**EDI Transmission**

(You must have EDIdEv Framework EDI installed on your machine to run the example programs on this page successfully.  [Click here to download an evaluation copy](http://www.edidev.com/eval_Fredi.html).)

The Internet has opened up many ways for companies to send and receive EDI files. Below are short descriptions of some them and how they are supported by the Framework EDI component.

Dial-Up

Dial-up transmission was the main method of sending and receiving EDI files before the high-speed Internet became popular.  Communications would traditionally be over dedicated lines directly between trading partners or through a VAN (Value-Added-Network).  But not only was this transmission slow, it was cumbersome and expensive, because several modems and lines were required to support multiple trading partners.  (Please note that the Framework EDI component has no dial-up functionalities.)

SMTP or email

Email over the Internet provides for a less expensive and simple means of sending and receiving EDI files.  However, EDI files are less secure if sent by email over the Internet, and the size of an account's mailbox limits the size of EDI files one can send.  Also, email generally requires a user to retrieve an attached message, which is not in accordance to the EDI definition of having no human intermediary.

Below is an example of how one can use the Framework EDI component to send an EDI file by SMTP:

'Send by SMTP  
Set oTransport = oTransports.CreateTransport  
  
' Set required parameters for email  
oTransport.SetSMTP  
Set oSmtpCfg = oTransport.GetSmtpCfg  
oSmtpCfg.ServerName = ""smtp.maildomain.com""  
oSmtpCfg.User = "useracountid"  
oSmtpCfg.Password = "accountpassword"  
oSmtpCfg.AddTo ("UserTo")  
oSmtpCfg.From = "UserFrom@domain.com"  
oSmtpCfg.FromDisplay = "UserFirst UserLastname"  
oSmtpCfg.Subject = "Subject Title"  
oSmtpCfg.MessageText = "Message Body. EDI file attached"  
oTransport.Send sEdiFile

FTP

With FTP communication servers becoming affordable, and easier to set up and operate, the FTP protocol has become one of the popular ways of sending and receiving files.  Trading partners can easily set up their own FTP server and have their EDI files uploaded directly to them.  However, with this protocol, files can only be sent/received into folders (directories) where they wait to get polled for processing, which can be taxing to a system depending on the polling interval.

Below is an example of how one can send and receive EDI files by FTP with the Framework EDI component:

'Send by FTP  
Set oTransport = oTransports.CreateTransport  
oTransport.SetFTP  
oTransport.Address = "www.domain.name"  
oTransport.Password = "password"  
oTransport.User = "username"  
oTransport.TargetPath = "/inboundfolder/"  
oTransport.Send sEdiFile  
  
'Receive by FTP  
Set oTransport = oTransports.CreateTransport  
oTransport.SetFTP   
oTransport.Address = "www.domain.name"  
oTransport.Password = "password"  
oTransport.User = "username"  
oTransport.TargetPath = "/inboundfolder/"  
oTransport.Receive sEdiFile

* [TransportFTP](http://www.edidev.com/samples/transmit/TransportFTP.zip) - Download an example program in VB6 that sends an EDI file by FTP.
* [ReceiveFTP](http://www.edidev.com/samples/transmit/ReceiveFTP.zip) - Download an example program in VB6 that gets an EDI file by FTP.

HTTP or HTTPS

With almost every company having a Web Site and a Web Server, the HTTP server is the most available of all communication servers, but yet only a handful of companies take advantage of this powerful and already available resource for sending and receiving EDI files.  Just like the FTP server, the HTTP server is inexpensive and is easy to set up; but the major advantage HTTP has over FTP, is that the HTTP protocol sends/receives files directly to an application or "destination application".  This makes it possible to have the *destination application* run a process immediately once a file is received.  One such process can be to trigger an alert or even to acknowledge the received file

Below is an example of how one can send an EDI file by HTTP with the Framework EDI component:

'Set up HTTP  
Set oTransport = oTransports.CreateTransport  
oTransport.SetHTTPS  ' or SetHTTP  
oTransport.InternetUrl = "https://website.net:12345/folder/destinationapplication.aspx/"  
  
Set oHttpCfg = oTransport.GetHttpCfg  
oHttpCfg.SendVerb = "POST"  
oHttpCfg.EnableProcessResponse = True  
  
'Send an EDI file by HTTP  
oTransport.SendFile("c:\testfolder\EdiFile.X12")  
  
'Get response from server  
sHttpResponse = oTransport.LastResponseInfo

Below are example programs of a HTPP client and a *destination application* to demonstrate how one can send and receive EDI files by HTTPS, and get an acknowledgment back immediately.

* [HttpClient.zip](http://www.edidev.com/samples/transmit/HttpClient.zip) - A VB .NET (VS2010 64-bit) example program that sends an EDI X12 file by HTTPS to a *destination application* on a server, and synchronously receives a TA1 acknowledgment.
* [EdiServer.zip](http://www.edidev.com/samples/transmit/EdiServer.zip) - An ASP VB .NET (VS2010 64-bit) example of a *destination application*.  This program receives EDI files, and then generates and sends a TA1 file acknowledgment back to the sender synchronously .

EDIINT AS2

EDIINT AS2 is basically a more sophisticated version of the above HTTP file exchange, but more complicated because of the extra standards it introduces over the HTTP protocol responses, and message format wrapped over the EDI standard.  The AS2 protocol is not specific to EDI files only, so another acknowledgment called the MDN is used to acknowledge receipt of AS2 files.  The extra layers in AS2 that are not related to EDI makes this protocol quite difficult to implement, and can become expensive if certification and support maintenance are required.

Below is an ASP .NET example program (with source code) of an **AS2 Server** using the Framework EDI .NET component to receive, decrypt and read AS2 messages. It can send back MDN synchronously or asynchronously.  [Click here](http://www.edidev.com/articles/As2Server/As2ServerExample.html) for a detail explanation of the AS2 Server (C#) example program.

* [vb\_As2ServerNet](http://www.edidev.com/samples/transmit/vb_As2ServerNet.zip) - ASP VB.NET [view code](http://www.edidev.com/samples/transmit/vbAs2ServerSourcecode.htm)
* [cSharp\_As2ServerNet](http://www.edidev.com/samples/transmit/cSharp_As2ServerNet.zip) - ASP C# [view code](http://www.edidev.com/samples/transmit/cSharpAs2ServerSourcecode.htm)

Below is an example program of an **AS2 Client** that creates, encrypts, signs and sends AS2 messages to an AS2 server:

* [vb\_As2ClientDemoNet](http://www.edidev.com/samples/transmit/vb_As2ClientDemoNet.zip) - VB.NET [view code](http://www.edidev.com/samples/transmit/vbAs2ClientSourcecode.htm)
* [cSharp\_As2Client](http://www.edidev.com/samples/transmit/cSharp_As2Client.zip) - C# [view code](http://www.edidev.com/samples/transmit/cSharpAs2ClientSourcecode.htm)

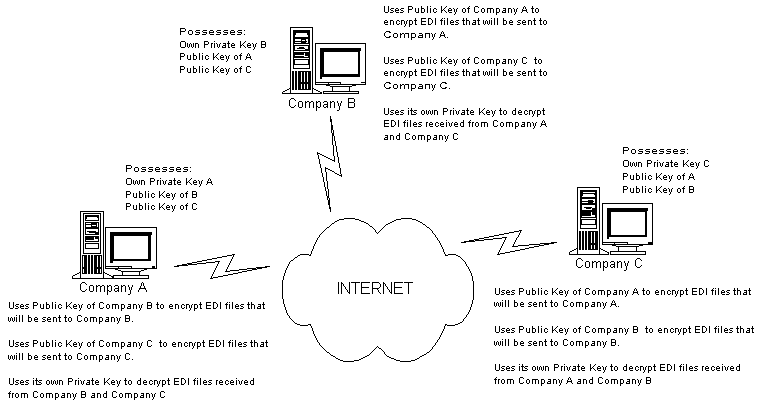
Below is a C# example program that shows how to create and read the same AS2 message:

* [As2CreateRead](http://www.edidev.com/samples/transmit/As2CreateRead.zip)

*AS2 Made Easy*  
For an article and example programs that show how to create an AS2 client in steps, [click here](http://www.edidev.com/articles/As2Step/creating_As2Client_Step.html)

Before exchanging secured AS2 messages, trading companies have to generate their own public and private keys.  Technically, these keys are held in a key container in the O/S.  The private key remains in the key container, but the public key can be exported to a certificate file so that it can be exchanged between trading companies.  Public keys are used for encrypting each other's EDI files, while the private key is used for decrypting.  Below are sample programs in VB6 to show how one can use the component to create and import security certificates.

* [CreateCert](http://www.edidev.com/samples/transmit/CreateCert.zip) - Creates a key container, certificate store and a certificate file.
* [ImportCert](http://www.edidev.com/samples/transmit/ImportCert.zip) - Imports a security certificate.



*Q.  Is it possible to send and receive AS2 messages with only the Framework EDI Professional License (without using the Enterprise License)?*   
A.  Yes, it is possible to send and receive AS2 messages with the Framework EDI Professional License.  However, it does not have AS2 development support because utilities such as the eSecurity Console and ISAPI Extension Console, which would facilitate AS2 development, are not available in the Professional License.

*Q.  Can the security certificates created by Framework EDI be used in a production environment?*   
A.  We don't recommend it because these security certificates are not trusted, or have not been specifically assigned to you, which means that it could be possible for someone else to create the same certificate you may have created for yourself.  Creating your own certificates (not trusted) should be limited to testing environments only.

*Q.  Is the Framework EDI AS2 functionality certified by Drummond?*   
A.  No, we have not sought for Drummond certification.  The functionalities in our component have not been limited or restricted to AS2 making it possible to create other message standards exchanged over HTTP (such as NAESB) with the Framework EDI component.