EDIDEV KNOWLEDGE LETTER
June 30, 2018
http://www.edidev.com

This is part three of a multipart article about managing and processing incoming EDI files from several trading partners. (The solution suggested here may not be suitable for all business scenarios.)

The Inbound Process - Part 3

==============================

The first step of the inbound process is to validate the EDI file. This is the action of checking the EDI file to see if it conforms to your specifications. This process is an essential step because if an EDI file does not meet your specifications some data may not get read correctly when translating it. In FREDI, validation occurs automatically when the EDI file is being read.

If an EDI file fails the validation it should be rejected; and if it passes, it should be accepted. Your trading partner should be notified of this status by sending them an acknowledgment file. In X12, it's either with a 997 Functional Acknowledgment, or a 999 Implementation Acknowledgment. In UN/EDIFACT, it is with a CONTRL file. All inbound EDI files (except for acknowledgment files) should be acknowledged.

With the previous versions of FREDI, you had to separate the acknowledgment files from the other EDI files to avoid acknowledging them. With the current version of FREDI, you don't have to because FREDI will ignore acknowledgment files during the acknowledgment process, and will therefore not create acknowledgment files for them. For example, it will not create a 997 for an inbound 997 EDI file even if it was embedded with other messages in the same EDI document.

Creating acknowledgment files with the FREDI component is as easy as enabling their property, and then just reading through all the EDI file's segments. For example:

```
        oEdiDoc.CursorType = DocumentCursorTypeConstants.Cursor_ForwardOnly;
        oAck = oEdiDoc.GetAcknowledgment();
        oAck.EnableFunctionalAcknowledgment = true;

        oSegment = oEdiDoc.FirstDataSegment;
        while (oSegment != null)
        {
                oSegment = oSegment.Next();
        }

        oAck.Save(sPath + "997_Output.X12");
```

Because we are already traversing through all the segments in an EDI file when validating and creating an acknowledgement file, it is very tempting to add the logic for translating the EDI file in the same "while" loop. And why not? It can save time, right?

Well, it depends on the situation.

The big issue with combining the process of acknowledging and translating the EDI file is that if the validation fails, the translation would have already occurred, which you would then have to rollback. This is messy and would also likely require manual intervention.

A likely scenario where one may combine the processes in one "while" loop to save time is when your inbound EDI files are small and few, and they are from a trading partner who you have been exchanging EDI files with for a while that there is an unlikely chance of rejecting their EDI files.

But, if your inbound EDI files are large and complex like health care claims or if you're dealing with hundreds of files, then it's better to have the validation and acknowledgement in one process, while keeping the translation in a separate process. You can save time by running the processes concurrently in separate threads or applications so that while one application is validating and acknowledging EDI files, the other can be translating them.

(to be continued)

EDIDEV
"We make EDI fun!"