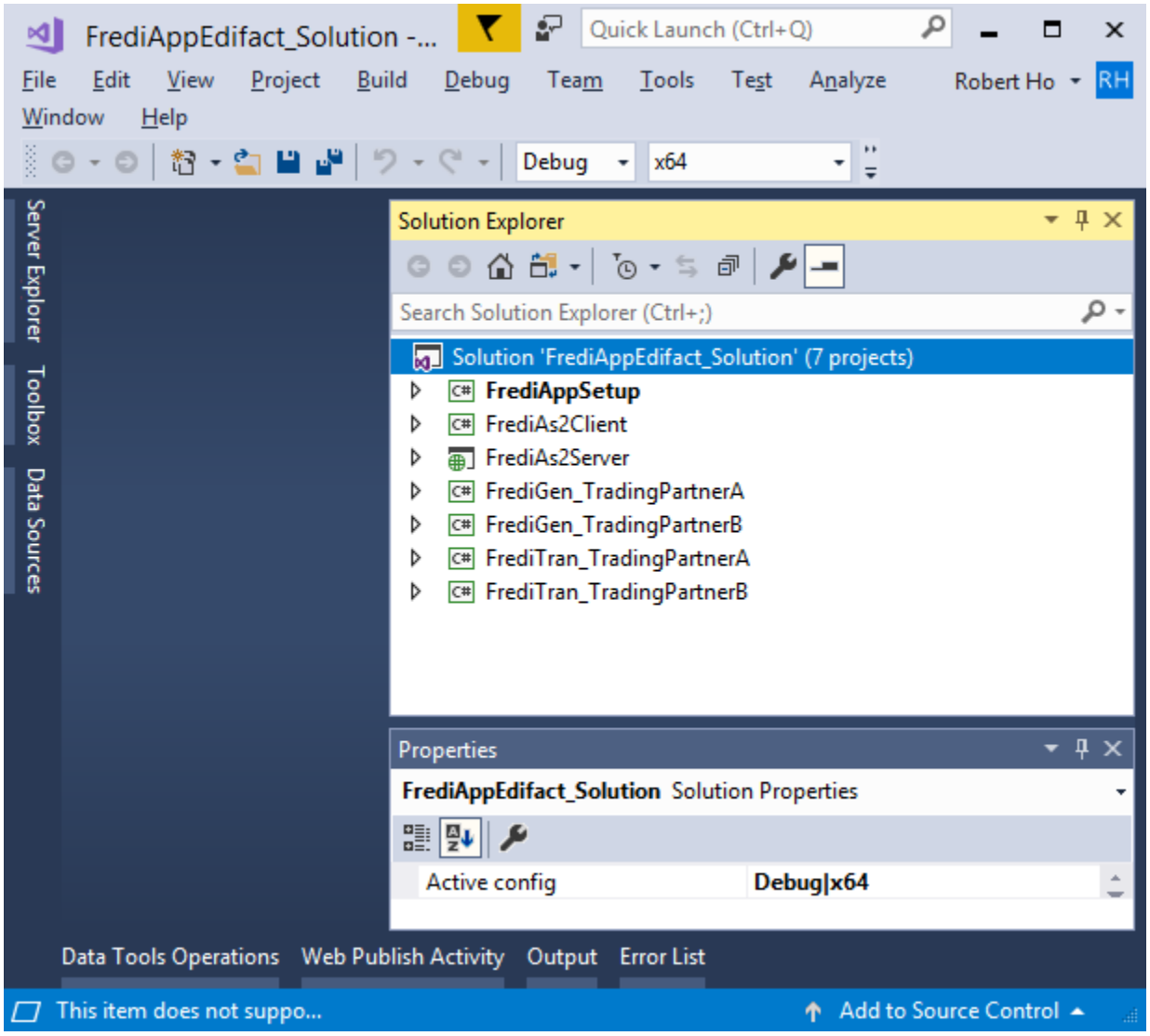
The FrediAppEdifact is a Visual Studio C# solution of an end-to-end EDI program. Its purpose is to give programmers a framework of a working EDI solution that they can build on or simply get inspiration from to create their own comprehensive EDI solution.

To start the project, launch your Visual Studio, and then Open Project *FrediAppEdifact_Solution.sln*

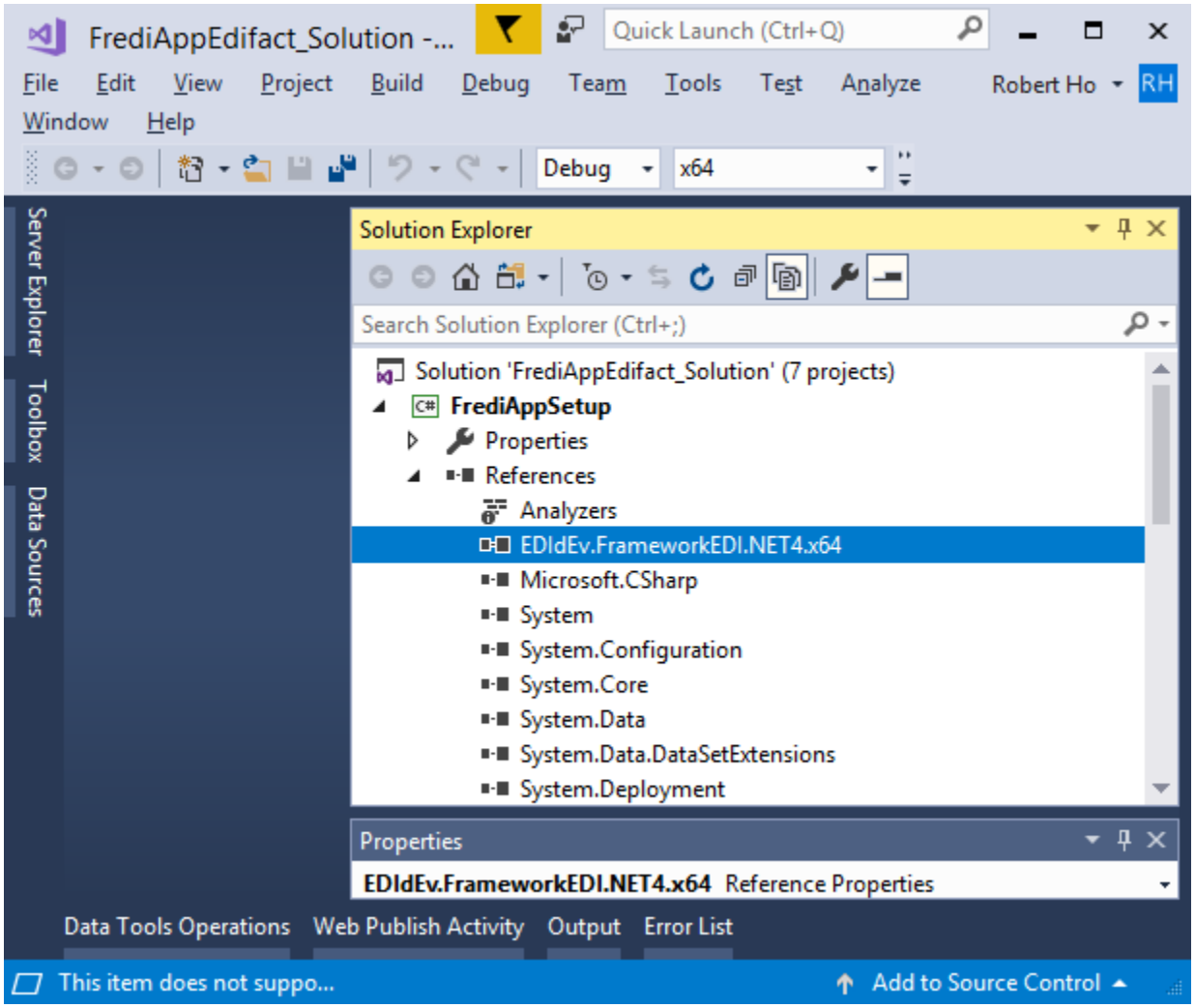


The FrediAppEdifact solution has the following projects:

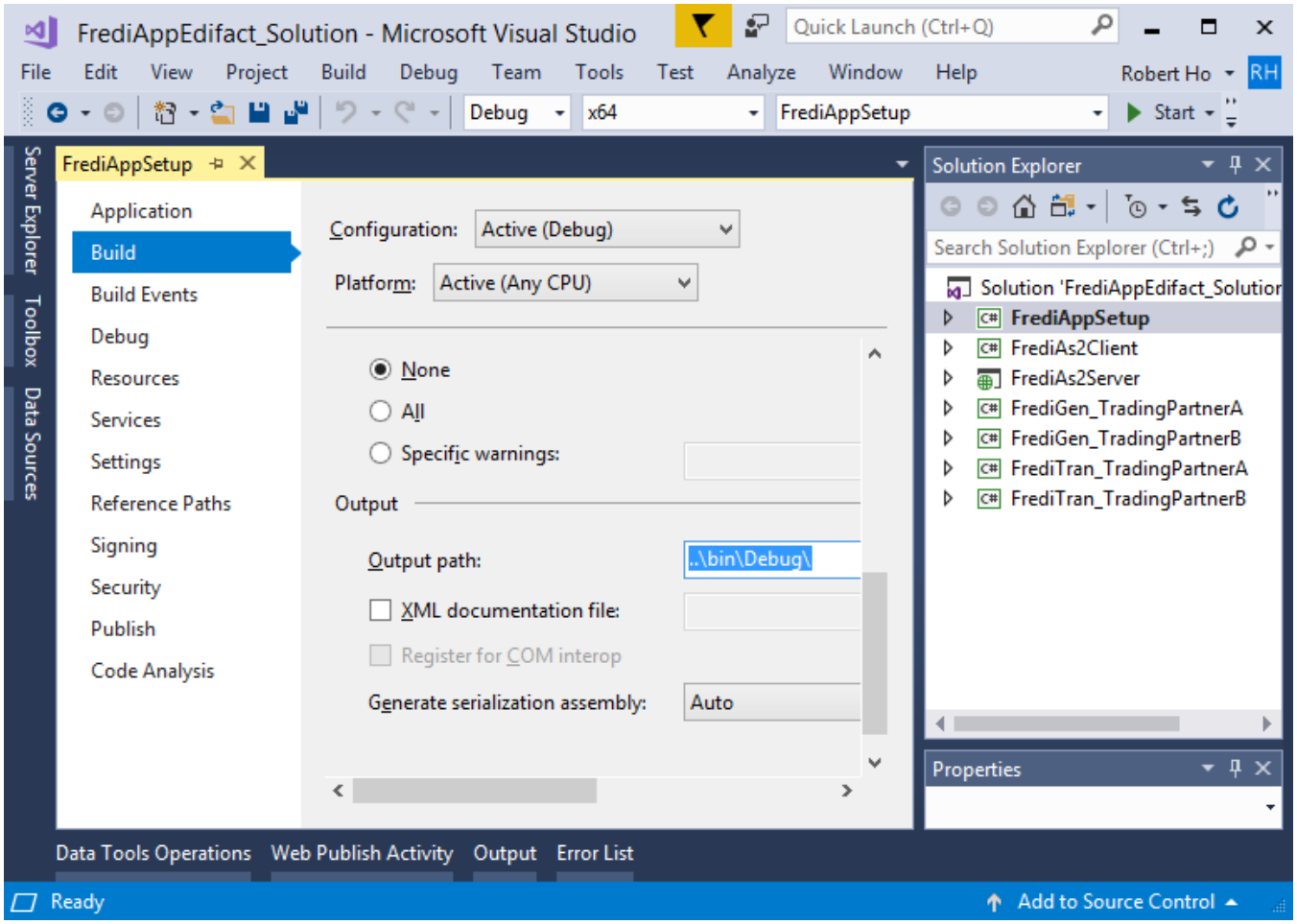
- FrediAppSetup – to configure your company's profile as well as your trading partners' information.

- FrediAs2Client – sends EDI files by AS2
- FrediA2Server – receives AS2 messages
- FrediGen_TradingPartnerA – Generates EDI files for one trading partner
- FrediGen_TradingPartnerB – Generates EDI files for another trading partner
- FrediTran_TradingPartnerA – Acknowledges and translates EDI files for one trading partner
- FrediTran_TradingPartnerB – Acknowledges and translates EDI files for another trading partner

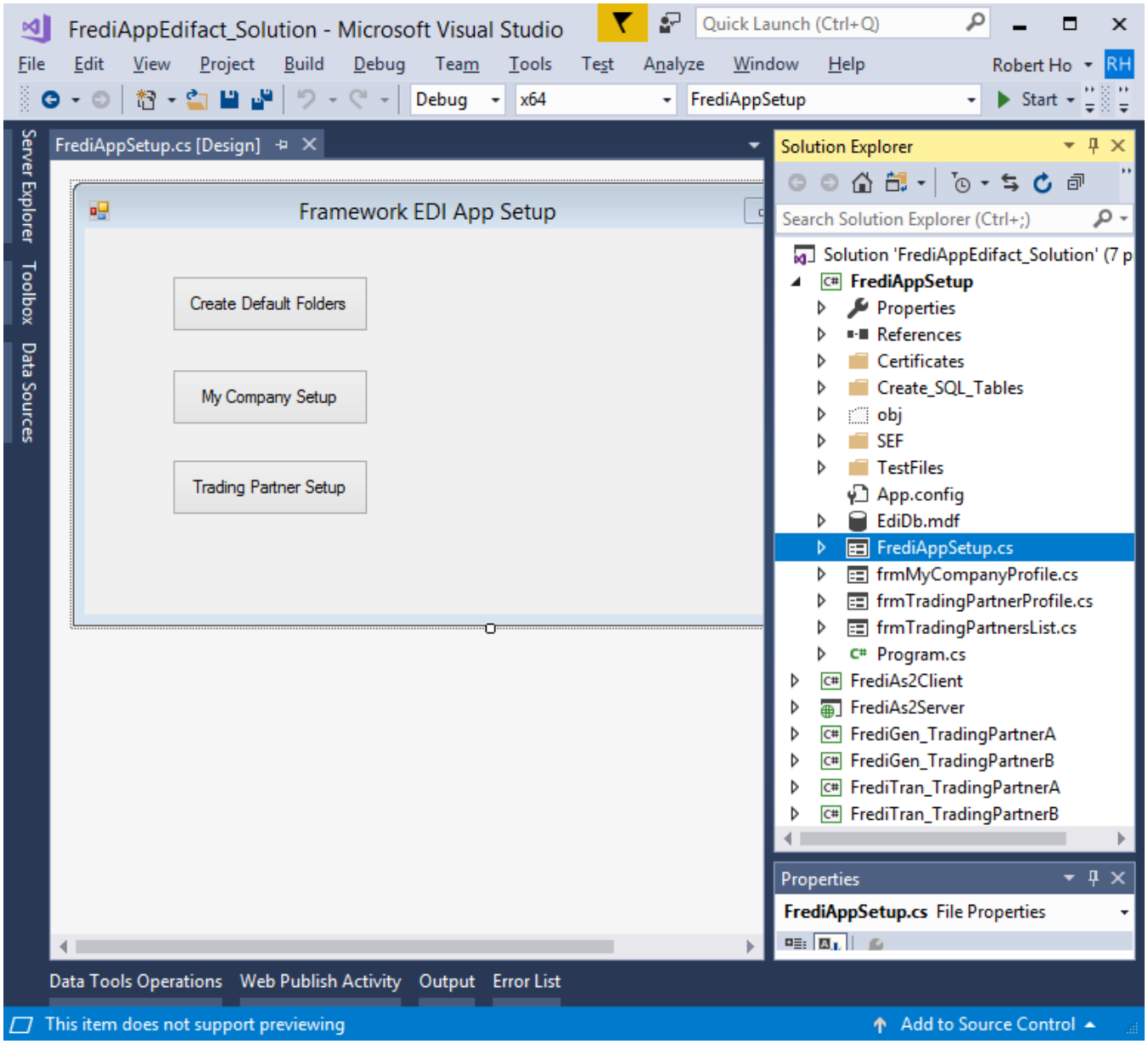
All five projects must reference the *EDIdEv.FrameworkEDI.NET4.x64.dll*. (You may have to remove the existing reference, and then add it back again because your FREDI version may be different from what the solution has referenced.)



The *debug* and *release* folders for all the projects (except for the frediAs2Server) all point to the same *bin* folder …*FrediAppEdifact_Solution**bin*\



The FrediAppSetup Project

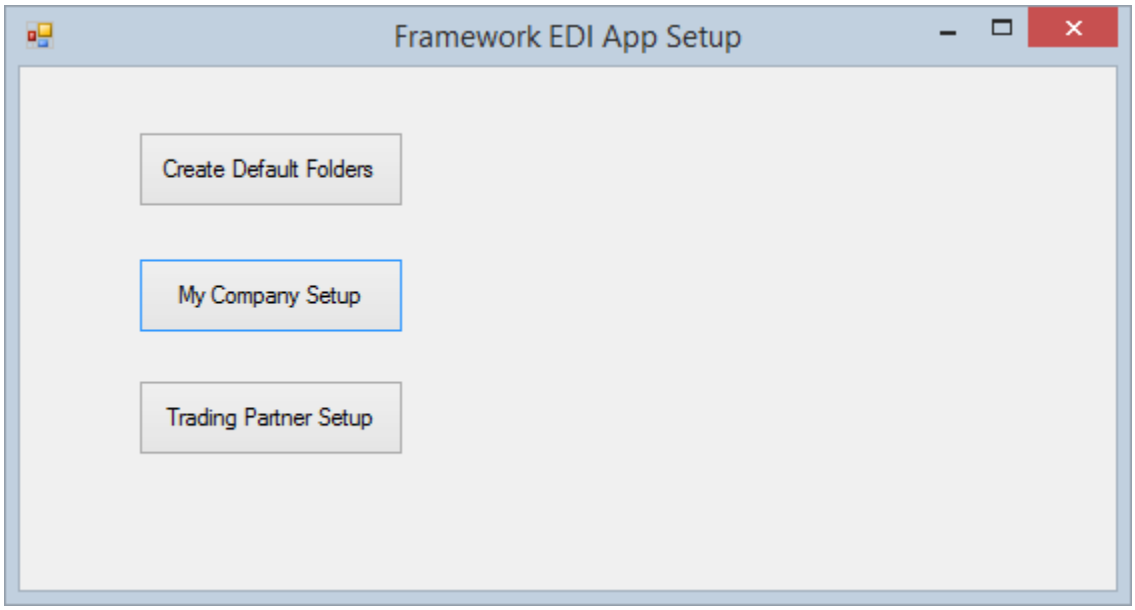


The FrediAppSetup program creates the default folders where the EDI files are saved at each step of the process. It is also where you can enter your company's profile as well as your trading partners' EDI and AS2 information.

Included with the project are sample EDI files in the *TestFile* folder; evaluation SEF files in the *SEF* folder; a test certificate in the *Certificates* folder; and SQL scripts for creating the database tables in the *Create_SQL_Tables* folder.

Also, included is a local SQL database named *EdiDb.mdf* that ties all five projects together. It will be used as an intermediate database and is the go-between storage for your data in your production database and EDI files.

The FrediAppSetup form

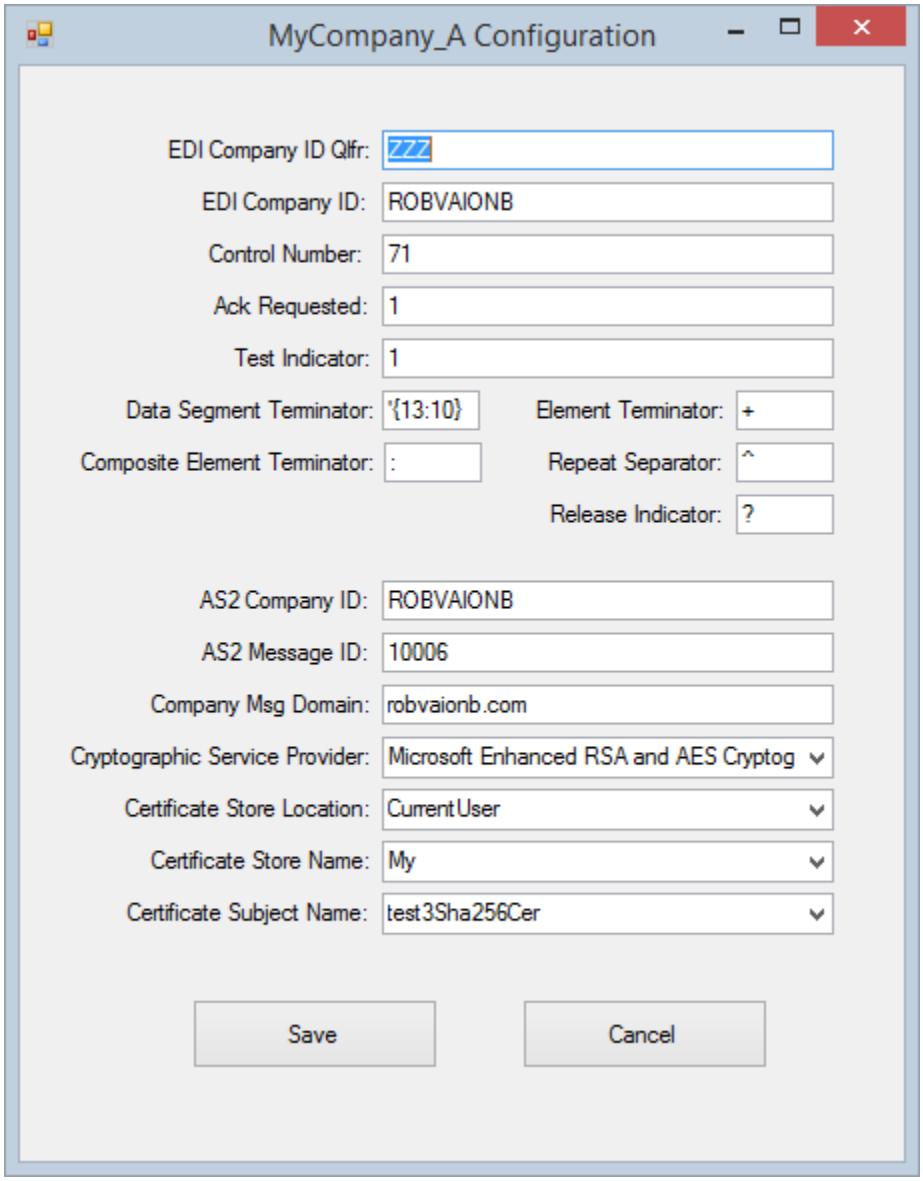


The FrediAppSetup form consists of three buttons:

The *Create Default Folders* button creates the following folders:

- AS2_OUTPUT – This is where copies of AS2 files that were sent, and MDN files that were received are located.
- EDI_CONTRL – This is where completed inbound CONTRL EDI files (after being processed) are stored.
- EDI_DONE – Inbound EDI files (other than CONTRL) that were accepted and translated are saved in this folder.
- EDI_INBOUND – Inbound EDI files that were just received by the AS2 Server and ready to get validated and acknowledged are in this folder.
- EDI_OUTBOUND – Outbound EDI files ready to get sent wait in this folder.
- EDI_REJECTED – Inbound EDI files that failed validation and rejected are moved into this folder.
- EDI_SENT – Outbound EDI files that were successfully sent are saved in this folder.
- EDI_SENT_FAILED – Outbound EDI files that were not successfully sent are kept here.

The *My Company Setup* button brings up the company configuration form:

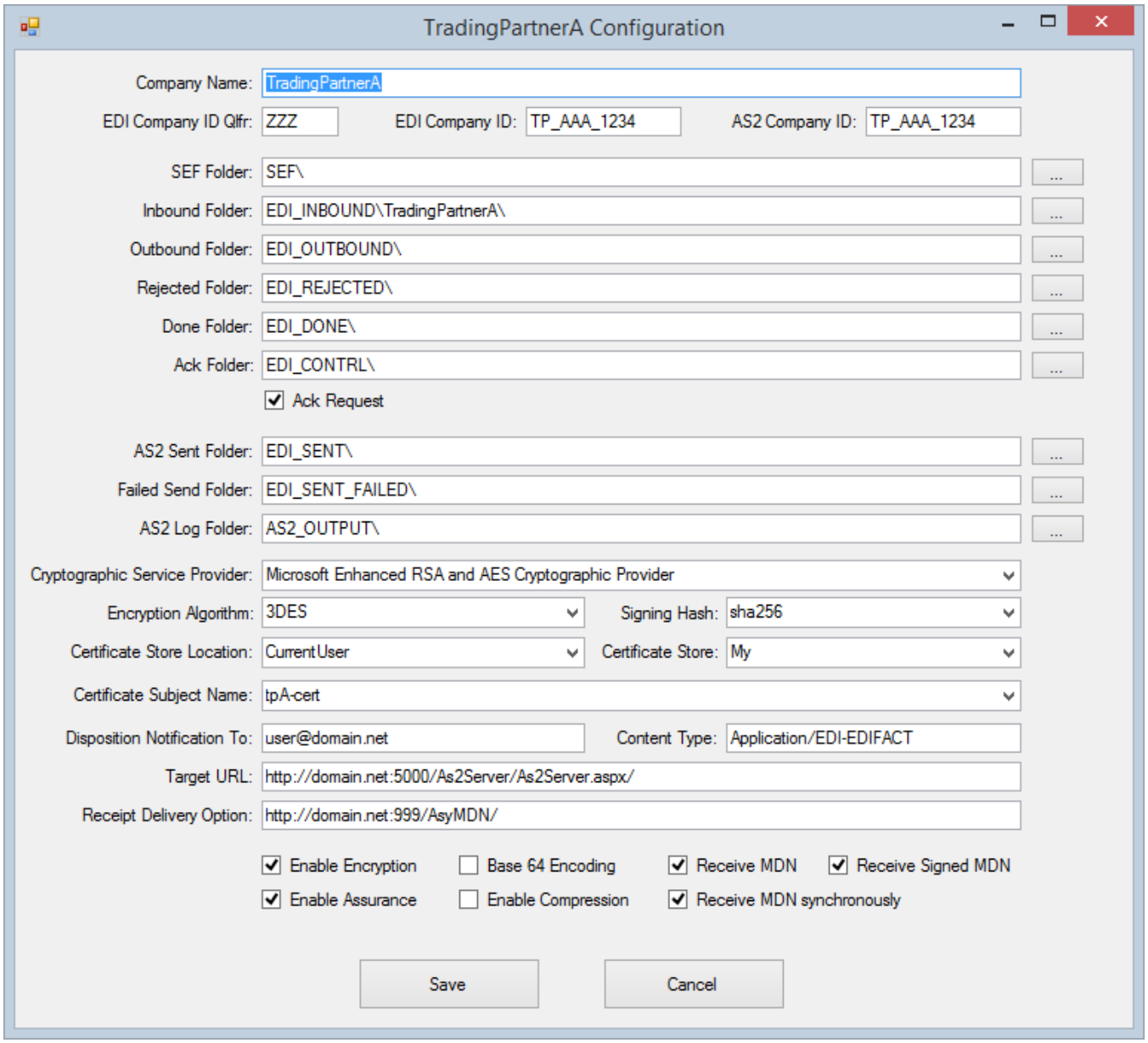


- EDI Company ID Qualifier – This is the value in the *Interchange ID Qualifier* (UNB02-02 or UNB03-02) of your EDI file.
- EDI Company ID – This is the value in the *Interchange Sender ID* ((UNB02-01) of your outbound EDI files. EDI files you receive will have this value in the *Interchange Recipient ID* (UNB03-01).
- Control Number – This is the next available number that will be used as an *Interchange Control Reference* (UNB05) for the next generated outbound EDI file. This

value is incremented by one whenever an EDI file is generated by the *FrediGen* program as well as when a 997-acknowledgment is created by the *FrediTran* program.

- Ack Requested – Enter 1 if you want your trading partner to send you a CONTRL; otherwise leave the field empty.
- Test Indicator – If you are generating a test EDI file, enter a "1"; otherwise leave the field empty.
- AS2 Company ID – This is your company ID that will be displayed in the *AS2-From* header of the AS2 message. Typically, this is the same value as your EDI Company ID.
- AS2 Message ID – This is the next available number and in conjunction with the *Company Msg Domain* will be used in the *Message-ID* header for the next AS2 message sent. This is used to reference your AS2 message.
- Company Msg Domain – This is your company domain, which will be concatenated with the *AS2 message ID* to create a globally unique *Message-ID* for your AS2 message.
- Certificate Subject Name – Select your security certificate name that will be used to decrypt and sign AS2 messages. This will be the certificate that you will send to your trading partner so that they can encrypt the AS2 message they send you as well as authenticate the messages they receive from you.
- Cryptographic Service Provider – Choose a CSP that will support the encryption and hashing algorithm that you and your trading partner are using.
- Certificate Store Location – Select the registry location where your certificate was installed.
- Certificate Store Name – Select the name of the Certificate Store where your certificate was installed. (See eSecurityConsole utility).

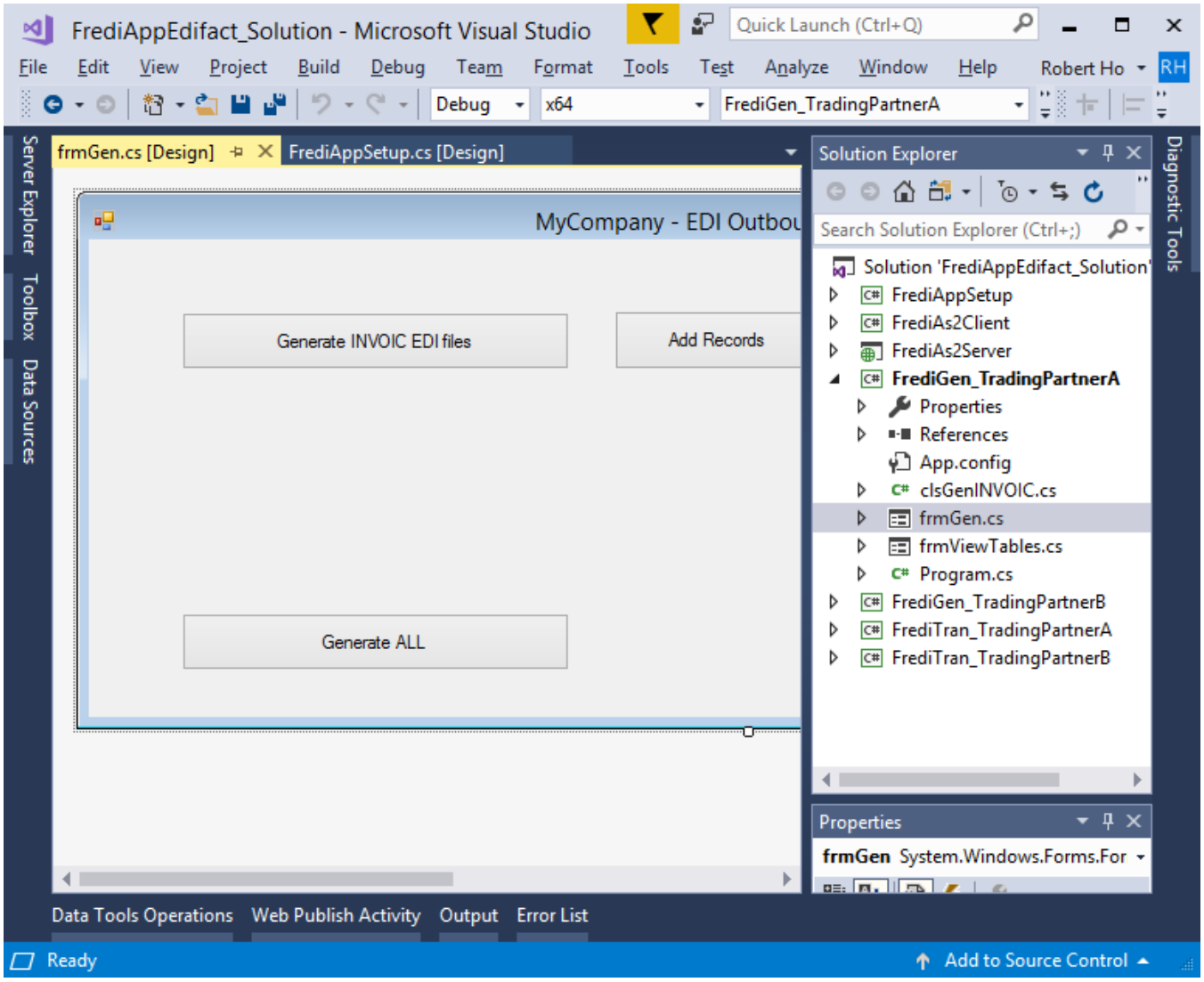
The *Trading Partner Setup* button takes you to the following form:



- Company Name – The name of the trading partner. This should be the same value as what was entered in the *TradingPartner* key field in the app.config file of the FrediGen and FrediTran programs.
- EDI Company ID Qlfr and Company ID – These are values in the *Interchange* segment position UNB02-02 and UNB02-01 respectively of inbound EDI files; and in position UNB03-02 and UNB03-01 of outbound EDI files.
- (See create Default Folder for meaning of folder fields.)

- Ack Request - Check the box if this trading partner wants a CONTRL acknowledgment sent to them.
- Cryptographic Service Provider – Choose a CSP that will support the encryption and hashing algorithm that you and this trading partner are using.
- Certificate Store Location – Select the name of the registry location where this trading partner's certificate (public key) was installed.
- Certificate Store Name – Select the name of the *Certificate Store* where this trading partner's certificate was installed. (See eSecurityConsole utility).
- Certificate Subject Name – Select the trading partner's security certificate name that you will use to encrypt the AS2 message you will send them. This will be the same certificate to authenticate the messages you receive from them.
- Encryption Algorithm – Choose the encryption algorithm that your trading partner is using.
- Signing Hash – Choose the hashing algorithm that this trading partner is using.
- Disposition Notification To – Enter your contact email should this trading partner want to contact you regarding the AS2 message.
- Target URL – Enter the trading partner's AS2 Server URL address. This is where you will be sending the AS2 messages for them. (Make sure to end it with a forward slash "/".)
- Receipt Delivery Option – If asynchronous MDN is requested, enter the address of where the MDN should be sent.

The FrediGen_TradingPartnerA Project

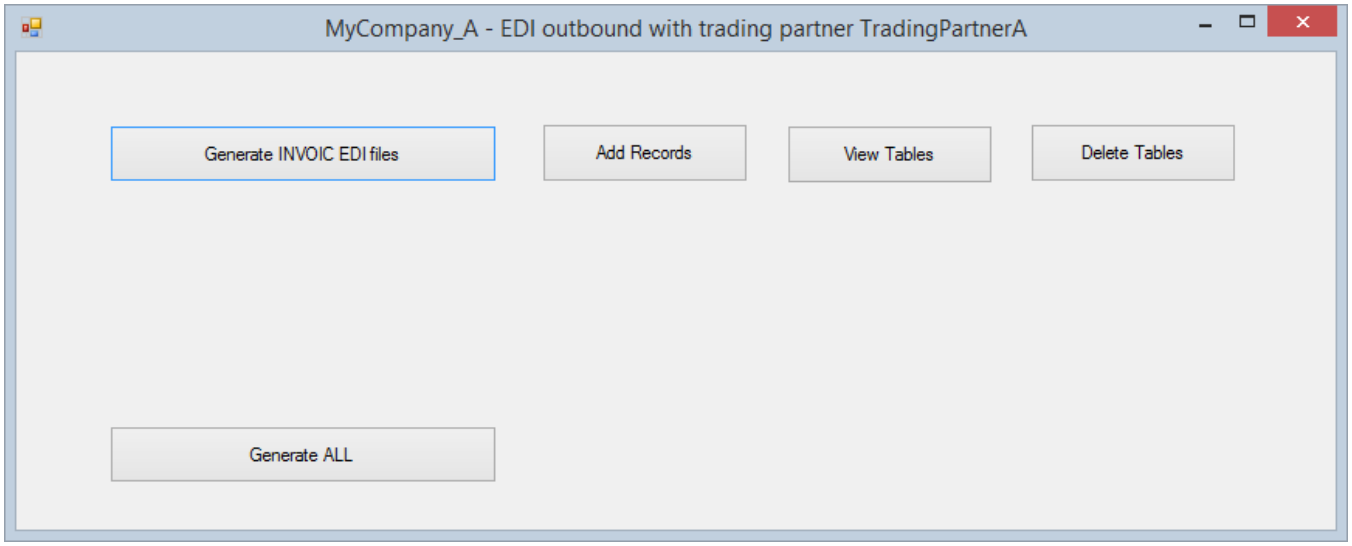


This program processes the EDI outbound for one trading partner. For this specific example, the trading partner is named "TradingPartnerA". If you have another trading partner, then you would need to create another project for them. You could copy this project and rename it with the other trading partner's name. For example, if your other trading partner's name was "TradingPartnerB", then you would rename the copy of the project to *FrediGen_TradingPartnerB*. Also, in the App.config file, you have to change the value in the "tradingPartner" key field to "TradingPartnerB". This name must be consistent with the value in the *Company Name* field of the *TradingPartner* table in the *EdiDb.mdf* database.

The *Outbound Process* has two steps:

The first step is to pull data from your production database and store them into the intermediate database. This is the function of the *Add Records* button; however, in our example, we are populating the intermediate tables with fixed static data. But in practicality, you would have to change this code to pull data from your own production database and populate the intermediate tables with them. (The *View Tables* and *Delete Tables* buttons will view and delete only the records that were added into the tables.)

Once the data are in the intermediate database, can you then do the next step, which is to generate an EDI file by clicking on the *Generate* button. When an EDI file is generated, the *StatusCode* field in the *InterchangeOutbound* table is updated to "1". When a *CONTRL* acknowledgment for this EDI file is received and processed, the *StatusCode* field is updated to "2".

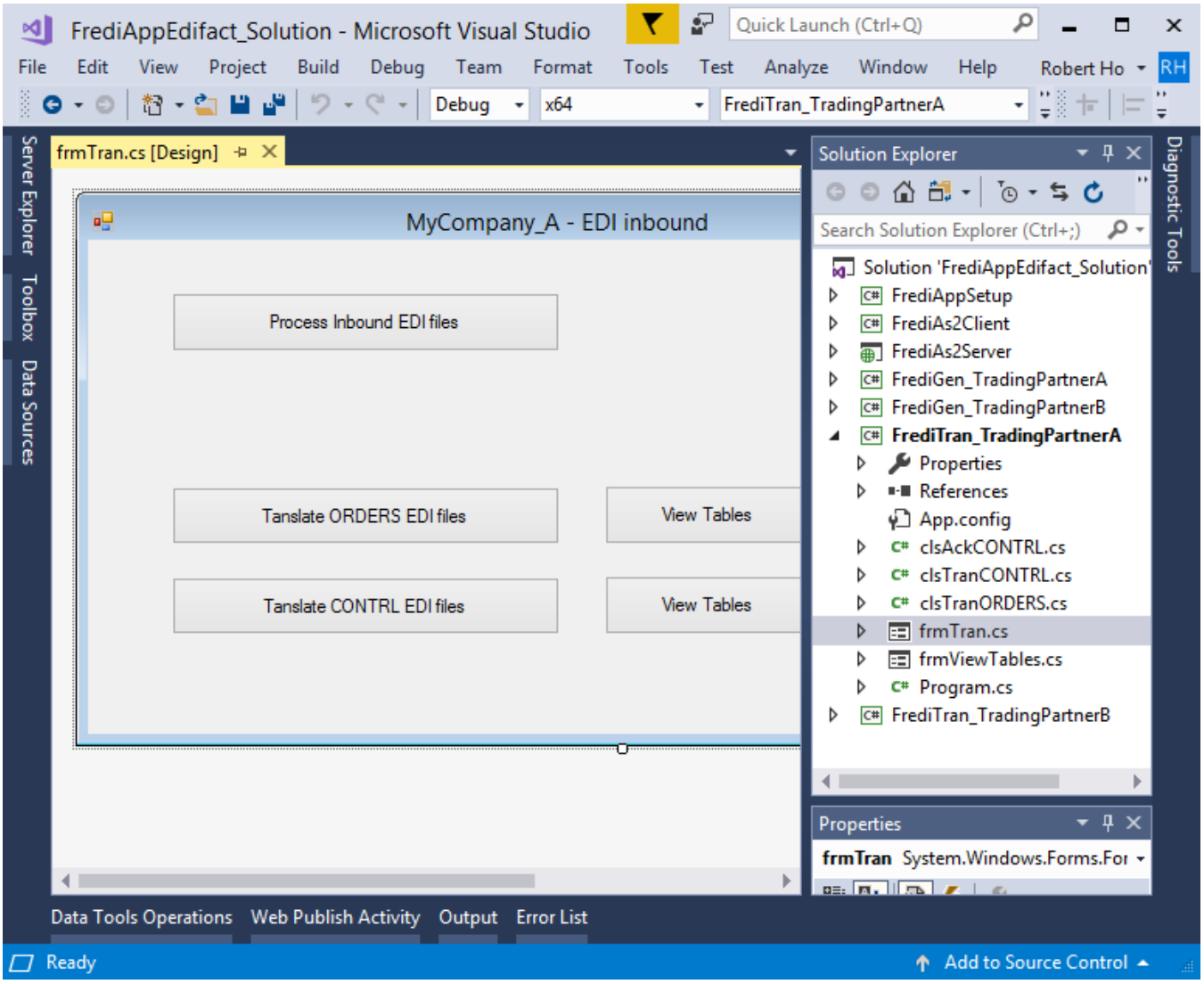


Note that for every Message document, you would need a set of these buttons. For example, you would need an *Add Records* and *Generate* button for *Message INVOIC*; and another set of *Add Records* and *Generate* button for *Message DESADV*. The methods for adding records into the database and generating the EDI files are kept in their respective classes named with the Message document name. For example, the class name that has the methods for generating Message *INVOIC* is *clsGenINVOIC*. The methods in these classes have the same name, but their functionality is specific to their Message document. If you wanted to create a DESADV outbound process for this trading partner, you could simply copy the INVOIC class, then modify the methods to the specifications of the DESADV and trading partner's requirements.

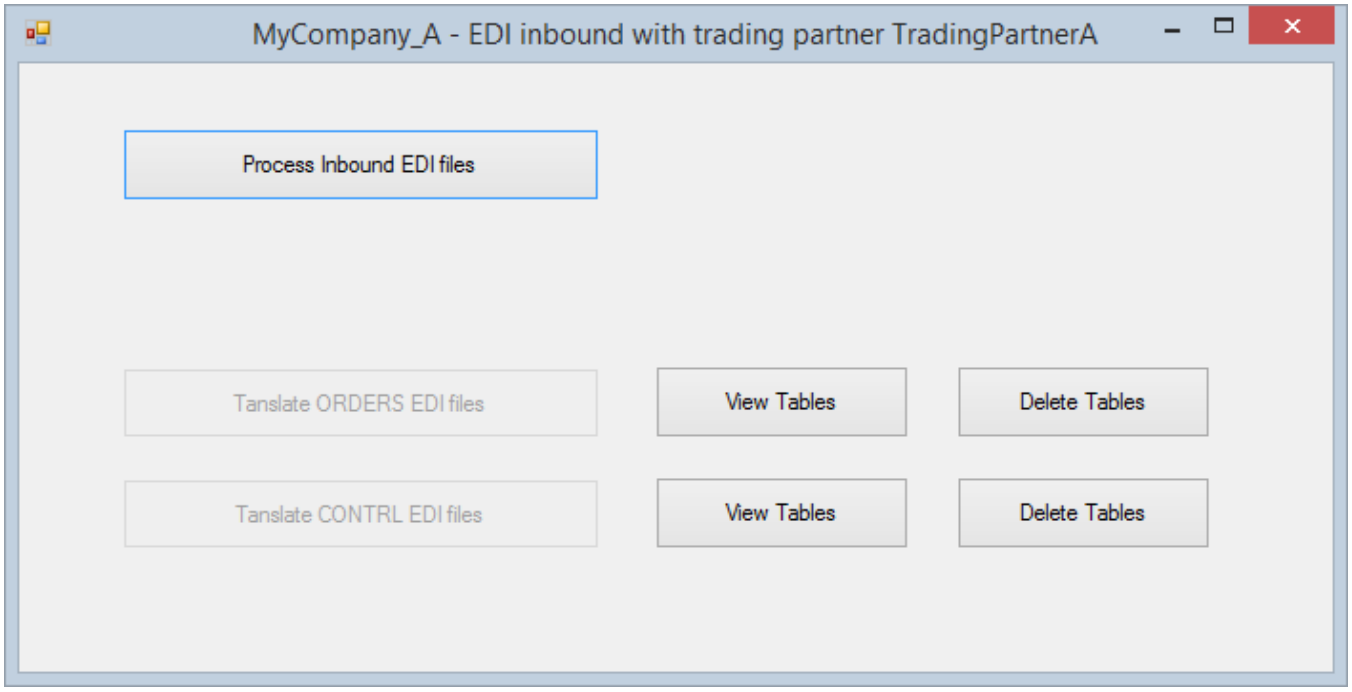
EDI files generated in this project will be put in the EDI_OUTBOUND folder.

The FrediTran_TradingPartnerA Project

This program processes the inbound EDI files for one trading partner. For this example, the trading partner is named “TradingPartnerA”.



Like the outbound program, you would have to create another program for other trading partner(s) should you have several of them. You can do this by using the existing one as your base and starting point. Just copy this project, rename it appropriately, change the trading partner's name in the app.config file, and then make the necessary modifications to the program according to your trading partner's specifications.



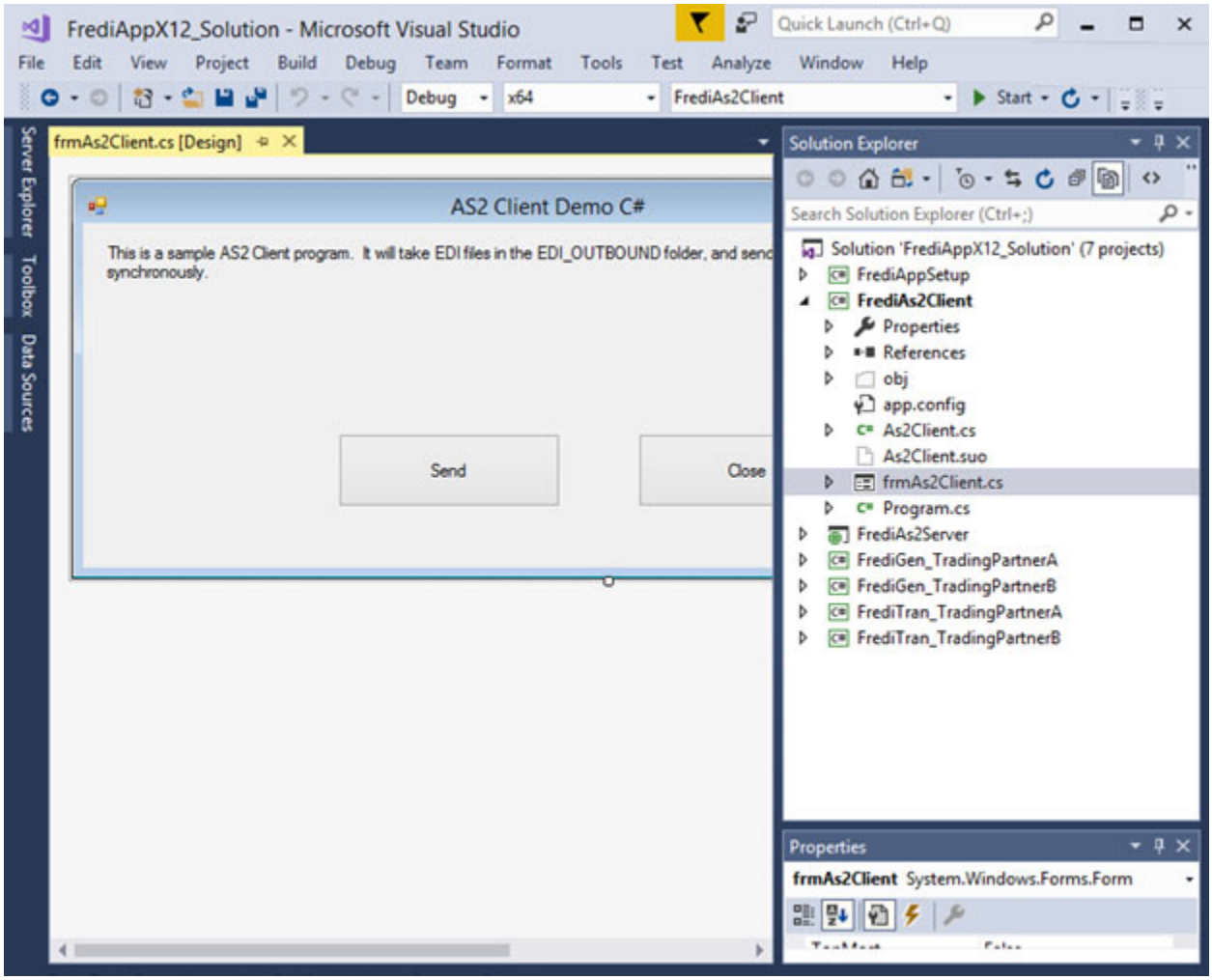
The *Inbound Process* starts by validating the EDI files (located in the trading partner's subfolder in the EDI_INBOUND folder) and at the same time creates the CONTRL acknowledgment (if requested). The acknowledgment files are created in the EDI_OUTBOUND folder ready to get sent back to the trading partner to acknowledge the EDI files received from them. If the inbound EDI file passes validation, it is immediately translated, and the data is saved in the intermediate database. The EDI file is then moved to the EDI_DONE folder once translation is completed. If the EDI file fails validation, it is moved to the EDI_REJECTED folder. A rejected EDI file does not get processed further.

If an inbound EDI file is a CONTRL, it will be translated immediately without going through the validation process. When completed, it will be moved to the EDI_CONTRL folder.

Like the design of the FrediGen project, the FrediTran project has a class for each *Transaction Set* it will be translating. So, this example has a class *clsTranORDERS* for translating inbound ORDERS EDI files, and a class *clsTranCONTRL* for translating CONTRL EDI files. And similarly, to add more Message documents and trading partners you can just copy classes and projects respectively and then make the necessary modifications that are specific to the Message documents and trading partner.

(The *clsAckCONTRL* class is for validating and creating CONTRL acknowledgment files and would be required in all copies of the FrediTran projects.)

The FrediAs2Client Project

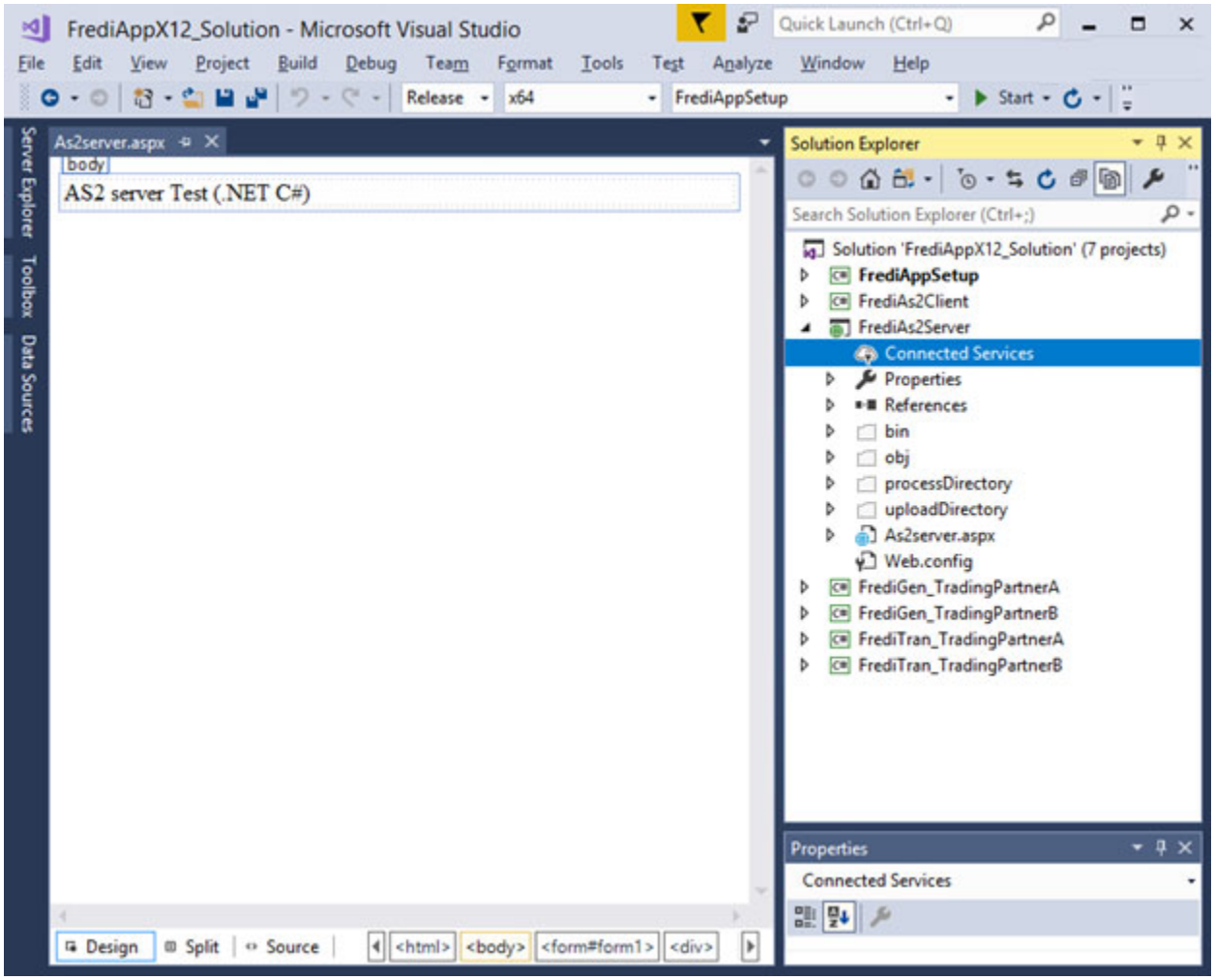


The As2Client program sends EDI files by AS2. It takes the EDI files in the EDI_OUTBOUND folder, reads their *Interchange Recipient ID* (UNB03-01), and looks it up against the EDI_CompanyID field of the *TradingPartner* table to determine where and how the EDI files are to be sent.

EDI files that were successfully sent are moved to the EDI_SENT folder, while those that failed are moved to the EDI_SENT_FAILED folder.

A copy of the sent AS2 file and the received MDN (acknowledging the receipt of the AS2 file) are put in the AS2_OUTPUT folder.

The FrediAs2Server Project



The AS2Server program receives AS2 messages from your trading partners. When an AS2 message is received, the program first reads its *AS2-From* header to obtain the sender ID. It then looks for this ID in the EDI_CompanyID field of the *TradingPartner* table to obtain the trading partner's setup and determine how the message should be decrypted, and where the extracted EDI file should go in the EDI_INBOUND folder.

Unlike the other projects (which are Desktop Applications), the FrediAs2Server project is a Web Application that requires IIS.