CHAPTER 1

EDI Schemas

This chapter provides an overview of working with EDI documents and BizTalk schemas. It introduces where the information for EDI schemas comes from, how to modify the schema based on a trading partner's requirements, how to validate the schema and document content, and how to deploy the schema once it has been completed. This chapter should be read along with the content of Chapter 2 on trading partner configuration, as it is essential to understand both concepts to be able to define and complete a full EDI schema and create a full EDI instance, with all header, context, detail, and footer (summary) information.

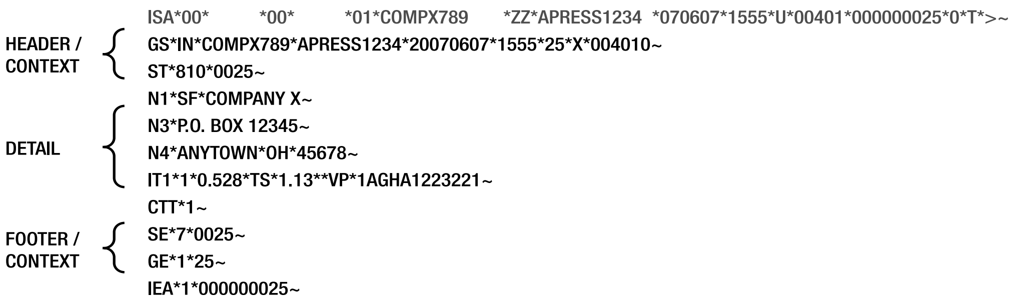
Schemas are considered the foundation of a BizTalk implementation—all other components are dependent on them being in place. The primary steps in defining and working with EDI schemas are the following:

- **Using an EDI implementation guide**: The EDI implementation guide is the starting point for all EDI solutions. It defines all of the documents that a trading partner uses, how those documents are formed (e.g., what segments and data elements are available), and what data is expected to be present when the document is delivered.
- **Determining the right BizTalk schema**: Based on the implementation guide, the appropriate BizTalk schema can be determined. There will likely be multiple schemas in any EDI solution, some specific to trading partners, others shared by multiple trading partners.
- **Modifying the BizTalk schema based on the trading partner**: Once the correct schema has been determined and added to a BizTalk project, the next step is to modify the schema to fit the requirements of the individual trading partner. Often this involves removing unneeded nodes (segments), modifying field lengths, and determining whether data elements are mandatory or optional.
- **Promoting fields**: Field promotion becomes a factor in implementations that will be using send port filtering or orchestration logic based on the content of an EDI document.
- **Validating and generating EDI instances**: During schema development, existing EDI instances will need to be validated and new file instances may need to be created. Validation generally involves taking a known instance of an EDI document that works for a given trading partner and validating it against the BizTalk schema. Based on the results, the schema may need to be further revised.

Schema Overview

All documents processed in BizTalk Server, whether they are EDI or otherwise, adhere to a schema. There are numerous EDI documents in existence, ranging from invoices and bills of lading to functional and technical acknowledgements, and all of these documents have their own schema. On top of that, each EDI trading partner may have its own variation on an individual EDI document: Company A may expect slightly different information in a slightly different format for its invoice than Company X does. Each of these unique documents is defined by a different BizTalk schema (also known as an *XSD*).

Figure 1-1 shows an example of an EDI document. The document has been split into the different parts that will be referred to throughout the course of this book.



```
ISA*00*          *00*          *01*COMPX789       *ZZ*APRESS1234     *070607*1555*U*00401*000000025*0*T*>~
GS*IN*COMPX789*APRESS1234*20070607*1555*25*X*004010~
ST*810*0025~
N1*SF*COMPANY X~
N3*P.O. BOX 12345~
N4*ANYTOWN*OH*45678~
IT1*1*0.528*TS*1.13**VP*1AGHA1223221~
CTT*1~
SE*7*0025~
GE*1*25~
IEA*1*000000025~
```

Figure 1-1. *Diagram of an EDI document representing an X12 810 invoice*

Because of the nature of EDI implementations, document structure, and trading partner requirements, it is common practice to have one BizTalk schema for every EDI document type per trading partner. On occasion, however, a schema may be shared between multiple trading partners; in this case, defining a "common schema project" can be helpful. Figure 1-2 illustrates a sample BizTalk project schema structure for two trading partners that both have unique invoice schemas and a shared bill of lading schema. One of the trading partners also has its own unshared purchase order schema.

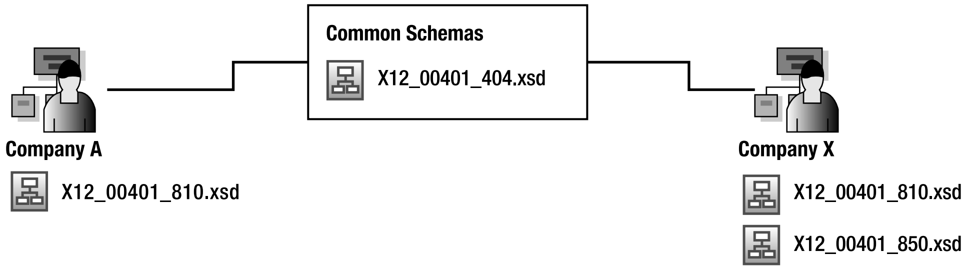


Figure 1-2. *Schema project structure*

Exercise 1-1 walks through setting up the sample code provided for this chapter. This chapter uses the files located in C:\Apress.Integration\Chapter 1.

Exercise 1-1. Preparing the Solution Files

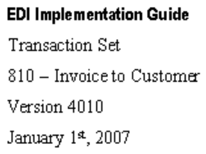
Use the steps in this exercise to prepare the environment on the local machine without having to work through each of the exercises individually:

1. Make sure that the Chapter1 sample files have been extracted and placed on the C:\ drive.
2. The components can be deployed in a single package by importing the Chapter1.MSI file available in C:\Apress.Integration\Chapter 1\BizTalk Application. Open BizTalk Administration Console and right-click the Applications folder. Select Import ➤ MSI File. This will create the EDI.Demonstration.Chapter1 BizTalk application and all of the send and receive ports used in this chapter. It will also import the schemas. Alternatively, there are two solutions that can be deployed manually (rather than using the MSI):
   a. Open the solution contained in C:\Apress.Integration\Chapter 1 called Apress.Integration.EDI810.CompanyX.sln. This solution contains two items: a modified version of the X12 810 schema, tailored to Company X, and a property schema containing information about promoted fields. Deploy this by right-clicking the solution in Visual Studio and selecting Deploy.
   b. Open the solution called Apress.Integration.EDIFACT.ServiceExtension.sln. This solution contains the X12 service schema with modifications. This can also be deployed by right-clicking the solution in Visual Studio and selecting Deploy.

EDI Implementation Guide and Document Layout

The single most important component in determining information about the trading partner is the EDI implementation guide. An implementation guide is available from virtually any company using EDI. This guide outlines all of the documents being exchanged and all of the segments and fields that are used within each document. Implementation guides are generally comprised of the following sections:

- **Transaction set specification**: The transaction set specification is very similar to a title page—it simply defines what document is being covered and gives some high-level information about that document. Figure 1-3 shows a sample transaction set specification page.
- **Introduction**: The introduction is an overview of all the segments that are used within the given transaction set. The segments are defined and it is noted whether they are mandatory or optional and how many times they are expected to appear in a document; any notes associated with the segment are outlined. The introduction is comprised of three parts: header, detail, and footer.
- **Segment definitions**: After the introduction, each segment is presented, including all of the data elements (or *fields*) that are available on the segment, the length, and whether they are mandatory or optional. Any notes associated with each data element are defined in the segment definitions.


EDI Implementation Guide

Transaction Set

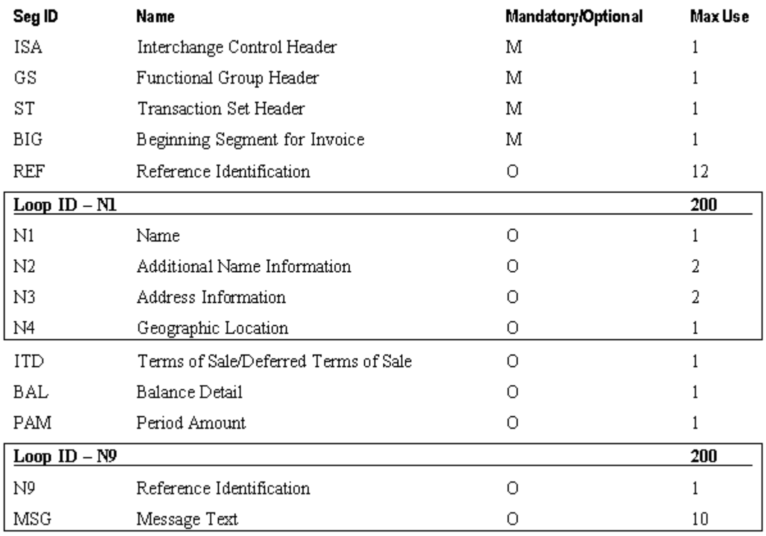
810 – Invoice to Customer

Version 4010

January 1st, 2007

Figure 1-3. *The transaction set specification page in the implementation guide*

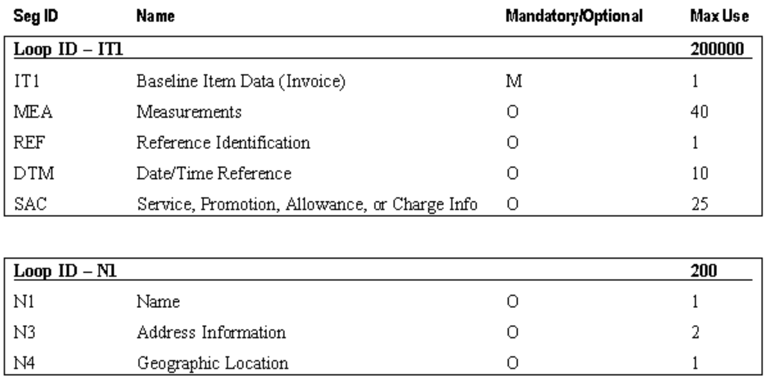
The following figures are representations of what should be found in the introduction of an EDI implementation guide. Figure 1-4 shows the header segments, whether the segment is mandatory or optional, and how many times the segments will appear in the document. Note that there are two looping segments, N1 and N9, both appearing up to 200 times in a given EDI instance.



Seg ID	Name	Mandatory/Optional	Max Use
ISA	Interchange Control Header	M	1
GS	Functional Group Header	M	1
ST	Transaction Set Header	M	1
BIG	Beginning Segment for Invoice	M	1
REF	Reference Identification	O	12
Loop ID – N1			**200**
N1	Name	O	1
N2	Additional Name Information	O	2
N3	Address Information	O	2
N4	Geographic Location	O	1
ITD	Terms of Sale/Deferred Terms of Sale	O	1
BAL	Balance Detail	O	1
PAM	Period Amount	O	1
Loop ID – N9			**200**
N9	Reference Identification	O	1
MSG	Message Text	O	10

Figure 1-4. *Header segment in the introduction*

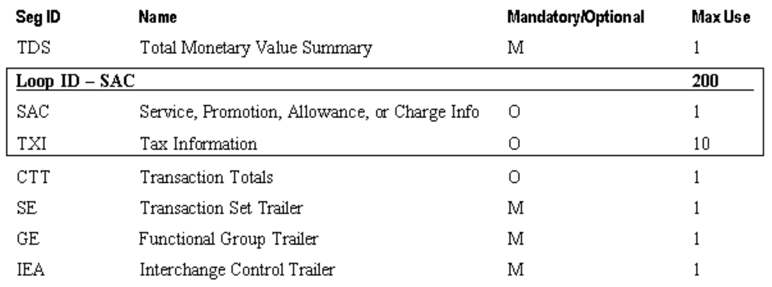
Figure 1-5 shows all of the detail segments and whether they are optional and what their usage is. The IT1 loop can appear one to many times, whereas the N1 loop can appear zero to many times.



Seg ID	Name	Mandatory/Optional	Max Use
Loop ID – IT1			**200000**
IT1	Baseline Item Data (Invoice)	M	1
MEA	Measurements	O	40
REF	Reference Identification	O	1
DTM	Date/Time Reference	O	10
SAC	Service, Promotion, Allowance, or Charge Info	O	25
Loop ID – N1			**200**
N1	Name	O	1
N3	Address Information	O	2
N4	Geographic Location	O	1

Figure 1-5. *Detail segment*

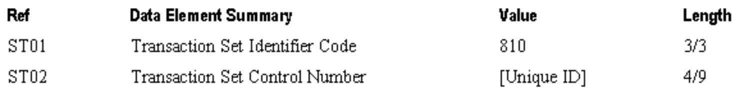
The final section of the implementation guide is the definition of the footer (or summary) segments. These segments are shown in Figure 1-6.



Seg ID	Name	Mandatory/Optional	Max Use
TDS	Total Monetary Value Summary	M	1
Loop ID – SAC			**200**
SAC	Service, Promotion, Allowance, or Charge Info	O	1
TXI	Tax Information	O	10
CTT	Transaction Totals	O	1
SE	Transaction Set Trailer	M	1
GE	Functional Group Trailer	M	1
IEA	Interchange Control Trailer	M	1

Figure 1-6. *Footer (summary) segment*

Figure 1-7 shows a sample version of a segment definition from an EDI implementation guide. The introduction states that the ST header (transaction set header) segment will be in the 810 EDI document, that it is a mandatory segment, and that it will appear one time. The segment definition for the ST segment shows the specific implementation information for that segment. For example, the first field, Transaction Set Identifier Code, has a constant value of 810 and must be exactly three characters long. The second field, ST02, is a number that will be automatically set by BizTalk, ensuring that it is a unique value.



Ref	Data Element Summary	Value	Length
ST01	Transaction Set Identifier Code	810	3/3
ST02	Transaction Set Control Number	[Unique ID]	4/9

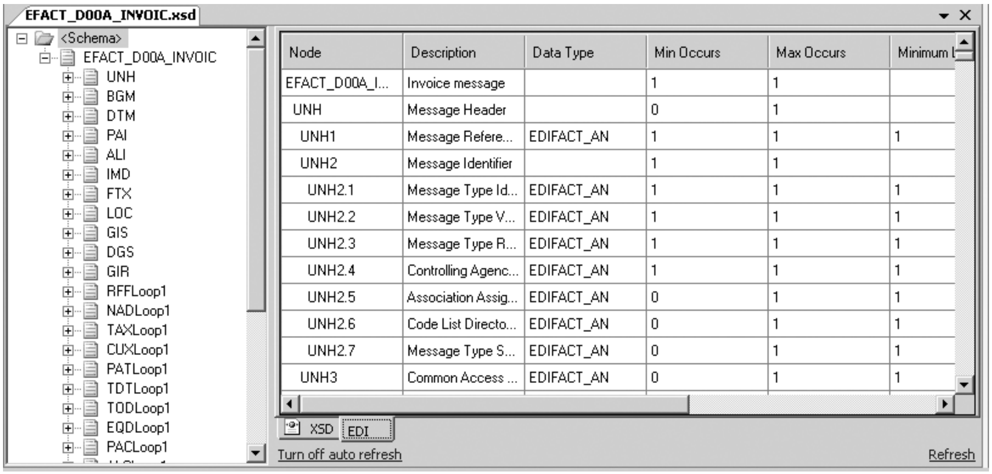
Figure 1-7. *ST segment*

Note The ST segment is different from the majority of segments in that it must be part of the EDI X12 schemas, but at the same time it has some values that are set by the EDI send pipeline. Most segments are either completely set by the pipeline (such as the ISA segment via trading partner configuration) or are completely handled in mapping.

Schema Development and Deployment

BizTalk Server 2006 R2 is packaged with thousands of EDI schemas that can be used as starting points for a BizTalk EDI implementation (accessing these schemas is shown in Exercise 1-2, later in this section). These schemas are designed to contain a superset of all the nodes that may be required by any trading partner for the specified document type. There will frequently be fields that need to be modified (i.e., changing a “required” field to “optional,” changing the maximum length of a field, or changing the type of data that can be stored in a field), and there will frequently be the need to eliminate a number of the nodes that do not pertain to the trading partner in question.

EDI schemas have an additional tab available to them in the Visual Studio Editor, shown in Figure 1-8. This tab gives quick access to some of the most prominent properties that are set for EDI data elements. Aside from this tab, working with EDI schemas is no different than working with any other type of BizTalk schema.



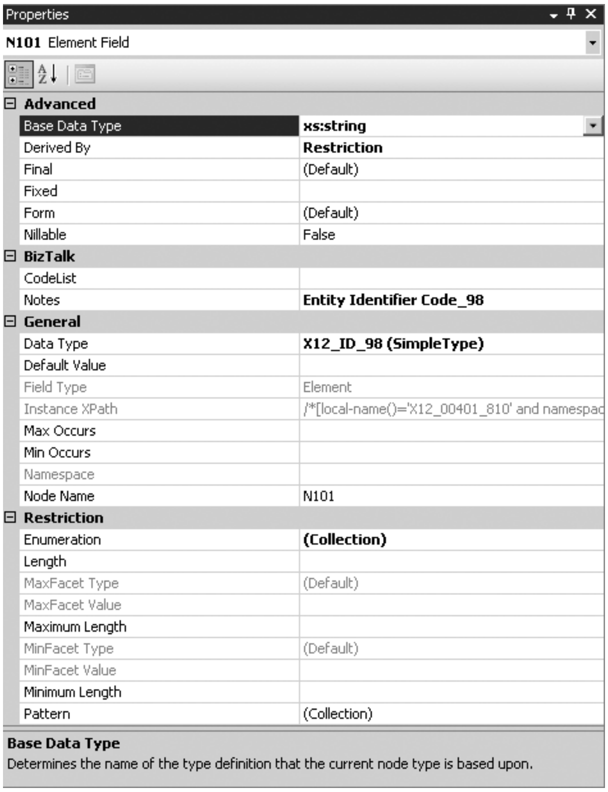
- <Schema>
  - EFACT_D00A_INVOIC
    - UNH
    - BGM
    - DTM
    - PAI
    - ALI
    - IMD
    - FTX
    - LOC
    - GIS
    - DGS
    - GIR
    - RFFLoop1
    - NADLoop1
    - TAXLoop1
    - CUXLoop1
    - PATLoop1
    - TDTLoop1
    - TODLoop1
    - EQDLoop1
    - PACLoop1

Node	Description	Data Type	Min Occurs	Max Occurs	Minimum
EFACT_D00A_I...	Invoice message		1	1	
UNH	Message Header		0	1	
UNH1	Message Refere...	EDIFACT_AN	1	1	1
UNH2	Message Identifier		1	1	
UNH2.1	Message Type Id...	EDIFACT_AN	1	1	1
UNH2.2	Message Type V...	EDIFACT_AN	1	1	1
UNH2.3	Message Type R...	EDIFACT_AN	1	1	1
UNH2.4	Controlling Agenc...	EDIFACT_AN	1	1	1
UNH2.5	Association Assig...	EDIFACT_AN	0	1	1
UNH2.6	Code List Directo...	EDIFACT_AN	0	1	1
UNH2.7	Message Type S...	EDIFACT_AN	0	1	1
UNH3	Common Access ...	EDIFACT_AN	0	1	1

Figure 1-8. *The BizTalk schema EDI tab in Visual Studio*

Figure 1-9 shows all of the properties that can be set in an EDI schema. The most important properties to pay attention to are as follows:

- **Base Data Type and Data Type**: These properties indicate what type of data is expected to be present in the field. There are standard types, such as string and int, but there are also a large number of EDI-specific fields, such as X12_ID. Each of the EDI-specific fields has different properties associated with it (such as enumerations) and will automatically set these values when selected. Generally, the value of the element is set appropriately by default. If a trading partner needs a different type of data in this field, it is easiest to simply set this to xs:string, which allows anything to be present. Of course, by doing this, any EDI validation will be removed from this field.
- **Notes**: This is a description property that states what the field represents. It can be helpful during development and testing.
- **Max and Min Occurs**: These represent how many times the node or element can occur in the document. Occasionally it may be necessary to change these values, especially when a node is marked as "mandatory," but the trading partner does not expect it to be present on every document.
- **Length, Maximum Length, and Minimum Length**: These three fields dictate how long the data in the field can be. It is frequently necessary to modify these values for a given element, since different trading partners expect different lengths of data.
- **Enumeration**: This field contains an array of values that can be entered and compared against a schema. If the value in the field does not match a value in the enumeration, it will not be valid. See Exercise 1-3 (in the "X12 and EDIFACT Schemas" section, later in this chapter) on changing these values.


Properties

N101 Element Field

Advanced	
Base Data Type	xs:string
Derived By	Restriction
Final	(Default)
Fixed	
Form	(Default)
Nillable	False
BizTalk	
CodeList	
Notes	Entity Identifier Code_98
General	
Data Type	X12_ID_98 (SimpleType)
Default Value	
Field Type	Element
Instance XPath	/*[local-name()='X12_00401_810' and namespac
Max Occurs	
Min Occurs	
Namespace	
Node Name	N101
Restriction	
Enumeration	(Collection)
Length	
MaxFacet Type	(Default)
MaxFacet Value	
Maximum Length	
MinFacet Type	(Default)
MinFacet Value	
Minimum Length	
Pattern	(Collection)

Base Data Type
Determines the name of the type definition that the current node type is based upon.

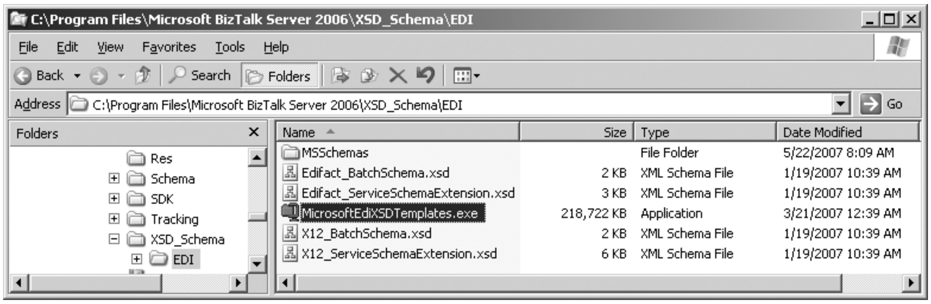
Figure 1-9. *EDI schema element properties*

Exercise 1-2 demonstrates how to access the default EDI schemas that ship with BizTalk Server 2006 R2.

Exercise 1-2. Accessing Default BizTalk EDI Schemas

There are a large number of EDI schemas that ship with BizTalk Server 2006 R2 and are made available when the EDI components are installed. They can be accessed as follows:

1. Browse to the directory c:\Program Files\Microsoft BizTalk Server 2006\XSD_Schema\EDI.
2. Before schemas are extracted, there is a single file: MicrosoftEdiXSDTemplates.exe. This is a compressed file that contains thousands of EDI schemas. Run this file to extract all of the schemas to the local path. Figure 1-10 shows the location of this file in Windows Explorer.


C:\Program Files\Microsoft BizTalk Server 2006\XSD_Schema\EDI
File Edit View Favorites Tools Help
Back Search Folders
Address C:\Program Files\Microsoft BizTalk Server 2006\XSD_Schema\EDI
Go
Folders
Res
Schema
SDK
Tracking
XSD_Schema
EDI

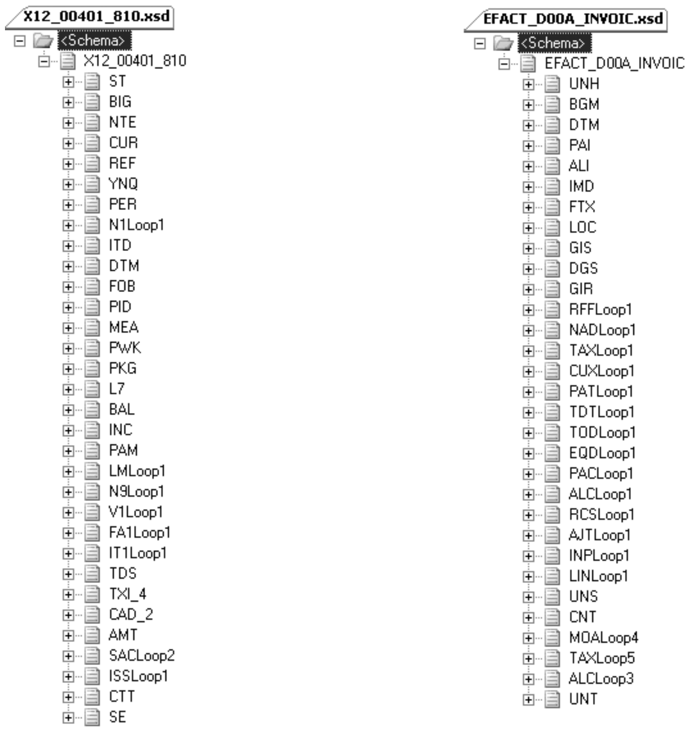
Name	Size	Type	Date Modified
MSSchemas		File Folder	5/22/2007 8:09 AM
Edifact_BatchSchema.xsd	2 KB	XML Schema File	1/19/2007 10:39 AM
Edifact_ServiceSchemaExtension.xsd	3 KB	XML Schema File	1/19/2007 10:39 AM
MicrosoftEdiXSDTemplates.exe	218,722 KB	Application	3/21/2007 12:39 AM
X12_BatchSchema.xsd	2 KB	XML Schema File	1/19/2007 10:39 AM
X12_ServiceSchemaExtension.xsd	6 KB	XML Schema File	1/19/2007 10:39 AM

Figure 1-10. *BizTalk EDI schemas*

X12 and EDIFACT Schemas

There are two EDI standards: X12 and EDIFACT. Briefly stated, the X12 standard is for United States/North American markets, and EDIFACT represents documents exchanged in European/United Nations markets. While the segments and elements within each standard differ greatly from one another, the basic structure of the document itself is similar, with header, detail, and footer segments. All of the exercises throughout this book are interchangeable between the X12 and EDIFACT schemas; the nomenclature and the BizTalk party EDI properties are the only things that differ between the two types. Figure 1-11 shows an instance of an invoice schema for both X12 and EDIFACT, illustrating the difference in naming conventions.

■ **Note** The default EDI schemas provided with BizTalk Server 2006 R2 can be used as is—without modification—for any trading partner. However, in most cases, some amount of modification will be necessary; different partners often have minor changes in their requirements that do not match the default instances. Removing unneeded nodes from the schema also simplifies the schema and mapping and will reduce the overall size of the file.



```
X12_00401_810.xsd
<Schema>
  X12_00401_810
    ST
    BIG
    NTE
    CUR
    REF
    YNQ
    PER
    N1Loop1
    ITD
    DTM
    FOB
    PID
    MEA
    PWK
    PKG
    L7
    BAL
    INC
    PAM
    LMLoop1
    N9Loop1
    V1Loop1
    FA1Loop1
    IT1Loop1
    TDS
    TXI_4
    CAD_2
    AMT
    SACLoop2
    ISSLoop1
    CTT
    SE
```

```
EFACT_D00A_INVOIC.xsd
<Schema>
  EFACT_D00A_INVOIC
    UNH
    BGM
    DTM
    PAI
    ALI
    IMD
    FTX
    LOC
    GIS
    DGS
    GIR
    RFFLoop1
    NADLoop1
    TAXLoop1
    CUXLoop1
    PATLoop1
    TDTLoop1
    TODLoop1
    EQDLoop1
    PACLoop1
    ALCLoop1
    RCSLoop1
    AJTLoop1
    INPLoop1
    LINLoop1
    UNS
    CNT
    MOALoop4
    TAXLoop5
    ALCLoop3
    UNT
```

Figure 1-11. *Comparing X12 to EDIFACT*

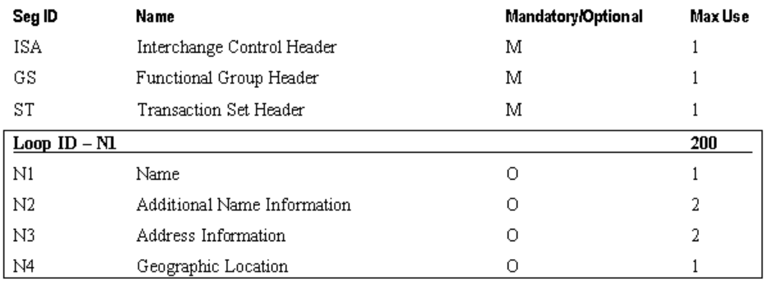
The steps in Exercise 1-3 illustrate how to work with an EDI schema in Visual Studio.

Exercise 1-3. Modifying and Deploying a BizTalk EDI Schema

This exercise walks through the steps necessary to modify a BizTalk EDI 810 schema based on information obtained from an EDI implementation guide. It also demonstrates how to change an enumeration on a specific node. Once the schema is complete, it will be deployed to a specific BizTalk application:

1. Create a new Visual Studio project that will contain one schema for trading partner Company X.
   - **a.** Add an existing schema to the project. Right-click the project and select Add ➤ Existing Item. Browse to c:\Program Files\Microsoft BizTalk Server 2006\ XSD_Schema\EDI\X12\00401. Add the file X12_00401_810.xsd (this is an X12 810 invoice) and click OK.
   - **b.** Rename the schema to X12_00401_810_CompanyX.xsd. In solutions where multiple trading partners may be receiving the same type of document, adding the name of the trading partner to the filename will help eliminate confusion.

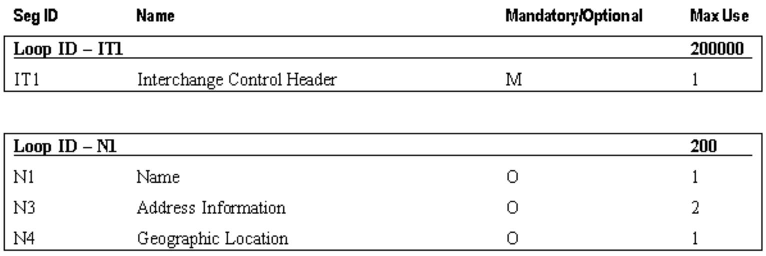
2. Open the schema so that it is visible in Visual Studio and can be edited. The schema will now be modified to more closely match the needs of this trading partner. The implementation guide is shown in Figures 1-12, 1-13, and 1-14. Remove all nodes from the schema that do not appear in the header, detail, or footer section of the implementation guide. Nodes can be removed by right-clicking the node or element in the schema and selecting Delete.



Seg ID	Name	Mandatory/Optional	Max Use
ISA	Interchange Control Header	M	1
GS	Functional Group Header	M	1
ST	Transaction Set Header	M	1
Loop ID – N1			**200**
N1	Name	O	1
N2	Additional Name Information	O	2
N3	Address Information	O	2
N4	Geographic Location	O	1

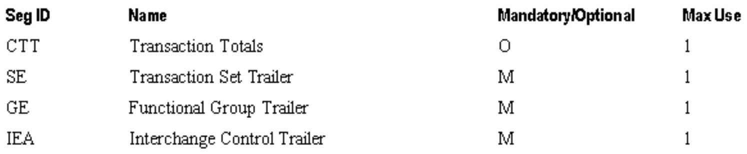
Figure 1-12. *Header in implementation guide*

■**Note** The X12 ISA, GS, GE, and IEA segments never appear in a schema (UNA, UNB, UNG, and UNZ for EDIFACT). These fields are either configured via trading partner settings or are automatically set by the BizTalk EDI pipeline. When creating and modifying a BizTalk EDI schema, ignore these segments.



Seg ID	Name	Mandatory/Optional	Max Use
Loop ID – IT1			**200000**
IT1	Interchange Control Header	M	1
Loop ID – N1			**200**
N1	Name	O	1
N3	Address Information	O	2
N4	Geographic Location	O	1

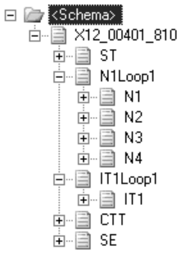
Figure 1-13. *Detail segment in implementation guide*



Seg ID	Name	Mandatory/Optional	Max Use
CTT	Transaction Totals	O	1
SE	Transaction Set Trailer	M	1
GE	Functional Group Trailer	M	1
IEA	Interchange Control Trailer	M	1

Figure 1-14. *Footer (summary) segment in implementation guide*

The final schema in Visual Studio should look like that shown in Figure 1-15.

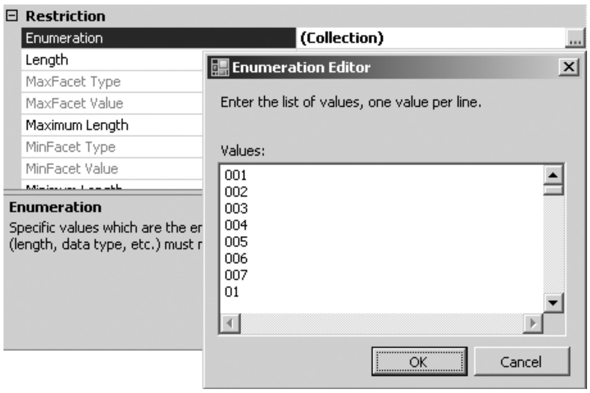


```
<Schema>
  X12_00401_810
    ST
    N1Loop1
      N1
      N2
      N3
      N4
    IT1Loop1
      IT1
    CTT
    SE
```

Figure 1-15. *Modified 810 invoice schema based on implementation guide*

3. This step illustrates how to change the values in an enumeration. Often additional values will need to be added to support a trading partner. To illustrate this, expand the N1Loop1 loop, and click N101. This is an identifier code and has an associated list of valid enumerations.

   a. In the properties for N101, click the ellipsis next to Enumeration, as shown in Figure 1-16.



Restriction
Enumeration (Collection)
Length
MaxFacet Type
MaxFacet Value
Maximum Length
MinFacet Type
MinFacet Value

Enumeration
Specific values which are the er
(length, data type, etc.) must r

Enumeration Editor

Enter the list of values, one value per line.

Values:

```
001
002
003
004
005
006
007
01
```

Figure 1-16. *Enumeration window*

   b. In the window that opens, add an additional value (or remove a value). Only those values that are available in this window will be allowed when validating the data against this schema.

4. Set the schema to deploy to a specific project and set a reference to a strong name key.

   a. Right-click the project and select Properties.

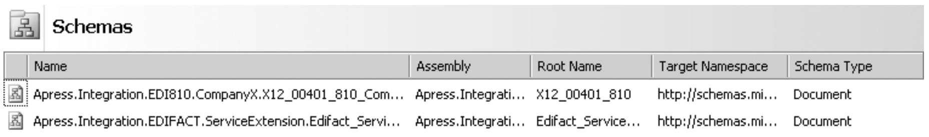
   b. Expand Configuration Properties and click Deployment.

   c. In the Application Name property, enter EDI.Demonstration.Chapter1.

   d. Expand Common Properties and click Assembly. Scroll down and click the Assembly Key File property and browse to a strong name key. One has been created in C:\Apress.Integration\Chapter 1 and is called Apress.Integration.Chapter1.snk.

5. Build and deploy the project by right-clicking the project and selecting Deploy.

6. Once the schema has been deployed, it will be shown in the BizTalk Administration Console under the EDI. Demonstration.Chapter1 application. Click Schemas to view all schemas that are part of the application, as shown in Figure 1-17.



Schemas

Name	Assembly	Root Name	Target Namespace	Schema Type
Apress.Integration.EDI810.CompanyX.X12_00401_810_Com...	Apress.Integrati...	X12_00401_810	http://schemas.mi...	Document
Apress.Integration.EDIFACT.ServiceExtension.Edifact_Servi...	Apress.Integrati...	Edifact_Service...	http://schemas.mi...	Document

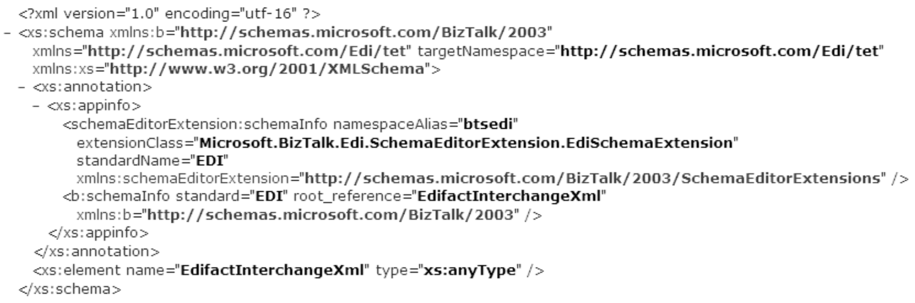
Figure 1-17. *Schema list in the BizTalk administration console*

Batch Schemas

The primary purpose of the EDI batch schemas for X12 and EDIFACT is that they are to be used when validating EDI document instances during schema development. For instance, if a full EDI instance exists, with the header and footer information included, it will not validate directly against a schema of the same type; the X12 and EDIFACT schemas do not support the header and footer segments. However, validating the same document against the batch schema, which is associated with the EDI schema of the same type as the document, will allow the header and footer nodes to be present in the input instance.

Note The batch schema must be added to the project, but only during design time. Trying to deploy a project that contains this project will end in failure; the schema must be removed before deployment. For validation, both the batch schema and the schema being validated against must be included in the project. Also note that no nonschema files, such as orchestration (.odx) files, should be added, as this will result in an error.

The batch schema is intended to be added to a project during the development phase to ease validation. When a document is validated against the batch schema, any other EDI schema that is in the same project will be included in the validation. For instance, if a project contains two schemas, an 810 and an 864, any EDI file instances run against the batch schema will be compared against both the 810 and the 864. An error will be generated if the input EDI instance fails validation against any of the schemas within the project. The EDIFACT batch schema is shown in Figure 1-18. A full demonstration of using a batch schema for instance validation is outlined in Exercise 1-6, in the "Validation and Generation of EDI Documents" section, later in this chapter.



```
  <?xml version="1.0" encoding="utf-16" ?>
- <xs:schema xmlns:b="http://schemas.microsoft.com/BizTalk/2003"
    xmlns="http://schemas.microsoft.com/Edi/tet" targetNamespace="http://schemas.microsoft.com/Edi/tet"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  - <xs:annotation>
    - <xs:appinfo>
        <schemaEditorExtension:schemaInfo namespaceAlias="btsedi"
          extensionClass="Microsoft.BizTalk.Edi.SchemaEditorExtension.EdiSchemaExtension"
          standardName="EDI"
          xmlns:schemaEditorExtension="http://schemas.microsoft.com/BizTalk/2003/SchemaEditorExtensions" />
        <b:schemaInfo standard="EDI" root_reference="EdifactInterchangeXml"
          xmlns:b="http://schemas.microsoft.com/BizTalk/2003" />
      </xs:appinfo>
    </xs:annotation>
    <xs:element name="EdifactInterchangeXml" type="xs:anyType" />
  </xs:schema>
```

Figure 1-18. *The EDIFACT batch schema*

Service Schema Extensions

The service schemas contain those elements within the EDIFACT and X12 schemas that can have additional enumerations specified. Occasionally it may be necessary to add additional enumerations to support trading partner requirements. Trading partner configuration is covered in great detail in the next chapter, but while discussing schemas it is important to understand that several of the header, context, and footer segments are set as EDI properties on the BizTalk party, and these values are added on during document processing in the EDI send pipeline. The values for a number of these fields are controlled by the service schemas and are never part of the transaction set schema. Exercise 1-4 demonstrates how to work with the service extension schema and illustrates its role in party configuration.

Exercise 1-4. Modifying and Deploying a Service Extension Schema

The steps in this exercise demonstrate how to add additional options in the drop-down fields for several elements in a service schema. This exercise works with an EDIFACT service schema and demonstrates how to add a new value to the UNB2.2 element in the trading partner EDI properties:

1. Ensure that the BizTalk Administration Console is closed. The values in the drop-down will only be refreshed when the Administration Console is first opened.
2. Open Visual Studio to create a new project that will contain the modified service schema and will allow the schema to be easily deployed. Add the EDIFACT service schema to the project. Right-click the project and select Add ➤ Existing Item. Browse to c:\Program Files\Microsoft BizTalk Server 2006\XSD_Schema\EDI. Add the file Edifact_ServiceSchemaExtension.xsd and click OK. Ensure that the project has a strong name key associated with it so that it can be deployed.
3. Open the schema in Visual Studio. There are two elements present that contain enumerations. They are UNB2.2 and UNB3.2. Add an additional value to the enumeration of UNB2.2:
   a. Right-click UNB2.2 and select Properties.
   b. In the Properties window, find the Enumeration property and click the ellipsis next to the (Collection) value.
   c. In the window that opens, enter the value DEMO. There will already be a value SGG. Make sure that the values are on different lines, as shown in Figure 1-19. Additional values can be entered.

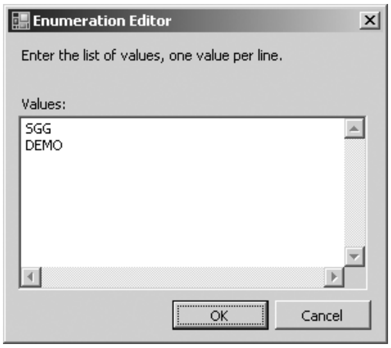
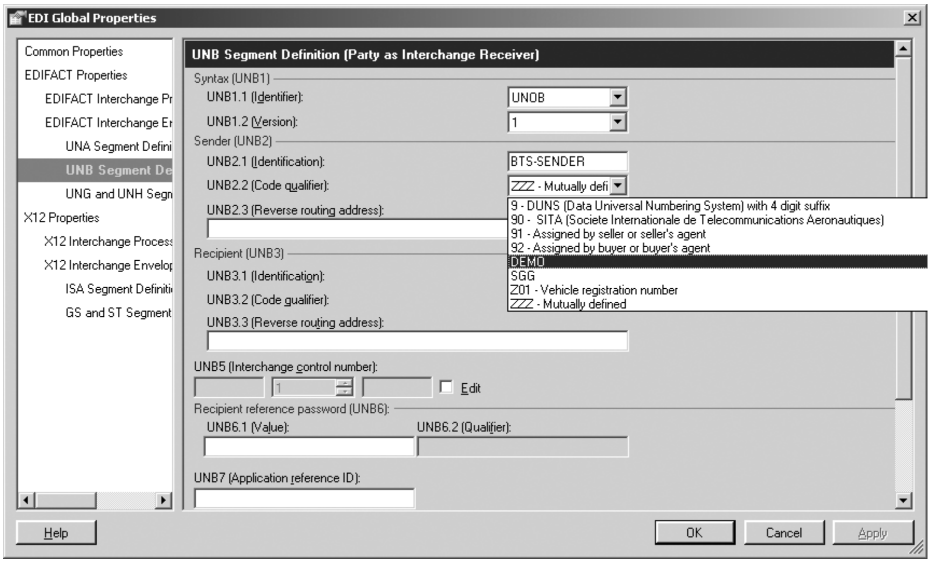


Figure 1-19. *Property enumeration values*

   d. Click OK.

4. Deploy the schema by right-clicking the project and selecting Deploy. This will make the new enumerations available, in addition to the default service schema that is part of the BizTalk EDI engine.
5. Open BizTalk Administration Console and click the Parties folder. The new enumeration(s) will be available both on the global EDI properties and all individual parties.
   a. Right-click the Parties folder and select EDI Global Parties.
   b. Click the UNB Segment Definition.
   c. View the list of enumerations in the drop-down next to UNB2.2. The new enumerations will appear as shown in Figure 1-20.



UNB Segment Definition (Party as Interchange Receiver)

Syntax (UNB1)
UNB1.1 (Identifier): UNOB
UNB1.2 (Version): 1

Sender (UNB2)
UNB2.1 (Identification): BTS-SENDER
UNB2.2 (Code qualifier): ZZZ - Mutually defi
UNB2.3 (Reverse routing address):

9 - DUNS (Data Universal Numbering System) with 4 digit suffix
90 - SITA (Societe Internationale de Telecommunications Aeronautiques)
91 - Assigned by seller or seller's agent
92 - Assigned by buyer or buyer's agent
DEMO
SGG
Z01 - Vehicle registration number
ZZZ - Mutually defined

Recipient (UNB3)
UNB3.1 (Identification):
UNB3.2 (Code qualifier):
UNB3.3 (Reverse routing address):

UNB5 (Interchange control number): 1 ☐ Edit

Recipient reference password (UNB6):
UNB6.1 (Value): UNB6.2 (Qualifier):

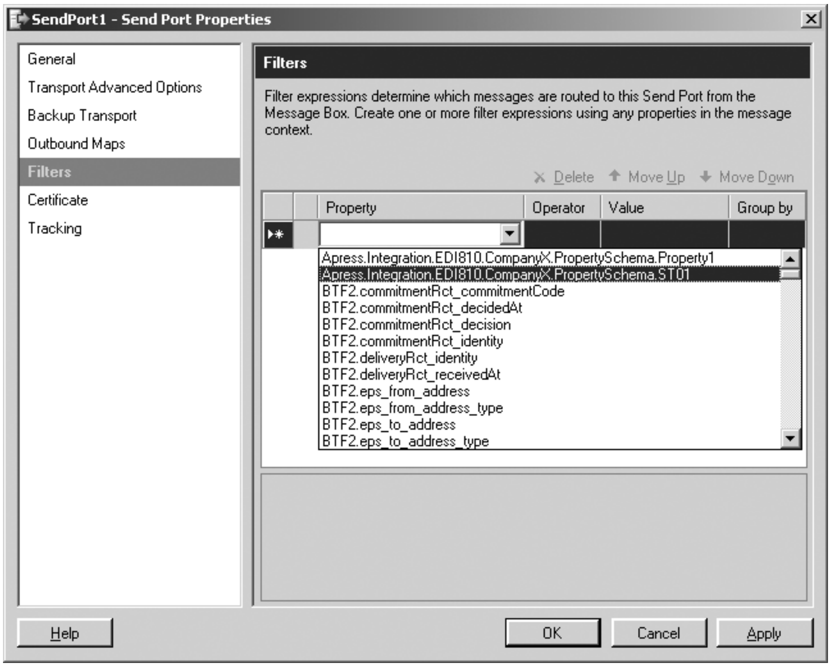
UNB7 (Application reference ID):

Figure 1-20. *EDI Global Properties dialog with modified UNB2.2 enumerations*

Promoting Fields

The promotion of properties on a schema allows the data within the promoted field to be accessible to external components, such as send ports and orchestrations. Promotion comes in two flavors: *distinguished* and *property*. Distinguished fields are those that are available within a certain context, such as an expression shape within an orchestration, and are accessible for determining conditional flow, much like properties on an object. Distinguished fields are not available on send ports and cannot be used for document routing on the MessageBox. A property field is one that is accessible by all BizTalk components and can be used for routing within the MessageBox and within orchestrations.

There is extra overhead with property fields, as they require a property schema (which can be automatically generated); distinguished fields do not require any additional schema. The property schema allows fields to be accessible to external components and is deployed along with the standard EDI schema. To illustrate this, work through the steps in Exercise 1-5 and then deploy the project (this includes the 810 schema and the property field). Once the schemas have been deployed, they will be accessible as a filter on a send port, as shown in Figure 1-21.


SendPort1 - Send Port Properties

General
Transport Advanced Options
Backup Transport
Outbound Maps
Filters
Certificate
Tracking

Filters

Filter expressions determine which messages are routed to this Send Port from the Message Box. Create one or more filter expressions using any properties in the message context.

Property	Operator	Value	Group by

Apress.Integration.EDI810.CompanyX.PropertySchema.Property1
Apress.Integration.EDI810.CompanyX.PropertySchema.ST01
BTF2.commitmentRct_commitmentCode
BTF2.commitmentRct_decidedAt
BTF2.commitmentRct_decision
BTF2.commitmentRct_identity
BTF2.deliveryRct_identity
BTF2.deliveryRct_receivedAt
BTF2.eps_from_address
BTF2.eps_from_address_type
BTF2.eps_to_address
BTF2.eps_to_address_type

Figure 1-21. *Send port filter showing promoted property as an option*

Exercise 1-5. Promoting Properties

This exercise illustrates how to promote both distinguished and property fields in a schema:

1. In Visual Studio, open the project created in Exercise 1-3.
2. In the schema editor, right-click the ST01 node and select Promote ➤ Quick Promotion. In the dialog box that asks whether to create a property schema, click OK. This will promote the field as a property field, create a property schema, and make the field available to external components once it has been deployed. Figure 1-22 shows a schema with the ST01 field promoted.

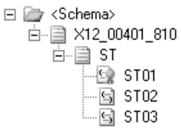
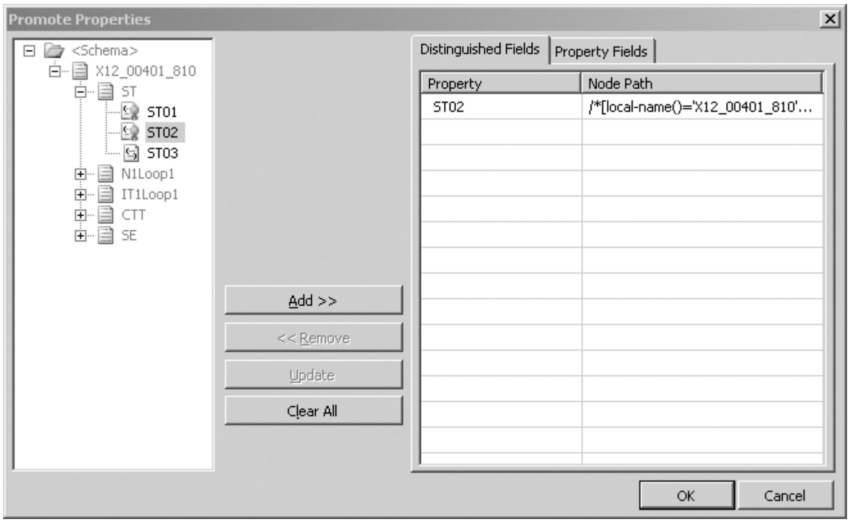


Figure 1-22. *Promoted property on a schema*

3. Right-click the ST01 field again and select Promote ➤ Show Promotions. There are two tabs: Distinguished Fields and Property Fields. The Property Fields tab will show the ST01 field that was just promoted.
4. Add a distinguished field by clicking the Distinguished Field tab, selecting ST02, and clicking Add. This would make the ST02 field available in an orchestration (if one is used). Figure 1-23 shows what the promotion of this field looks like. Click OK when complete.



Property	Node Path
ST02	/*[local-name()='X12_00401_810'...

Figure 1-23. *Promote Properties window*

5. Double-click the newly created property schema (PropertySchema.xsd). Notice that there are two fields: one is Property1 and the other is ST01. When this schema is automatically generated, the Property1 field is included (it is always included when creating a schema via Quick Promotion). It can be removed by right-clicking and selecting Delete.

■ **Note** This exercise uses ST01 for demonstration purposes. By default, this field is already promoted in the EDI engine and is available without setting the field as a promoted property on the schema.

Validation and Generation of EDI Documents

Once a schema has been created, it will always be necessary to validate individual EDI instances, ensuring that the schema and document instances match requirements for the trading partner. It may even be helpful to generate an instance of the EDI document in its native flat file format. There are a variety of items that a schema can validate, including the following:

- **Existence of all expected nodes (complete hierarchy)**: Schema validation can ensure that the segments are in the correct order, that all expected segments exist, and that all of the mandatory data elements within a segment are present.
- **Proper formatting and naming**: Ensuring that all of the segments are named properly, that they have the expected data element separators, and that the segment terminators are correct are part of validation.
- **Content type and length**: The data within each data element in a segment is controlled by the schema. Items such as the expected length and whether the data is in the correct format (int vs. string) can be validated.

Throughout the development and test phases of any EDI implementation, file instances need to be validated. The steps shown in Exercise 1-6 outline the simplest way to validate a document. Once mapping becomes part of the picture, more complex document validation techniques may need to be incorporated (this is explained in Chapter 5).

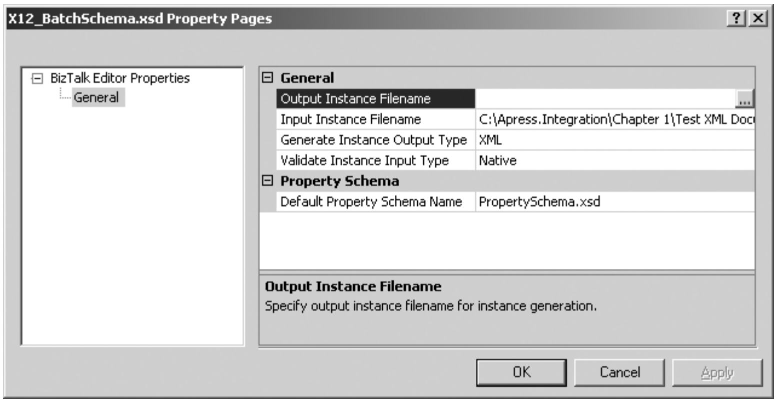
Exercise 1-6. Validating an EDI Instance Against a Schema

The following steps introduce the validation of an instance of an EDI document with the header and footer segments included:

1. In Visual Studio, open the project created in Exercise 1-3.
2. Adding the batch schema to this project will allow full EDI documents, with header and footer segments, to be validated and created. To add this schema, take these steps:
   a. Right-click the project and select Add ➤ Existing Items. Browse to the directory c:\Program Files\Microsoft BizTalk Server 2006\XSD_Schema\EDI.
   b. Select the schema X12_BatchSchema.xsd and click OK.
3. This exercise demonstrates an invalid instance first. Right-click the newly added schema and select Properties. Point the Input Instance Filename property to the Invalid-Input.txt file in the directory C:\Apress.Integration\Chapter 1\Test Documents. The content of this file is shown in Listing 1-1. Figure 1-24 shows the appropriate settings for the different properties.

Listing 1-1. *Invalid EDI Instance*

```
ISA*00*          *00*          *01*COMPX789       *ZZ*APRESS1234
*070607*1555*U*00401*000000025*0*T*>~
GS*IN*COMPX789*APRESS1234*20070607*1555*25*X*004010~
ST*810*0025~
N1*SF*COMPANY X~
N3*P.O. BOX 12345~
N4*ANYTOWN*OH*45678~
IT1*1*0.528*TS*1.13**VP*1AGHA1223221~
REF*INVALID
CTT*1~
SE*100*0025~
GE*1*25~
IEA*1*000000025~
```

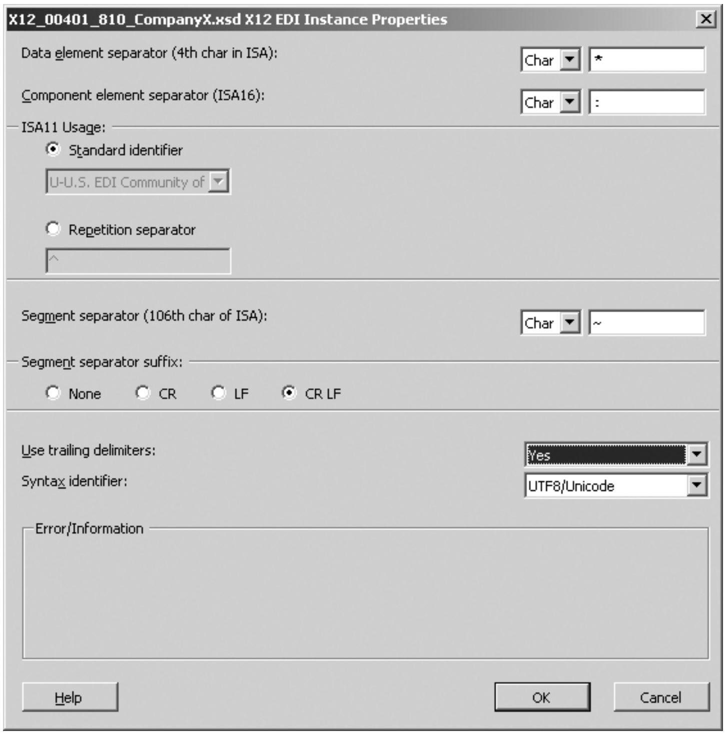


General	
Output Instance Filename	
Input Instance Filename	C:\Apress.Integration\Chapter 1\Test XML Doc
Generate Instance Output Type	XML
Validate Instance Input Type	Native
Property Schema	
Default Property Schema Name	PropertySchema.xsd

Output Instance Filename
Specify output instance filename for instance generation.

Figure 1-24. *Setting properties on the batch schema for validation*

4. Right-click the batch schema in Visual Studio and select Validate Instance.
5. In the EDI Instance Properties window (shown in Figure 1-25), accept all of the defaults and click OK. The EDI Instance Properties window allows different formats of the EDI document to be validated. For instance, if the document contains all of the segments, but their line end separator is a > instead of a line feed (CR LF), the value can be set in this window.


X12_00401_810_CompanyX.xsd X12 EDI Instance Properties

Data element separator (4th char in ISA): Char *

Component element separator (ISA16): Char :

ISA11 Usage:

(●) Standard identifier: U-U.S. EDI Community of

( ) Repetition separator: ^

Segment separator (106th char of ISA): Char ~

Segment separator suffix: ( ) None ( ) CR ( ) LF (●) CR LF

Use trailing delimiters: Yes

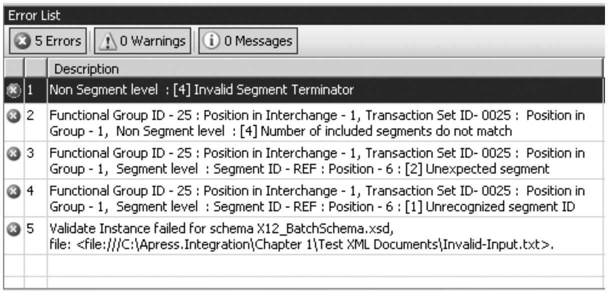
Syntax identifier: UTF8/Unicode

Error/Information

Figure 1-25. *EDI Instance Properties window*

A number of errors will be generated, as shown in Figure 1-26. These errors are as follows:

• **"Number of included segments do not match"**: This refers to the value in SE01, which is 100. The value should be an accurate count of all of the detail lines, which in this case is 7.


Error List

5 Errors | 0 Warnings | 0 Messages

	Description
1	Non Segment level : [4] Invalid Segment Terminator
2	Functional Group ID - 25 : Position in Interchange - 1, Transaction Set ID- 0025 : Position in Group - 1, Non Segment level : [4] Number of included segments do not match
3	Functional Group ID - 25 : Position in Interchange - 1, Transaction Set ID- 0025 : Position in Group - 1, Segment level : Segment ID - REF : Position - 6 : [2] Unexpected segment
4	Functional Group ID - 25 : Position in Interchange - 1, Transaction Set ID- 0025 : Position in Group - 1, Segment level : Segment ID - REF : Position - 6 : [1] Unrecognized segment ID
5	Validate Instance failed for schema X12_BatchSchema.xsd, file: <file:///C:\Apress.Integration\Chapter 1\Test XML Documents\Invalid-Input.txt>.

Figure 1-26. *Error list from invalid instance*

- **"Unexpected segment"**: The REF node is not included in the schema and is not part of the EDI implementation guide for this trading partner. The node should not exist in the sample EDI instance.

There are several other errors shown in this document, but they overlap those already discussed. Once the items in the previous two exceptions are fixed, the other errors will disappear.

6. Now validate a valid instance of the same EDI document. Right-click the batch schema and select Properties. This time set the input filename to the Valid-Input.txt file in the directory C:\Apress.Integration\Chapter 1\Test Documents. The content of this file is shown in Listing 1-2. The REF segment that caused the errors has been removed.

Listing 1-2. *Valid Version of the EDI Instance*

```
ISA*00*          *00*          *01*COMPX789       *ZZ*APRESS1234
*070607*1555*U*00401*000000025*0*T*>~
GS*IN*COMPX789*APRESS1234*20070607*1555*25*X*004010~
ST*810*0025~
N1*SF*COMPANY X~
N3*P.O. BOX 12345~
N4*ANYTOWN*OH*45678~
IT1*1*0.528*TS*1.13**VP*1AGHA1223221~
CTT*1~
SE*7*0025~
GE*1*25~
IEA*1*000000025~
```

The previous exercise demonstrates the validation of an instance of an EDI document. It is also possible to generate a new file instance, using the same basic approach, as shown in Exercise 1-7. Generating a document can be useful when no existing version of the EDI document exists in its native format (text file) or when trying to generate test data for development purposes.

Exercise 1-7. Generating an EDI Instance

It is useful to be able to create sample instances of an EDI document from a schema, and this can be easily accomplished by using the batch schema, as shown in these steps:

1. In Visual Studio, open the project created in Exercise 1-6.
2. Right-click the batch schema and select Properties.
3. Set the Output Instance File Name to a path on the local machine, and give the output file a name (such as output.txt).
4. Set the Generate Instance Output Type to Native.
5. Click OK to save the settings.
6. Right-click the batch schema again and select Generate Instance. This will create a dummy instance of the schema in EDI format.

Final Discussion

As the foundation to all BizTalk implementations, development of the schema can be a complex and time-consuming process. However, using the EDI implementation guide for a trading partner and understanding how to transfer the content of that guide to the development of an EDI schema will eliminate much of the complexity. Using property promotion to make fields in the schema available to external BizTalk components, such as ports and orchestrations, can aid in the simplification of the solution as a whole. Once the schema is fully defined, knowing how to validate existing EDI documents will ensure that the schema developed will match the requirements of the trading partner.

With an understanding of EDI schemas and how to work with them in BizTalk Server 2006 R2, the next step is to work through the configuration of the trading partner. Schema development and trading partner configuration are often intermixed since an EDI document consists of header, detail, and footer segments, many of which are defined in the trading partner properties (rather than the schema). Some of these segments are present in the schemas, some are present in the trading partner configuration, and some cross over between the two. Without knowledge of trading partner configuration, an EDI schema is incomplete, and without knowledge of EDI schemas, trading partner configuration cannot be completed.