two of them are:

- Double click on the file from Windows Explorer.
- Try a word processor and use a fixed-pitch font such as Courier. If you change the file, remember to save it as a plain text file.

For details about how to quickly make data from the message/set model screen, see [Generating Data from a Message/Set Model](#) on page 69.

6 Common Commands

Save or Save As

To save the currently-open model, which can be a:

- message/set model
- enveloping model
- script model
- external model

...you can use any of these:

File | Save Saves the model to the current name, which is displayed on the title bar. If the model has never been saved, this operates just like **File | Save As**.

File | Save As Allows you to enter a name and description. Instead of modifying a model distributed with TDG (their names start with a \$), use Save As and then customize the copy.

 toolbar button Same as **File | Save**.

If you are using Save As, or if the model is new and has never been saved before, provide a name and a description. Enter a name for the model. You can override the name and description shown, if you want to create a new model from an existing one. The description is important because it appears in lists, where it helps you see the purpose of the model.

If a model of the same kind already has the name, you will be asked whether to overwrite. If you do not, you will have a chance to enter a new name.

Use of **File | Save** is not actually required to preserve your work, since, when necessary, TDG will ask whether to save changes. However, we recommend strongly that you save frequently.

Copying a Model

To copy a model, open it and then use **File | Save As** to save it under another name.

Export and Import

TDG lets you export all or selected models to an external file for import into a TDG database. This is one means of sharing models between people working on the same project. This same export file can serve as a backup. For more information, please see [Backing Up and Restoring Your Work](#).

The three main methods of exporting and importing are:

- **From within TDG.** Use **File | Export** to get an external copy of the currently opened message/set model, enveloping model, script, or external model. Use **File | Import** to import a TDG export file. This is described in [Exporting and Importing from within TDG](#) below.
- **From within TDG.** Use **File | Export All** if you do not currently have a message/set model, enveloping model, script, or external model open. This will export all models at once.
- **From the command line.** To export or import single or multiple models, use the **FSdosutl** utility from the command line. This is described in [Exporting and Importing from the Command Line](#) below.

Exporting and Importing from within TDG

To export one model at a time from within TDG:

1. Start TDG and open the model or script that you wish to export by using **File | Open | Object Types | <select object type> | <select model> | Open**. If it is an external model, you need to find it in use somewhere, then double-click on its name in the top pane's Model column.
2. Look at the title bar to confirm what will be exported. In many cases, it will be the whole message/set model, enveloping model, or script. In other cases it could be an external model of a segment, group, loop, or element. Everything needed by the model will be exported. For example, if you export a script, all message/set models, enveloping models, and external models in the script will be included in the export.

3. Choose **File | Export**. Choose a drive, folder, and filename for the export. You can use long file names as allowed by your operating system.

To import one TDG export file:

1. Have the exported file where the importing PC can access it.
2. Close any models that might be in the import file.
3. Choose **File | Import**.
4. Choose the file containing the export.

Exporting and Importing from the Command Line

All of the following commands should be issued from the command line in EDISIM's Bin folder.

Exporting models from the command line

You may wish to export models to give to another employee, to install on a new PC, or to import into a new version of EDISIM. From the command line, you can export all models or one script. You cannot do other types of wildcard exports.

1. Close all models in TDG.
2. Go to the DOS command line in EDISIM's Bin folder. If you are using a workstation (network user) installation, go to the Bin folder on your local PC.
3. Type:

```
fsdosutil -m -Efilename [-escriptname]
```

Where:

-m	The -m (in lower case) indicates models. If omitted, the export will contain guidelines or MIGs from Standards Editor.
-E	The -E (in uppercase) specifies the DOS filename for the export, with a path if desired. Do not insert a space after the E.
<i>filename</i>	The file that is to contain the export, preceded with a path if you don't want it to go into the Bin folder.
-e <i>scriptname</i>	Optional. The e must be lower case and immediately followed by the name of a script to export. Do not type the brackets. If omitted, all models (including samples) are exported.

Examples:

To export all models to file c:\Temp\Allmodel.exp:

```
fsdosutl -m -Ec:\temp\allmodel.exp
```

To export script **newpo** to file c:\Temp\Newpo.exp:

```
fsdosutl -m -Ec:\temp\newpo.exp -enewpo
```

The export file contains not only a backup image of the **newpo** script, but all set and enveloping models it contains, along with any external models (for segment, loop, composite, and element) used anywhere in the script. Default Element models that might be used by the script when it is generated will not be exported.

Importing models from the command line

To import models previously exported from TDG, use FSdosutl from the command line in the local Bin folder of the receiving EDISIM:

```
fsdosutl -m -Ifilename
```

-m The -m (in lower case) indicates models. If omitted, the export will be assumed to contain guidelines or MIGs from Standards Editor.

-I The -I (capitalized) indicates import. Do not insert a space after the I.

filename *filename* is the name of a file that was exported with FSdosutl or with **File | Export** from within TDG. It can contain one or many models. Include a path if the file does not reside in the current Bin folder.

Example: To import models from c:\Temp\Allmodel.exp:

```
fsdosutl -m -Ic:\temp\allmodel.exp
```

Export File Compatibility

All export files produced by TDG or FSdosutl are upwardly compatible. They can be imported using the same or newer versions of TDG.

Print

For more extensive printing options, see the Document Builder section of the EDISIM Reference Manual.

Print will route a formatted copy of an open message/set or enveloping model to the printer or to a DOS file. The printout includes all associated segments, elements, values, external models, etc. This can be handy for documenting your testing program.

File | Print | Print to Printer sends the currently opened message/set model or enveloping model to your default printer.

File | Print | Print to File creates a text report of the currently opened message/set model or enveloping model. You will go to a typical Windows file box where you can select a name and folder. Default file type will be PRN. If you open this file in a word processor, you might consider using very small left and right margins, or use landscape orientation, since it is a wide report.

7 Functions

What are Functions?

Functions automatically compute and provide element values. You can recognize a function by the curly brackets that enclose it. Here, the {CTLSET} function is used to compute the value for element 329 at ST02.



For details on how to get functions into your test data value list, and how to edit them, see [Functions Folder](#) on page 48.

Like values, functions are appropriate only in the right context. For example, you would not want to use {DATE}, which yields today's date, if you are simulating receipt of a transaction that is ten days old. You could use {DATE-10}, though.

There may be more than one function that will produce the same value. For example, you can use either {CTLINT} or {VALUE(ISA13)} to insert the interchange control number into IEA02. Normally, either would work. However, if you then change the value of ISA13, different values would result in IEA02.

Complete List of Functions

Functions marked with \pm can be used in calculations: {DATE + 5}.

{ASSOC}

This supplies the association assigned code, such as EURDEC. It is often null except for industry-assigned subsets. Example value returned: EURDEC in UNG and UNH data element 57.

{CTLGRP \pm }

The control number for the functional group. Initialized in the enveloping model with the Counters selection. Automatically increments with each new functional group.

{CTLINT \pm }

The control number for the interchange. Initialized in the enveloping model with the Counters selection. Automatically increments with each new interchange.

{CTLSET \pm }

The control number for the transaction set or message. Initialized in the enveloping model with the Counters selection. Automatically increments with each new transaction set.

{DATE \pm }

The system date on the computer at generation time (in *yyymmdd* or *yyyymmdd* format, depending on length of element): 980518 or 19980518 is May 18, 1998.

{ELTSUM(*xxx**yy*)}

The numeric sum of the values of all occurrences of the identified element *up to this point* in this transaction set. *xxx* refers to the (2 or 3 digit) segment ID, *yy* refers to the position within the segment. Example: {ELTSUM(PO102)} is the sum of the contents of the element in position 02 of the PO1 segment.

{*ELTSUM(*xxx**yy*)}

*ELTSUM (note the *) is like ELTSUM but allows forward referencing. With this, functions in one element can use data that appears in another element later in the script. See [Forward Referencing](#) on page 83.

{EMPTY}

TDG will not include any data for this element.

{FUNCID}

The functional identifier for this group of transaction sets (used in the GS control segment). In X12, including UCS/WINS and TDCC, this is a two-character code such as PO for 850, IN for 810, etc. In EDIFACT, this is the same as {TRANID}.

{HLCHILDREN}

Value is 1 if there are any HL loops embedded or called from the current HL loop, and 0 otherwise. {HLCHILDREN} is the default element value for element 736, the Hierarchical Child Code.

{HLPARENT}

When used within a nested HL loop, this function returns the Hierarchical ID Number (HL01) of the HL loop to which this one is subordinate. {HLPARENT} is the default value for element 734, Hierarchical Parent ID Number.

{LOOPID}

The value of the loop ID in an element (if the standard has not been violated at the point of insertion, such as when purposely creating errors). See [LOOPID Example](#) on page 83.

{NUMGRPS[±]}

The number of functional groups, including the current one, within this interchange *up to this point*.

{NUMSEGI[±]}

The number of segments, including the current one, in this interchange *up to this point* (incl. all control segments encountered).

{NUMSEGT[±]}

The number of segments, including the current one, in this transaction set or message *up to this point* (including ST and SE or UNH and UNT if encountered).

{NUMSETG[±]}

The number of transaction sets or messages, including the current one, within this functional group *up to this point*.

{NUMSETS[±]}

The number of transaction sets within this interchange *at this point* including the current one.

{RAND(*min,max*)}

A random number between *min* and *max*. *max* must be larger than *min*. **{RAND (1 , 99) }** would give a value between 1 and 99, inclusive. *min* and *max* can be up to 9 digits:
{RAND (22222222 , 99999999) } .

Up to two RAND functions can be on one line to give a random number with up to 18 digits (you cannot use more than two RANDs on one line):

{RAND (111111111 , 999999999) } {RAND (000000000 , 999999999) }

To increase the length of the value, you can use a combination of one RAND function (not two), plus up to 26 integers:

1285667470{RAND(1,999999999)}967342516

{RELEASE}

The release being used. Example value returned: 1 in UNG and UNH data element 54.

{SEDELIM}

This supplies the subelement delimiter as shown in the enveloping model. See [Enveloping Models](#) on page 55.

{SEGCOUNT(*seg,lev*)}

The number of times an identified segment has occurred up to this point in this transaction set. *seg* is the segment tag. *lev* is optional, and is the loop level at which the segment should be counted. A *lev* of **0** is the current loop level, **1** is the parent loop, etc. Example: **SEGCOUNT(PO1)** or **SEGCOUNT(NOI,1)**.

{TIME[±]}

The system time on the computer at test data generation time (in *hhmm* format)

{TRANID}

The transaction set or message identifier (used in the ST or UNH segments). EDIFACT: six characters such as **ORDERS**. X12, UCS/WINS, TDCC: 3 digits such as **850**.

{VALUE(*xxx,yy*)}

The value of the most recent previous occurrence of the identified element. *xxx* refers to the (2 or 3 digit) segment ID, *yy* refers to the position within the segment. Example: **VALUE(ST02)**

{VER}

The abbreviated version identifier of the EDI standard used in the message/set model name. Example value returned: **3030**. **{VER}** is used in the GS control segment. See also VRI, VERSION, RELEASE, and ASSOC.

{VERSION}

This supplies the version identifier being used. Example value returned: **92** in UNG and UNH data element 52.

{VRI}

Supplies the version, release, and industry code being used. This system variable is the default for the X12 and TDCC GS envelope header, GS08 Data Element 480. Example value returned: **003010VICS**. 003 is the version, 010 is the release, and VICS is the industry code.

LOOPID Example

{LOOPID} inserts the value of the loop ID in an element (if the standard has not been violated at the point of insertion, such as when purposely creating errors).

Example: The loop ID for the first LDT loop is 3210, so the LS01 (element 447) above it should contain the value 3210, and the LE01 (element 447) after it should contain the value 3210. Using {LOOPID} in the Test Data pane for the LS01 and LE01 will automatically provide this.

{LOOPID} generally goes in LS and LE segments, and contain the value of the loop that they surround. The value inserted by {LOOPID} is either:

- the 4-digit loop ID as shown above, or
- where there is no 4-digit loop ID, the name of the loop's first segment.

Element 447 contains {LOOPID} as the default value for segments LS and LE. Although {LOOPID} will be the default in transaction 997, element 447, segment AK3, you might want to change it because it is not meaningful in that transaction.

Forward Referencing

***ELTSUM** sums the numeric values of all occurrences of a particular element in the transaction set. The element that it is summing can appear LATER in the transaction set. (*ELTSUM is similar to ELTSUM, which sums the data in an element occurring PREVIOUSLY in the transaction set.)

The format is: {*ELTSUM(×××××)}

×××× is the 2- or 3-character segment ID.

×× is the position within the segment.

Example: Enter { *ELTSUM (RMT03) } in X12-3010, segment BPS02 in the 820 transaction set. This sums the contents of all occurrences of RMT03 in the transaction set.

Functions in Default Values

Many default values (those provided by TIBCO Foresight) are actually functions. These are listed below.

Envelope segments

The TDG section of the EDISIM Reference Manual contains a complete list of functions in envelope segments.

function	location	element	
BG	BG05	29	{DATE}
	BG06	30	{TIME}
	BG07	404	{CTLINT}
EG	EG01	404	{CTLINT}
	EG02	405	{NUMGRPS}
	EG03	97	{NUMSETG}
	EG04	96	{NUMSEGT}
GE	GE01 Number sets	97	{NUMSETG}
	GE02 Control number	28	{CTLGRP}
GS	GS01 Functional ID	479	{FUNCID}
	GS04 Date	29	{DATE}
	GS05 Time	30	{TIME}
	GS06 Group control number	28	{CTLGRP}
	GS08 Version/Release ID	480	{VRI}
IEA	IEA01 No. functional groups	116	{NUMGRPS}
	IEA02 Control number	112	{CTLINT}}
ISA	ISA09 Interchange Date:	108	{DATE}
	ISA10 Interchange Time:	109	{TIME}
	ISA13 Control number:	112	{CTLINT}
	ISA16 Component Elem delim	115	{SEDELIM}
SE	SE01 No. segments	96	{NUMSEGT}
	SE02 Control number	329	{CTLSET}
ST	ST01 Transaction set ID	143	{TRANID}

	ST02 Control number	329	{CTLSET}
TA1	TA101 Interchg Control Number	I12	{CTLINT}
	TA102 Interchg Control Date	I08	{DATE}
	TA103 Interchg Control Time	I09	{TIME}
UNB	UNB0401 Date of Preparation	0017	{DATE}
	UNB0402 Time of Preparation	0019	{TIME}
	UNB05 Interchg Control Ref	0020	{CTLINT}
UNE	UNE01 Number of Messages	0060	{NUMSET}
	UNE02 Funct. Group Ref. Num.	0048	{CTLGRP}
UNG	UNG01 Funct Grp Identification	0038	{FUNCID}
	UNG0401 Date of Preparation	0017	{DATE}
	UNG0402 Time of Preparation	0019	{TIME}
	UNG05 Funct Grp Ref Num	0048	{CTLGRP}
	UNG0701 Msg Type Ver Num	0052	{VERSION}
	UNG0702 Msg Type Rel Num	0054	{RELEASE}
	UNG0703 Assoc Assigned Code	0057	{ASSOC}
UNH	UNH01 Message Ref Num	0062	{TRANID}{CTLS}
	UNH0201 Message Type Ident	0065	{TRANID}
	UNH0202 Msg Type Vers Num	0052	{VERSION}
	UNH0203 Msg Type Rel Num	0054	{RELEASE}
	UNH0205 Assoc Assigned Code	0057	{ASSOC}
UNT	UNT01 Num of Segs in Message	0074	{NUMSEG}
	UNT02 Message Ref Num	0062	{TRANID}{CTLS}
UNZ	UNZ01 Interchg Control Counter	0036	{NUMGR}
	UNZ02 Interchange Control Ref	0020	{CTLINT}

CTT Segment and Functions

The CTT01 and CTT02 elements are context sensitive to the transaction set being used. However, since TDG's default values are not sensitive to the context in which they are used, you must explicitly insert the function for CTT01 and CTT02 as shown below into the appropriate transaction set.

Trans.Set	CTT01	CTT02
------------------	--------------	--------------

810	{SEGCOUNT(IT1)}	{ELTSUM(IT102)}
819	{SEGCOUNT(JIL)}	{ELTSUM(JIL03)}
822	{SEGCOUNT(ACT)}	na
830	{SEGCOUNT(LIN)}	{ELTSUM(FST01)}
832	{SEGCOUNT(LIN)}	na
840	{SEGCOUNT(PO1)}	{ELTSUM(PO102)}
843	{SEGCOUNT(PO1)}	{ELTSUM(PO102)}
844	{SEGCOUNT(CON)}	{ELTSUM(QTY02)}
845	{SEGCOUNT(CON)}	{ELTSUM(QTY02)}
849	{SEGCOUNT(CON)}	{ELTSUM(QTY02)}
850	{SEGCOUNT(PO1)}	{ELTSUM(PO102)}
852	{SEGCOUNT(LIN)}	na
855	{SEGCOUNT(PO1)}	{ELTSUM(PO102)}
856	{SEGCOUNT(HL)}	{ELTSUM(SN102)}
860	{SEGCOUNT(POC)}	{ELTSUM(POC03)}
861	{SEGCOUNT(RCD)}	{ELTSUM(RCD02)}
862	{SEGCOUNT(LIN)}	{ELTSUM(FST01)}
865	{SEGCOUNT(POC)}	{ELTSUM(POC03)}
866	{SEGCOUNT(DTM)}	{ELTSUM(QTY02)}
869	{SEGCOUNT(HL)}	na
870	{SEGCOUNT(HL)}	{ELTSUM(PO102)}

HL Segment and Functions

TDG has two functions that can automatically coordinate HL (Hierarchical Level) segments. See Transaction Set 856, Loop HL, for an example.

{HLPARENT}

When used within a nested HL loop, this function produces the Hierarchical ID Number (HL01) of the HL loop to which this one is subordinate. {HLPARENT} is the default value for element 734, Hierarchical Parent ID Number.

{HLCHILDREN}

This function will return 1 if there are any HL loops nested within the current HL loop, and 0 otherwise. {HLCHILDREN} is the default value for element 736, the Hierarchical Child Code.

The HL segment contains four elements:

01 element 628 (Mandatory)	AN 1-12	Hierarchical ID Number
02 element 734 (Optional)	AN 1-12	Hierarchical Parent ID Number
03 element 735 (Mandatory)	ID 1-2	Hierarchical Level Code
04 element 736 (Optional)	ID 1-1	Hierarchical Child Code.

The mandatory ID Number, HL01 (element 628) is usually assigned an ascending sequence number representing the ordinal position of the HL within the transaction set. The function, {SEGCOUNT(HL)}, which is the default element value for data element 628, can be used to assign this ID number: the first HL01 in the transaction set will be assigned 1, and will be incremented by 1 in each subsequent HL segment within the transaction.

The optional Parent ID number, HL02 (element 734), is used to indicate the Hierarchical ID Number of the HL segment to which this one is subordinate. If the current HL has no parent, it should be unused so no value is generated. For all other nested HLs, use of the function {HLPARENT} yields the ID number of the enclosing loop's HL segment. {HLPARENT} is the default value for element 734. Note that the HL loops must be nested in order for the {HLPARENT} function to produce the correct parent ID.

The mandatory Hierarchical Level Code, HL03 (element 735) determines the context of the current HL loop, e.g., group, category, item, etc. Generally, you'd select the appropriate code value from the list of acceptable codes for element 735.

The optional Child Code, HL04 (element 736) is used to indicate whether this HL loop contains any nested HL loops. The default value for element 736 is the function {HLCHILDREN}.

Nesting Your HL Loops

If your HL loops are not nested (if you see them all at once), use the **Edit | Insert | Types | Set** and insert each lower level HL loop down a level into its parent HL loop. They should be the last item in the parent loop.

Then, highlight the original loop headers and use **Edit | Delete** to remove them.

8 Advanced Topics

External Models

After you have customized a segment, loop, group, composite, or element, you might want to use it elsewhere. That's what the external model lets you do: give a name to it so that you can then use it in another place - perhaps in another transaction set, message, or standard. Using external models can save a lot of effort and give you consistency.

An external model includes all underlying components. For instance, an external model for a segment would include all the elements that it contains, plus all the values that each element contains. It might contain other external models.

When you use an external model, you are simply sharing the original. If you make changes to the external model in one location, those changes will apply globally - wherever this external model is used.

These commands let you work with external models:

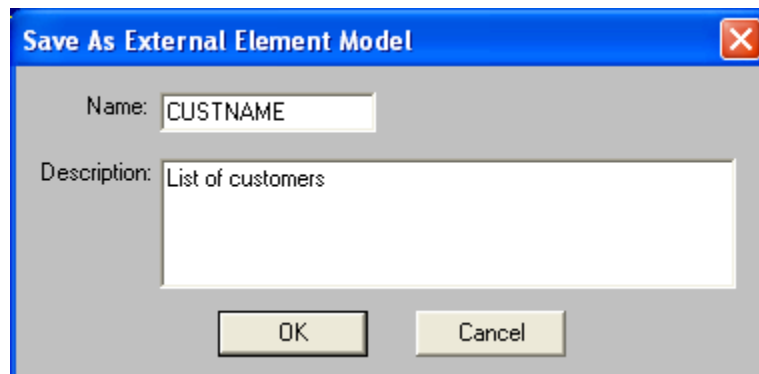
- **File | Save Externally** lets you create an external model by naming the currently highlighted object and all of its subordinates. See [Creating an External Model](#) below.
- **Edit | Replace:** attaches an external model to the current object. See [Using An External Model](#) on page 90.
- **File | Save Locally:** detaches the currently highlighted object from the external model, but leaves the model's structure behind. You then can change it and have the changes be strictly local. It will be completely independent of the external model. See [Detaching an External Model \(Embed\)](#) on page 91.

For details on how values lists cycle in external models, please see [How Values Lists Cycle](#) on page 92.

Creating an External Model

To create an external model:


1. From the top pane, customize the object and all of its subordinates.
2. Click on the object and choose **File | Save Externally**.
3. Enter a name (see [Model Names](#) on page 22) and description for the model. If the name already exists, you will be asked whether to overwrite the model.



After pressing **OK**, the model is available for use in any model that contains that element, segment, loop, or group.

Using an External Model

To replace the current object with an existing external model of the same kind of object:

1. Go to the top pane.
2. Highlight the object that you wish to replace with an external model.
3. Choose **Edit | Replace** or use the  toolbar button. Click the **Types** line at the bottom and select the external model type. TDG displays a list of all models of that type.
4. Choose the model that you wish to use and click **OK**. This attaches the selected external model to the current object.

You will see the name of the external model in the Model column in the top pane and all objects in the external model will have a shaded background.

Editing an External Model

When you edit an external model, you are making a global change that affects all places where the external model is being used. If you want your changes to be strictly local, see [Detaching an External Model \(Embed\)](#) below.

To edit an existing external model:

1. Go to the top pane of a model that has the external model attached.
2. Double-click on the external model's name in the Model column.
3. Look at the title bar. You are now editing the external model. Make any changes that you'd like to make.
4. Click on the closed folder button on the toolbar or choose **File | Close**.
5. When asked if you want to save, answer Yes.


Detaching an External Model (Embed)

You may wish to change an object that has an external model attached, but have the change be strictly local. To do this:

1. In the top pane, highlight the object with the attached external model. You should see the model's name in the Model column.
2. Choose **File | Save Locally**.
3. The external model name should disappear from the Model column, and the shading should disappear. All other aspects of the external model will remain.

Replacing an External Model with Something Else

You may wish to replace an object that has an external model attached.

1. In the top pane, highlight the object with the attached external model. You should see the model's name in the Model column.
2. Choose **Edit | Replace** or use the  toolbar button.
3. Under **Types**, choose the type of object that you would like to use.
4. Choose the desired object from the list and click **OK**.

For details about types of objects, see [Inserting a Segment, Loop, or Group](#) on page 43.

Deleting an External Model

To delete (as opposed to detach) an external model:

1. Choose **File | Delete Models**.
2. In the Types area at the bottom, choose the type of model.
3. In the list of models, select the model.
4. Click **Delete**.

If you get the diagnostic Internal Problems Deleting Model, this means that the external model is used elsewhere. Instead of deleting, try using **Edit | Replace** and replacing it with the Set definition or the dictionary definition. You can then delete the segment. You can also consider clicking off the Use column.

How Values Lists Cycle

Each element in TDG has a values list, even if it just contains one value. You don't have to be concerned about including the right number of values, because TDG starts again at the top of the list if it runs out of values for the element.

Understanding what causes value lists to cycle (to use the next value in a given list) helps you to know:

- When to use a segment repetition count.
- When to insert segments in line versus using a loop.
- How external models are shared.

We use the term model to refer to any of these:

- A script
- A message/set model
- An enveloping model
- A loop, segment, composite, or element that has been saved as an external model.

Every model remembers whenever any element value list within it is used. With each use, the model will provide the next value in its value list. Several examples follow.

Segment Repeated by Segment Repeat Count

Assume you have one PID segment, but it has a repeat count of 2:

Segment	Use
---------	-----

ST	1	
PID	2	(PID05 contains numeric values 1, 2, 3, etc.)
SE	1	

Element PID05 contains numeric values 1, 2, 3, etc. The message/set model will treat element PID05 as a single value list and the output would resemble:

```
ST...
.
.
.
PID ... 1
PID ... 2
.
.
.
SE...
```

Segment Repeated on Different Lines within a Message/Set Model

Suppose you have two separate PID segments, each with a use of 5:

Segment	Use	
ST	1	
PID	5	(contains numeric values 1, 2, 3, etc.)
PID	5	(contains alphabetic values A, B, C, etc.)
SE	1	

Each PID contains element PID05 with a value list of 10 item descriptions. The two lists are independent throughout the script:

```
ST...
.
.
.
PID ... 1
PID ... 2
PID ... 3
PID ... 4
PID ... 5
PID ... A
```

PID ... B
PID ... C
PID ... D
PID ... E
.
.
.
SE...

If the message/set model is used repeatedly within the script, each of these lists will continue to cycle. The next time the first PID was encountered, the value of its PID05 would be 6.

Segment Repeated by Loop Repeat Count within a Message/Set Model

Here, you have two PID segments. One is contained in a loop for which you have set a repeat count of two. The other is not in a loop.

Segment	Use	
ST	1	
FOB	1	
N1 loop	2	(loop is not an external model)
N1	1	
PID	1	(values list contains digits 1-10)
PID	1	(values list contains digits 1-10)
MEA	1	
SE	1	

Two value lists will cycle here for element PID05: one for the loop, regardless of repeat count, and the other for the separate PID. Output data would resemble:

ST ...
.
.
.
N1...
PID ... 1(from PID in loop)
N1...
PID ... 2(from PID in loop)
PID ... 1(from PID below loop)

SE ...

Segment Repeated by Loop Repeat Count within an External Loop Model

This is the same as the previous case, except the loop is itself a model, since it has been saved with **File | Save Externally**. Now it is the loop model (not the set model) that tracks its own count. Therefore, if this same external loop model is used within the same set model, or within a different set model within this same script, it will continue its original cycling.

Segment	Use	Notes
<i>First Set-Enveloping Pair</i>		
ST	1	
➤N1 SmallN1	3	(loop is an external model)
N1	1	
PID	2	(values list contains digits 1-10)
PID	1	(values list contains digits 1-10)
SE	1	
<i>Second Set-Enveloping Pair</i>		
ST	1	
➤N1 SmallN1	3	(loop is an external model)
N1	1	
PID	2	(values list contains digits 1-10)
PID	1	(values list contains digits 1-10)
SE	1	

Output would resemble:

ST ...

(external loop model N1 starts - Qty 3)

N1 ...

PID ... 1

PID ... 2

N1 ...

PID ... 3
 PID ... 4
 N1 ...
 PID ... 5
 PID ... 6
 (external loop model N1 ends)
 PID ... 1
 SE...
 ST...
 (external loop model N1 starts - Qty 3)
 N1 ...
 PID ... 7
 PID ... 8
 N1 ...
 PID ... 9
 PID ... 10 (last value in list used)
 N1 ...
 PID ... 1 (starting over with first value in list)
 PID ... 2
 (external loop model N1 ends)
 PID ... 2
 SE...

Events that Affect Value List Cycling

The following events affect value list cycling:

- An enclosing segment has a repeat count greater than 1.
- An enclosing loop has a repeat count greater than 1.
- The enclosing message/set model is used multiple times within the script.
- For external element or composite models: The element is used elsewhere within the same script.
- For external segment or loop models: An enclosing loop is used elsewhere within the same script.

This example shows the effect of repetition or use counts.

Pair Qty	Loop Count	Segment Count		
2			message/set model: Sm850	
	3		PO1 loop (not an external model)	
		4	PID segment (not an external model)	
			PID05 element (not an external model)	
				value 1 value 2 value 3 value 4 value 5 value 6 value 7 value 8 value 9 value 10

The PID05 (description) element will be invoked 24 times. If there are 10 values in its value list, the full list will cycle twice, and the first four values will be used a third time. The script is the largest entity in which cycling occurs: multiple scripts created by the Script Repetition Count do not influence cycling.

Backing Up and Restoring Your Work

Be sure to back up your work. Files do get corrupted. All hard drives will eventually fail.

One common reason for database corruption is that two TDG sessions are running at once and using the same database. This can happen if the TDG database is on a network. Since TDG does not support database sharing, each user on a LAN must have a separate database.

Another possible reason for database corruption is a system lockup or power failure during update of the database.

The methods of backing up your work are:

- **File | Import** and **File | Export** from within TDG will import and export models one at a time.
- **From within TDG.** Use **File | Export All** if you do not currently have a message/set model, enveloping model, script, or external model open. This will export all models at once.

- Copy the entire contents of EDISIM's Models folder to back up the entire TDG database. This should include **models.mdb** file. With this method, you cannot select individual models to back up or restore. Restorations overwrite the existing database completely.
- A command-line utility called FSdosutl is slower than copying the database files, but restorations can go into an existing database without overwriting what is already there. For details, see [Export and Import](#) on page 74.

Back up your work to more than one place. One backup can be to a separate folder on your hard drive. Another backup should be to a floppy disk, to the network, or to somewhere else that is not part of your computer.

When backing up, don't overwrite your most recent backup. For example, if you back up to floppy disks, do not always back up to the same disk. Alternate between several disks or several sets of disks. Use the same principal if you back up to tapes, network drives, CDs, etc.

If Your PC Becomes Locked Up

Each time you save from within TDG, the saved information is also written to a temporary file called \$\$\$1.TMP, \$\$\$2.TMP, etc. If you experience a system failure and have to reboot, then discover that your model is not usable, you can import the temporary file to restore your work. This may be necessary if you use a write disk cache, such as SMARTDRV.

Look at the size, date, and time of each \$\$\$n.TMP filename. Identify the last one created that is not zero bytes long. It is probably the correct file to import.

Now import it with **File | Import**. You should then be able to restart TDG and access your models and scripts. Check the model you were last working on.

Files ending with .TMP may be deleted once they are no longer needed for importing.

Incomplete Saves

If TDG cannot complete a save because the disk is full, memory runs out, etc., it will attempt to restore the database to its pre-save condition. When you re-start TDG, you will see a diagnostic about the incomplete save. Wait until the diagnostic disappears, and then check whatever you were working on.

Wrapping and Folding Data with EDITWRAP

From within TDG, you can specify these formats for your EDI file:

- (default) appear one segment per line, terminated by a segment terminator (such as an exclamation mark, !) and a CR/LF.
- be placed all on one line, with no CR/LFs. You can specify this wrapped method by selecting the **Wrap Data** check box in the Test Data Generation box.

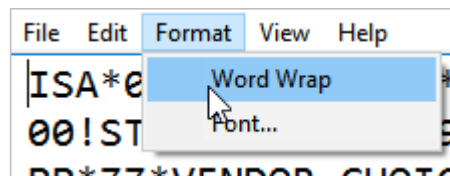
This choice is explained in [Generating Data from a Script](#) on page 68. You can also choose to use **Suppress Padding**, which is active only when Wrap Data is selected. If it is not selected, the file is padded out to 80 characters.

When viewing a test data file with Notepad or by other means, the data may appear to be wrapped when it isn't. Check Notepad's setting for **Edit | Word Wrap**.

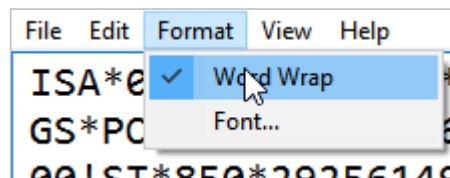
If you have other wrapping or folding needs, use EDITWRAP. This DOS utility processes an EDI data file and saves it to a new file name.

Wrapped and Unwrapped Data

To see a data file as first created without wrapping, open it in Notepad. Check that Notepad's **Format | Word Wrap** does not have a check beside it.



To see a data file with wrapping, open it in Notepad. Check that Notepad's **Format | Word Wrap** has a check beside it.



Unwrapping, Wrapping, and Folding Data with EDITWRAP

EDITWRAP is a DOS utility in EDISIM's Bin folder. It lets you wrap, unwrap, and fold EDI data in various ways. The file may be from TDG or from elsewhere.

EDITWRAP Command Format

The following is complete list of command line arguments for EDITWRAP. To abort EDITWRAP, type *Ctrl+C*.

EDITWRAP [-I<inputfile>] [-O<outputfile>] [-T<tname>] [-w<width>] [-f] [-nl] [-t{c|xx}] [-b<seglist>] [-U]

where:

- I** Input file, the file produced by TDG. Include the path if the file is not in your current folder. (default: *script.txt*, where *script* is the name of the script.)
- O** Output file, the file created by EDITWRAP, containing the processed data. Include a path if you don't want the file to go to your current folder. (default: EDITDATA)
- T** Totals file. This is used when forward ELTSUM functions have been used (e.g., `{*ELTSUM(RMR04)}` in the BPR segment of the ASC X12 820 transaction) and you are running TDG from DOS. See [Forward Referencing](#) on page 83 for details about forward referencing. Include a path if the file is not in your current folder. (default: *script@.txt*).

-T is not available for NCPDP data.
- w** Width is the record length (default: 80). All segments will be strung together and then chopped into records of the specified size. Each ISA will start on a new line, and the previous incomplete line will be padded with blanks. If width is specified as 0 (`-w0`), then no wrapping occurs; this is used when you used forward referencing with the `*ELTSUM` function. The new totals are put in the data but it is not otherwise reformatted.
- f** Fold. EDITWRAP is to insert a CR/LF after reaching the width specified in the `-w` parameter (or, if `-w` is not used, records will be folded after the 80th character). Each segment starts on a new line, and if a segment is longer than the specified width, it is broken to a new line.
- nl** Data is strung together and then chopped at every width boundary (specified with `-w`), but a new segment does not cause the data to break to a new line.
- t`xx`** Terminator. Specifies which character or hex code is the segment terminator. If `xx` is a single character, EDITWRAP assumes it is the segment terminator; If `xx` is two characters, EDITWRAP assumes it is the 2-digit hex ASCII representation of the segment terminator.

If the data is already unwrapped, the specified character will replace segments separated by CR/LF. If the input data is wrapped and you include a `-U` parameter, each segment terminator will have a CR/LF appended to it.
- b`<seglist>`** Break to a new line for certain segments. Example:
-bUNB,UNG would cause every UNB and UNG segment

to start on a new line. -b accompanied by no segment list means that the data should not be broken to new lines for any segments.

-b is not available for NCPDP data.

-U Unwrap. U unwraps data, breaking each segment to a new line. Each segment terminator (specified by the -t option) will have a CR/LF inserted immediately afterward.

Example: Removing CR/LF

This example removes all CR/LF in the file.

```
EDITWRAP -Iinput.txt -Ooutput.txt
```

Example: Adding CR/LF after Segment Terminator

This example adds a CR/LF after each segment terminator. The segment terminator in the input file is ~.

```
EDITWRAP -Iinput.txt -Ooutput.txt -U -t~
```

The output file will start each segment on a separate line. If data has CR/LF within segments, they will remain. Any inter-segment blank padding is eliminated. Only one segment terminator can be used throughout the transmission file, even if there are multiple interchanges.

Example: Adding CR/LF at Specific Length

EDITWRAP can fold lines at a specified length. After using

```
EDITWRAP -Ifilename -f -w40
```

the example file would resemble the following. Existing CR/LFs are left alone, but additional ones are added for lines longer than 40 characters.

```
ISA*00*                *00*                *01*S5003
8906666   *01*R54358902378   *930825*062
0*U*00200*000000113*0*T*~!
GS*PO*S50038906666*R54358902378*930825*0
620*113*T*002000!
ST*850*292561499!
BEG*06*SA*042K07076***871019!
NTE*GEN*                IF SHIPMENT CANNOT
  BE MET BY REQUESTED DATE,!
NTE*GEN*                CALL P
LANT IMMEDIATELY.!
N1*BT**09*0020174400895!
N1*ST*MAGNOLIA I   *9*0020174400042!
N3*I-80 HIGHWAY 10!
N4*BLACKSBURG      *SC*29702!
N1*SF*CHEMICALS INTERNATIONAL INC*92*014
4BAC!
ITD*ZZ*ZZ*****1/15 1-25TH 16/31 1-1
0TH!
FOB*PP*ZZ*VENDOR CHOICE      *ZZ*CIF*DE*B
UYER'S PLANT!
PO1*01*0000552*LB*0000003.8200**PI*99012
036*VP*27052!
```

J2X***DISPERSAL ORANGE D-3RA LIQ!
PER*BD*FRED D. JOHNSON!
SCH*00000552*LB***002*871020*900415!
CTT*1*001!
SE*000017*292561499!
GE*1*113!
IEA*1*000000113!
ISA*00* *00* *01*S5003
8906666 *01*R54358902378 *930825*062
0*U*00200*000000114*0*T*_!
.
.
.

9 Samples

Sample Scripts

\$SAMP1	(If X12 standards or industry guidelines were installed) Includes message/set models \$SAMP810 and \$SAMP850. These were built from the SAMP1 guideline, which you can see in Standards Editor.
\$SAMP2	(If EDIFACT standards or industry MIGs were installed) Includes message models \$SAMINV and \$SAMORD. These were built from the SAMP2 MIG, which you can see in Standards Editor.

Sample Message/Set Models

X12

\$SAMP810	Built from SAMP1 guideline - Used in Script \$SAMP1
\$SAMP850	Built from SAMP1 guideline - Used in Script \$SAMP1

EDIFACT

\$SAMINV	Built from SAMP2 guideline - Used in Script \$SAMP2
\$SAMORD	Built from SAMP2 guideline - Used in Script \$SAMP2

Sample Enveloping Models

\$X12	To be used with versions of X12 standards older than 3072.
-------	--

\$X12Y2K	To be used with version of X12 standards for 3072 and newer Used in Script \$SAMP1
\$EDIFACT	Sample PD model for EDIFACT - includes UNG/UNE service segments.
\$EDIFAC1	Sample Enveloping Model without UNG/UNE service segments.
\$EDIFAC2	ISO 9735 Version 4 without UNG/UNE service segments.
\$EDIFAC4	ISO 9735 Version 4 including UNG/UNE service segments.
\$UCSBGEG	UCS customers who use BG/EG envelope segments.
\$TDCC	GS only with no interchange.

10 Appendix A: TDG Sample HIPAA Transaction Models

Introduction

These sample transaction models are installed in TIBCO Foresight® EDISIM®'s Samples\HIPAA\TDGModels directory.

Import the corresponding guideline into Standards Editor before importing these models into Test Data Generator.

Please see ReadmeHIPAAsamples.txt in that directory for directions.

Sample HIPAA Models based on X12-4010

Model Name	Generates	Guideline needed in Standards Editor
HB270	270 Eligibility, Coverage or Benefit Inquiry with four 2000C loops, including: 2 subscriber-only loops 2 subscriber with dependent loops	270AA120
HB271	271 Eligibility, Coverage or Benefit Response with four 2000C loops, including: 2 subscriber-only loops 2 subscriber with dependent loops	271AA120
HB276	276 Status Request with four 2000D loops, including: 2 subscriber-only loops 2 subscriber with dependent loops	276AA120
HB277	277 Health Care Claim Status Notification with four 2000D loops, including: 2 subscriber-only loops 2 subscriber with dependent loops	277AA120
HB278RP	278 Health Care Services Review Information - Response to Subscriber Request	278RES
HB278RQ	278 Health Care Services Review Information for Subscriber	278REQ
HB820	820 Payment Order/Remittance Advice - Individual	820AA120
HB834	834 Benefit Enrollment and Maintenance - Member enrollment (non-audit)	834AA120
HB835	835 Health Care Claim Payment/Advice - 1 claim payment	835AA120
HB837D	837D Health Care Claim: Dental with four 2000C loops, including: 1 subscriber-only with 2 claims 1 subscriber with a dependent with 2 dependent claims	837AQ220
HB837I	837I Health Care Claim: Institutional with four 2000C loops, including: 1 subscriber-only with 2 claims 1 subscriber with a dependent with 2 dependent claims	837AQ320

Model Name	Generates	Guideline needed in Standards Editor
HB837P	837P Health Care Claim: Professional with four 2000C loops, including: 1 subscriber-only with 2 claims 1 subscriber with a dependent with 2 dependent claims	837AQ120
HIPAAIN	Inbound HIPAA X12-4010 enveloping model example	<i>n/a</i>

Sample HIPAA Models based on X12-5010 Errata

Model Name	Generates	Guideline needed in Standards Editor
5010_Errata_Env.exp	Envelope for X12-5010 and later using a tilde segment terminator.	<i>n/a</i>
HB5010-270X279	270 Eligibility, Coverage or Benefit Inquiry, including: 1 subscriber-only loop	270-X279
HB5010-271X279	271 Eligibility, Coverage or Benefit Information, including: 1 subscriber only loop that has 1 EB loop	271-X279
HB5010-275X210	275 Patient Information, including: 1 LX loop	275-X210
HB5010-276X212	276 Health Care Status Claim Request, including: 1 subscriber-only loop 1 subscriber with dependent loop	276-X212
HB5010-277X212	277 Health Care Information Status Notification, including: 1 subscriber-only loop 1 subscriber with dependent loop	277-X212
HB5010-278X217Q	278 Health Care Services Review Information – Request, including: 1 subscriber-only loop	278X217Q
HB5010-278X217R	278 Health Care Services Review Information – Response, including: 1 subscriber with dependent loop	278X217R
HB5010-820X218	820 Payment Order/Remittance Advice, including: 1 ENT loop for Organizational Summary 1 ENT loop for Individual Remittance	820-X218
HB5010-834X220	834 Benefit Enrollment and Maintenance, including: 1 INS loop containing one HD loop with one LX loop	834-X220
HB5010-835X221	835 Health Care Claim Payment/Advice, including: 1 LX loop containing one CLP loop with one SVC loop	835-X221

Model Name	Generates	Guideline needed in Standards Editor
HB5010-837DX224	837D Health Care Claim - Dental, including: 1 subscriber loop with 2 claims 1 subscriber with dependent loop containing 2 claims	837-X224
HB5010-837DX224A2-Errata	837D Health Care Claim – Dental Errata 1 subscriber loop with 2 claims 1 subscriber with dependent loop containing 2 claims	837-X224
HB5010-837IX223	837I Health Care Claim – Institutional including: 1 subscriber loop with 2 claims 1 subscriber with dependent loop containing 2 claims	837-X223
HB5010-837IX223A2-Errata	837I Health Care Claim – Institutional Errata 1 subscriber loop with 2 claims	837-X223
HB5010-837PX222	837D Health Care Claim - Professional, including: 1 subscriber loop with 2 claims 1 subscriber with dependent loop containing 2 claims	837-X222
HB5010-837PX222A1-Errata	837D Health Care Claim – Professional Errata 1 subscriber loop with 2 claims 1 subscriber with dependent loop containing 2 claims	837-X222
HIPAA-5010	Enveloping for HIPAA 5010 transactions.	<i>n/a</i>

11 Index

{
} 25

A

application values *See* values,application values
ASSOC function 80

B

backing up 19, 98
BG 58, 60
blue pane *See* detail pane

C

changing guidelines/MIGs in Standards Editor 55
circle icon 25
code values *See* values,code values
colors 24
composite repeat counts 41
composites 46
control segments *See* enveloping models,control segments
copying a model 74
Couldn't find EDI standard for Model 70
counters *See* enveloping model,counters
creating data *See* generating data
CTLGRP function 80
CTLINT function 80
CTLSET 58
CTLSET function 79, 80

CTT 85

D

Data Value box 52 data
values directory 47
data, generating *See* generating data
DATE function 79, 80
default values *See* values,default
deleting segments and loops/groups 44
delimiters *See* enveloping models,delimiters
detail pane 47
directories 21
displaying subordinates *See* subordinates-showing
or not showing
drag 51

E

Edisim5.BaseSettings.ini 21
EDITWRAP 100
EG 60
element
repeating 63
element repeat counts 41, 42
ELTSUM function 80, 83
EMPTY function 81
empty values 50
enveloping model 3, 14, 31, 55
control segments 60
counters 61
creating vs. reusing 55
customizing 56

- delimiters 15, 62
- description 57
- envelope type 57
- enveloping method 14, 59
- exporting 64
- opening or creating 56
- saving 64
- enveloping models
 - counters 15
- export 74
- external models 9, 40, 41, 89

F

- faded text 25
- file location 21
- folded data 99
- folder 24
- folders 21
- formulas *See* values/functions
- Fsdosutl 75
- Fstdg.ini 21
- FUNCID function 81
- functions *See* values/functions

G

- GE 60
- GENCOD 58
- generating data 16, 17, 18, 33, 68
 - choosing a file name 69
- gray background 25
- groups *See* loops and groups
- GS 60

H

- HL 86
- HL segments and functions 86
- HLCHILDREN function 81, 86
- HLPARENT function 81, 86

I

- icons 24
- IEA 60
- import 74
- inbound EDI 2
- insert segments, loops, and groups 10, 43

- ISA 57, 60
- ISO 9735 Version 4 58

K

- keystrokes 23

L

- lists of items 27
- literals *See* values, literals
- location of files 21
- lockups
 - what to do 99
- LOOPID function 81, 83
- loops and groups 46
 - external models 9
 - used/unused 8

M

- Max column 7, 41
- Max Use column 7, 40
- menu 23
- message/set model 3, 5, 6, 30, 37
 - closing 55
 - deleting 54, 63, 67
 - exiting 55
 - exporting 54
- Min column 7, 41
- Model column 7, 40

N

- names 22
- navigator 39
- null values 50
- NUMGRPS function 81
- NUMSEGI function 81
- NUMSEGT function 81
- NUMSETG function 81
- NUMSETS function 81

O

- Object column 40
- outbound EDI 2
- overview 4
- overview of TDG steps 29

P

- padding
 - suppress 69, 100
- page icon 25
- Print command 77

R

- RAND function 82
- random number function 82
- Referenced element not checked 70
- RELEASE function 82
- Rep column 41, 42
- Repeat column 7, 10, 40, 42
- repeat count for elements and composites 41
- repeating element 63
- repeating elements 58
- replacing objects 45
- Req column 7, 40

S

- saving 9, 73
 - message/set model 54
- screen - adjusting 5
- script 3, 17, 32, 65
 - count 67
 - deleting 67
 - deleting lines 66
 - exporting 68
 - functional group 67
 - inserting lines 66
 - opening or creating 65
 - repeat count 67
 - replacing objects 66
 - saving 67
- SEDELIM function 82
- SEGCOUNT function 82
- SEGCOUNT function in CTT 85
- shaded background in top pane 41, 90
- starting TDG 3
- steps in using TDG 4
- STX 58
- subordinates-showing or not showing 23
- suppress padding 69, 100
- symbols 24

T

- TDCC 57
- test data pane 47
- TIME function 82
- top pane 39, 40
- TRAD93 58
- TRADACOMS 58
- TRANID function 82
- tutorial 3
- Type column 7, 40

U

- UN4ICS 58
- UNA 60
- UNB 58, 60
- UNE 60
- UNG 58, 60
- UNZ 60
- Use column 7, 40, 46
- used/unused 11, 40, 42

V

- VALUE function 80, 82
- values 11
 - application values 49
 - cannot change 54
 - code values 12, 49
 - code values-black 49
 - code values-faded 49
 - code values-yellow 49
 - copying from other software 50
 - cycling through list 12, 92
 - default 11, 48
 - deleting 14
 - dragging to Test Data list 51
 - editing values in Test Data list 51
 - empty 50
 - errors 13
 - functions 12, 48, 53, 79, 80
 - functions in HL segments 86
 - how to use 47
 - importing 50
 - inserting, deleting, or moving 52
 - literals 12, 50
 - null 50

VER function 82
VERSION function 82
VRI function 83

W

WINS 57

wrapped data 99

X

X12 (Y2K) enveloping 57

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

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of TIBCO Software Inc.

TIBCO, the TIBCO logo, the TIBCO O logo, and EDISIM are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the readme.txt file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

This and other products of TIBCO Software Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (<https://www.tibco.com/patents>) for details.

Copyright © 1991-2021. TIBCO Software Inc. All Rights Reserved.