**EDI Validation**

Validation is an important step when processing EDI files. It ensures a more robust automated process by detecting and rejecting EDI files with anomalies that could break a translation program and interrupt production.

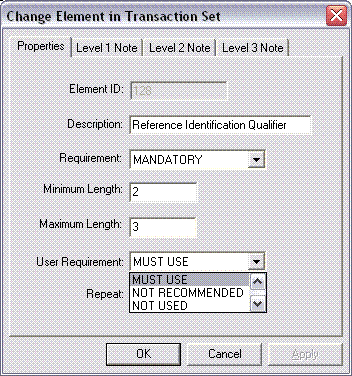
The Framework EDI is a powerful component that gives users almost unlimited ability to validate an EDI file.  It uses a Standard Exchange Format (SEF) file to obtain the EDI implementation guideline, which it would use to match against an EDI file it is validating.  Discrepancies between the SEF file and EDI file would then be reported as errors.  The eFileManager Utility is an example of an application that uses the Framework EDI component for validating EDI documents.

Framework EDI's validation checks for the following:

***Position***  
Verifies the positioning of data segments and elements that they are in the correct order.

***Requirement***  
Checks to see if segments specified as *Mandatory* in the SEF file exist in the EDI document. Segments that are specified as *Optional* do not have to exist in the document.

***User Requirement***  
Data segments or data elements specified *Must Use* must exist in the EDI file; those specified as *Not Used* must not exist, while those marked *Used* may or may not exist in the EDI file. The "User Requirement" has precedence over the "Requirement" property.



***Minimum and Maximum Length***  
The length of the data in a data element should be within the minimum and maximum length range.

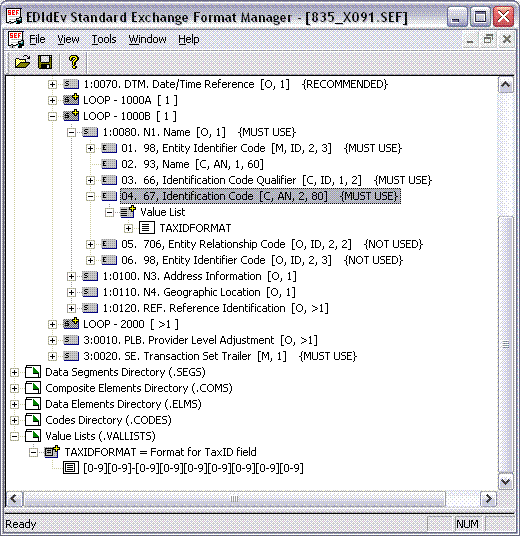
***Maximum Usage***  
An instance of a data segment in an EDI file can not exist more times than the value specified.

***Loop Repeat***  
An instance of a loop or group of segments in an EDI file cannot be repeated more times than the value specified.

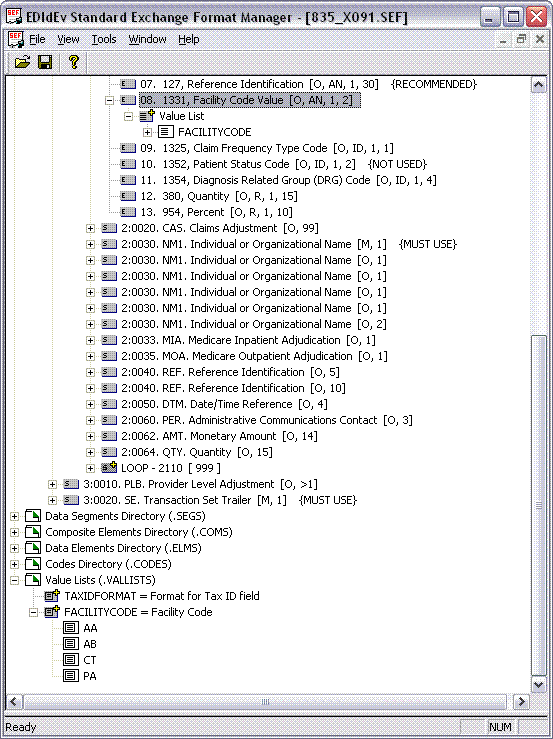
***Syntax Rules***  
*Syntax Rules* in data segments are enforced by the Framework EDI component. They define the relationship of their data elements.  An example of a *syntax rule* in a data segment is - if either data element(1) or data element(2) is present, then data element(3) must be present.

***Code List***  
Framework EDI reads the *Code List* of data elements in a SEF file to determine if a code used in a data element of an EDI file is valid.

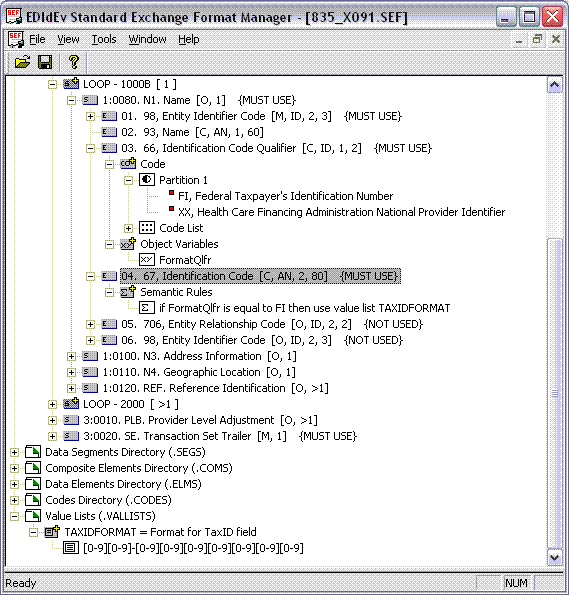
***Value Reference* and *Value List***  *(available only in FREDI version 5.0.2005.403 and after )*  
Framework EDI reads the *Value Reference* and *Value List* of data elements in a SEF file to make sure that their values in the EDI file follow the value expression defined, or is included in the list of values defined.  For example, a data element in a SEF file with a Value Reference expression "[0-9][0-9][0-9][0-9][0-9]" means that the format of the value of the same data element in the EDI file can only contain numeric characters and must be 5 characters long.



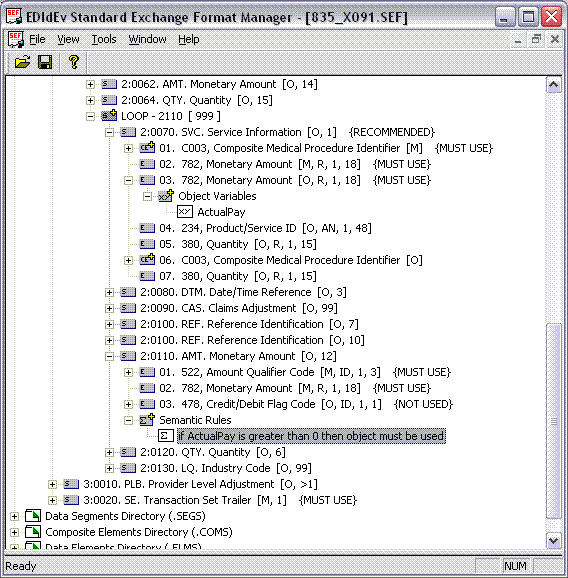
An example of the use of *Value List* is to have a list of values that are allowable for a specific data element.  This is similar to a *code list*, but a *Value List* is more flexible in that a conditional statement (*semantic references)* can call it.



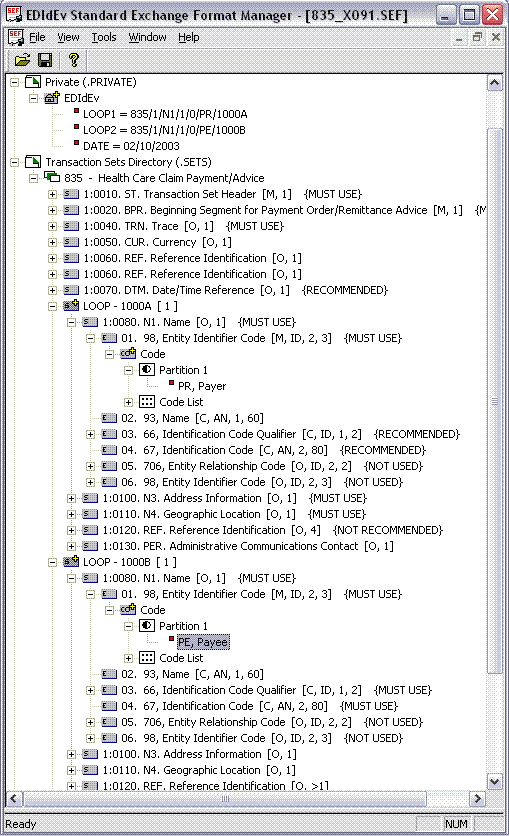
***Semantic References***  *(available only in FREDI version 5.0.2005.403 and after )*  
Framework EDI reads the *Semantic References* of data segments and data elements in a SEF File to determine if a conditional value, or existence of a data segment or data element in an EDI file is valid.  An example of a *semantic reference* is - if data element(3) equals "FI" then data element(4) use value reference TAXIDFORMAT.



Another example of a semantic reference is - if data element(3) value in SVC segment is greater than 0, then data segment AMT must be used.



***Triggers***  
*Triggers* in SEF files (under PRIVATE EDIdEv) identify the schema for each instance of data segments and loops, thus enabling Framework EDI to validate instances (identified by their qualifier) of a data segment or a loop in an EDI file.  For example, an EDI file can have a loop with two instances having different requirements: one loop instance having the Payer information and having data segments N3 and N4 as *mandatory*; while the second loop instance having the Payee information, but having data segments N3 and N4 as *optional*.



By using SEF files to obtain the EDI implementation guideline, Framework EDI can almost have no limit to what it can validate in an EDI file.  Basically, if a rule can be specified in the SEF file, then Framework EDI will enforce the same rule in an EDI file.  SEF files are text files, which can be edited with any text editor, or with our SEF Manager utility.  For more details about editing a SEF file with the SEF Manager, please read [Creating an Implementation Guideline](http://www.edidev.com/articles/CreatingIG/fredi_create_ig.html).

Below is a sample Visual Basic program that demonstrates how the Framework EDI component uses a SEF file to validate an EDI file.

[[View and run similar VB.NET source code](https://secure.edidev.net/edidev-ca/samples/vbWebAck/)]

Dim oEdiDoc As Fredi.ediDocument  
Dim oWarnings As Fredi.ediWarnings  
Dim oWarning As Fredi.ediWarning  
Dim nWarningCount As Integer  
Dim i As Integer  
  
Set oEdiDoc = New Fredi.ediDocument  
  
'Load SEF file  
oEdiDoc.ImportSchema sPath & "837\_X098.SEF", Schema\_Standard\_Exchange\_Format  
  
'Load EDI file  
oEdiDoc.LoadEdi sPath & "837OUTPUT.X12"  
  
'Check if FREDI detected any errors  
Set oWarnings = oEdiDoc.GetWarnings  
nWarningCount = oWarnings.Count  
  
'If error count is greater than 0 then errors were found  
If nWarningCount > 0 Then  
  
   'Display errors in a List box  
   For i = 1 To oWarnings.Count  
      Set oWarning = oWarnings.Warning(i)  
      List1.AddItem oWarning.Code & " " & oWarning.Description  
   Next  
Else  
   'No errors found  
End If  
  
Set oEdiDoc = Nothing  
MsgBox "Done"