Neuron ESB 3.0.2 Release! More great things from the Neuron ESB Product Team

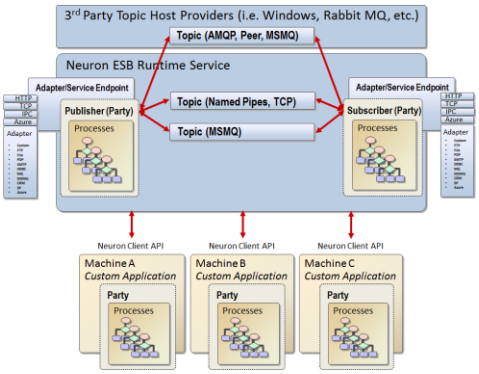
Just 2 months after we released our groundbreaking 3.0 version of Neuron ESB (read about the features); we're following it up with a 3.0.2 update. This update not only fixes issues we've found or were reported, but we also took the opportunity to include a number of new enhancements and features that customers asked for, and quite frankly, made a lot of great sense.

One of the advantages that we have within the Neuron ESB Product Team is our agility and ability to respond and partner with our customers to continually extend and make our product better with every release/patch/update. Our goal is to always deliver a great experience to our customers when they develop solutions on the Neuron ESB platform.

All the changes included in the 3.0.2 update can be found in the Neuron ESB Change Log which gets installed with Neuron ESB. Users can download the latest Neuron ESB Build, which includes the 3.0.2, from the Neuron ESB web site download page. In this blog post, I thought I would elaborate on some of the new enhancements and features we've added to the update.

Adapters

Adapters are key piece of capability in the world of an integration broker. They serve as the bridge to and from the bus between applications, databases and protocols. The completeness of "what" you ship as well as how easy it is for others to build their own adapters is critical in accelerating the development of any solution. Neuron ESB hands down has one of the easiest adapter frameworks to learn in the industry. However, we continue to deliver a more complete and richer set of adapters with every release. This update is no exception.



SFTP Adapter

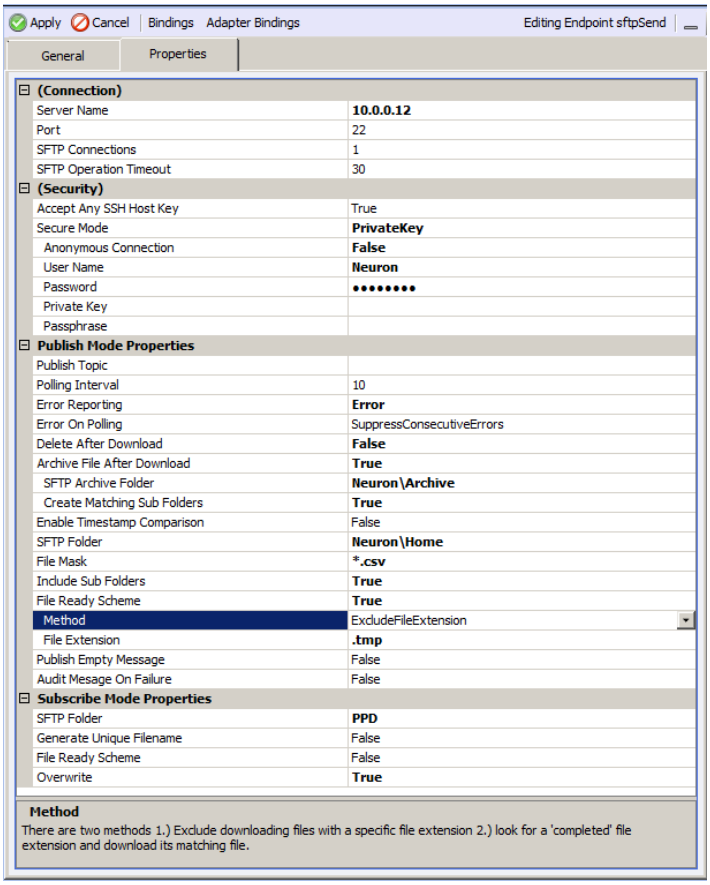
I've had several customers ask for an SFTP, so it was time that we built one and added it the library of adapters that we ship. We already shipped an FTP/FTPS adapter, so this rounds out our "FTP" family rather nicely. This has got to be one of the most full featured SFTP adapters on the market. A good comparison would be against the new BizTalk SFTP adapter…which supports only the barest connectivity features, making it NOT very useful for most customers. One of the benefits we have when we work directly with customers is that we get an opportunity to really understand their needs and use cases. It seems like most of them faced many of the same issues in the FTP world. Issues like:

- Not being able to delete the file being downloaded, yet need to ensure it's not downloaded twice. This could be due to permissions, but also due to the fact that other entities also need to

download the same file. Oh. And this kind of capability should survive shut downs or server failures.

- Being able to recursively search through a folder hierarchy for files to download. For instance, why duplicate the endpoint 50 times for every sub folder permutation or worse…duplicate the management and monitoring 50 times for every sub folder permutation when you could do it once.
- Being able to archive the file downloaded to another location on the SFTP server….effectively without saturating the bus.
- Ensuring that the file to download is actually completely written to the SFTP server before the download begins (i.e. File Ready strategies)
- Ensuring that the file being sent to an SFTP server isn't picked up by someone else on the SFTP server before its completely uploaded to the SFTP Server

In all of these cases, it only makes sense to build these capabilities directly into the adapter. For instance, you could build your own duplicate detection process that gets fired after the file is downloaded…and discard the file if it checks positive. But that would result in saturating a network with unnecessary downloads. Bandwidth costs both time and money for an organization.



The picture above illustrates all the properties that can be configured for a Neuron ESB SFTP adapter endpoint, depending on if the adapter is monitoring an SFTP site and publishing files to the bus, or is receiving messages from the bus and uploading them to an SFTP server.

A Neudesic consultant spoke with me the other day and described to me a challenge that a customer had using someone else's adapter. They had to validate the message received with the actual size reported by the SFTP server. Because "their" adapter didn't support this (didn't actually support much), they had to write a lot of SFTP specific and elaborate code in their own process which they'll have to maintain moving forward. Means they also had to learn all about SFTP and how to talk to it…which is what the adapter is supposed to abstract away. I think it's a feature I'll add to our adapter next week ☺

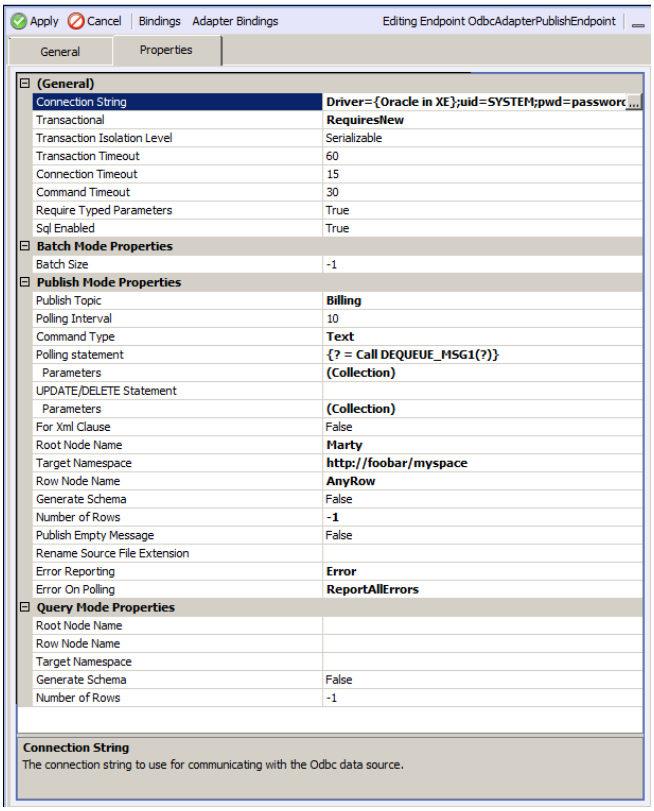
In any case, the Neuron ESB SFTP (and now the FTP/FTPS) adapter has so many features; I'm going to dedicate an entire Blog article on it next week. Stay tuned.

FTP Adapter

After all the work we did on the SFTP Adapter feature set, it only made sense that we ported those same features to our existing FTP/FTPS adapter. There's one minor exclusion though, recursive searches through folder hierarchies. However, expect that feature soon.

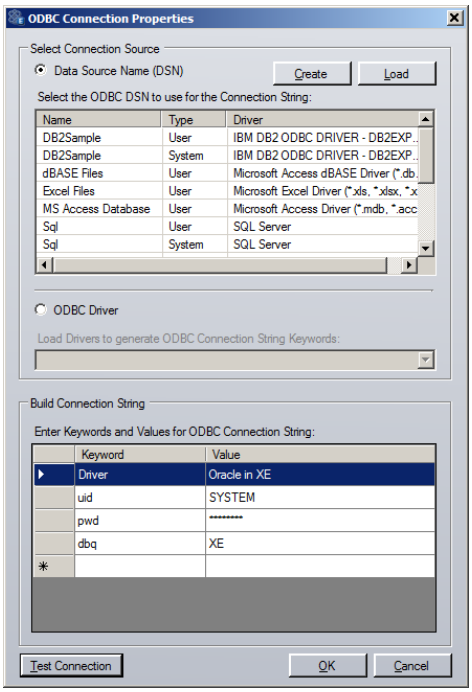
ODBC Adapter

After building out the Neuron ESB ODBC adapter, I know why companies like Microsoft won't produce their own. It's hard! We have customers today using our ODBC Adapter against data sources like Oracle, DB2, Lotus Notes, Excel, Flat files and Microsoft Sql Server. We've been able to garner a lot of great feedback from our customers regarding its ease of use and capabilities. Our customer interaction has allowed us to augment our adapter, even beyond "vanilla" ODBC. For example, when using the Microsoft Sql Client ODBC driver, our ODBC adapter will natively process FOR XML clauses. It even handles RETURNVALUE or OUTPUT type of parameters for stored procedures and can optionally generate XSD Schemas.



In this release we've added some minor things like configurable Command and Connection timeouts. However, we took the opportunity to completely rewrite our transaction support. Now rather than just choosing to have the adapter participate in the ambient transaction or not, users can choose either to enlist in the underlying transaction (if using MSMQ based topics…there will be one), not have it

within a transaction at all (i.e. the stored procedure being called may have its own transaction semantics written into it), or start up an entirely new transaction, separate from the ambient one. The latter allows the use of the Light Weight Transaction Manager rather than enlisting something heavier like the Microsoft Distributed Transaction Coordinator against target data sources that support it. And even if they don't, the locking of resources is minimized because the transaction won't be enlisted in the ambient transaction if using MSMQ Transactional Topics. Hence some performance and concurrency gains to boot!



We shipped an ODBC Connection String Builder User Interface with Neuron ESB 3.0. With that release we reworked the UI and made a lot of operations asynchronous. The ODBC Connection String Builder can be accessed either through the Connection String property of the ODBC Adapter or ODBC Process Step. However, many of the features only worked with the 32 bit (x86) install of Neuron. Now all these features work with the 64 bit install of Neuron. The UI allows for users to select existing ODBC data sources, create new ones, enter connection string key words and values, interrogate ODBC drivers to retrieve their supported connection string key words and descriptions and test the resulting connection string to ensure a connection can be made to the data source.

In addition, we reworked the existing Batch mode capability. Batch mode essentially allows users to perform "batch" inserts into a table efficiently and fast (similar to using BCP in Microsoft Sql Server). Under the hood, the ODBC Adapter groups all the inserts into one or more batched statements (the number of statements within a batch is configurable), sending the entire batch directly to the database for the server to process, rather than sending one insert at a time over the wire.

In previous releases, we only supported inserting directly into a Table and by default, all column data types were sent over as varchar. This had some limitations. First, many organizations are loathe to provide permissions directly to tables. Most prefer to manage all data access through stored procedures (a good thing really☺). Also, many scenarios require more of an insert/update type strategy, which can only be done with stored procedures. Lastly, some database servers could do automatic casting of "some" data types, but not all. And some just don't do automatic casting well at all. Hence, having every column being resolved as a varchar quickly became problematic.

In this release we added support for calling stored procedures as batched statements, as well as strongly typing the data types of columns and parameters. For instance, the following XML formatted message, if sent to the ODBC adapter would result in the batched execution of the "uspUpdatePhoneNumber" stored procedure three times, with each passing strongly typed parameters.

```
<Batch>
   <Statement type="StoredProcedure" name="uspUpdatePhoneNumber">
```

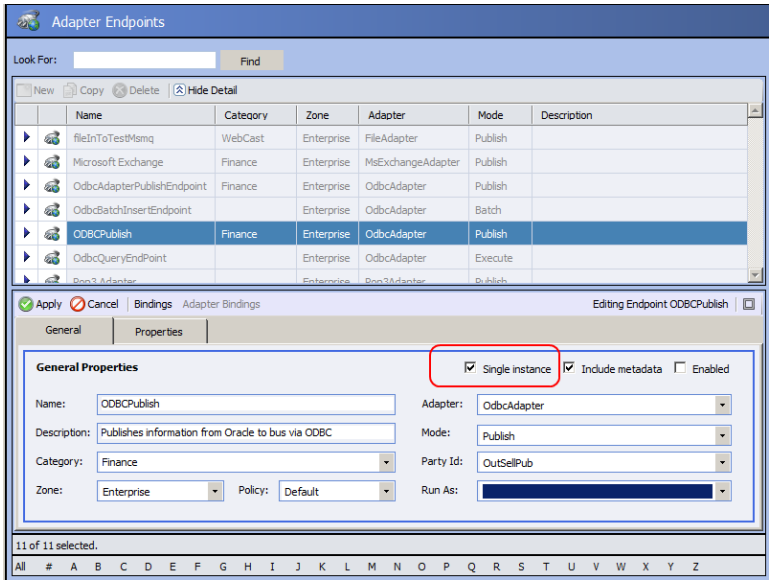
```
        <Fields>
            <Field type="int" name="Id">2</Field >
            <Field type="varchar" name="PhoneNumber">22222222</Field >
        </Fields>
        <Fields>
            <Field type="int" name="Id">21</Field >
            <Field type="varchar" name="PhoneNumber">21212121</Field >
        </Fields>
        <Fields>
            <Field type="int" name="Id">24</Field >
            <Field type="varchar" name="PhoneNumber">24242424</Field >
        </Fields>
    </Statement>
</Batch>
```

The ODBC adapter does a nice job in allowing Neuron ESB to participate in more complex Master Data Management solutions.

Single Instance Mode

Deploying Neuron to highly available environments usually means running at least 2 or more Neuron servers in a load balanced configuration. However, depending on how messages are published to the bus, this configuration could be problematic for most other vendors. Take for example the scenario where 2 identical Neuron servers are up and running based on the same ESB Solution Configuration. What "if" the solution contained an FTP Adapter endpoint, or some other adapter endpoint that doesn't support locking of resources or transactions, that needs to publish messages to the bus. Such a configuration would normally result in duplicate messages being published, usually not the desired result. Most vendors don't handle this scenario at all. When I was at Microsoft, our answer was to tell customers to write a lot of their own code to try to prevent this when using BizTalk Server.

In Neuron 3.0.2, we augmented our "Single Instance" setting for adapter endpoints. In previous releases, this just worked across servers and specifically was designed to handle the scenario described above. In other words, if you had an endpoint marked as "Single Instance", the servers would monitor each other and ensure only one instance of the adapter endpoint would be running at any one time. If an endpoint went down, was disabled or was stopped, the other endpoints on the other servers would be notified and one would launch an instance of the same endpoint to take over for the failed one. This provided high availability of adapter endpoints in load balanced scenarios. In this release, we extended this capability to the runtime instance level, rather than just the server level. So now you can have the same solution or endpoint deployed more than 1 dedicated Neuron runtime instance on the same server. If an endpoint goes down in one runtime instance, it will start up in the other…all on the same box.

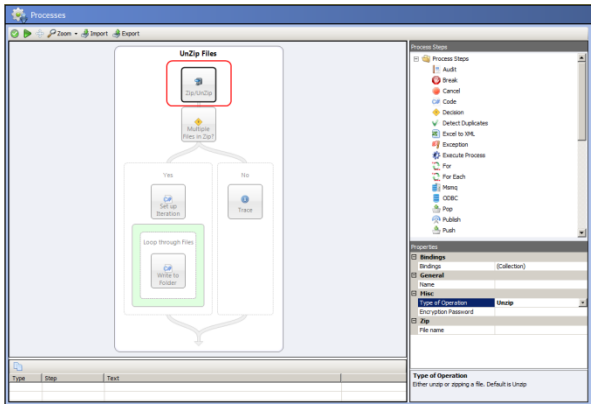


Business Processes

I think we delivered some pretty good things to make building and testing business processes easier within the Neuron ESB Explorer. We always try to deliver "productivity" out of the box. If we can find an opportunity to reduce the amount of code someone has to write in our designer, we'll try to take the opportunity to do so.

Zip/Unzip Process Step

I decided to move forward and include this functionality. Turns out it was easy for me to build. It seems we have a lot of customers that need to retrieve and process zip files from FTP sites….so this little Process Step makes life very easy for them. It supports password encryption as well as multiple files within a zip. A process can even be put together to recursively iterate through a zip in case it contained other zip files. This can be tested directly within the business process designer of the Neuron ESB Explorer.

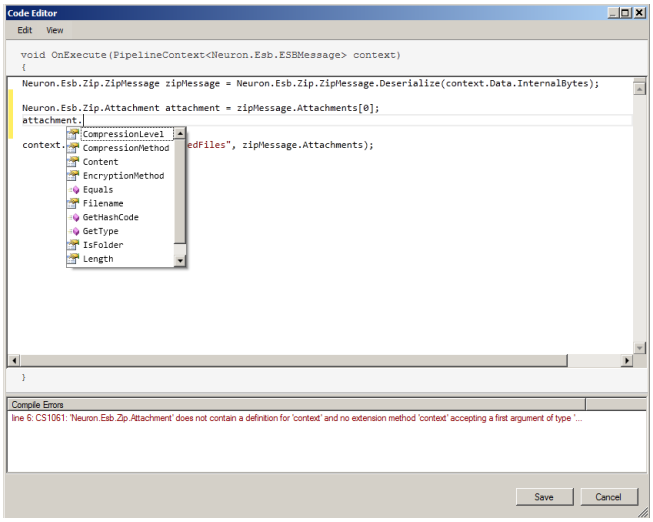


The diagram above depicts a normal use case using the Zip/Unzip Process Step.

- Message is received
- Zip/Unzip will process it (will automatically detect if it's a zip file archive)

- There a check to see if there are multiple files in the zip.
- If there are, a new Zip API is used in conjunction with the ForEach Process Step to iterate though them all and republish each one to the bus.

We included an API so that users could work directly with the unzipped messages, whether there was only one…or many. This way they could easily loop through and republish or process them. Here's a screen shot of using the API with a Code Process Step:



ODBC Process Step

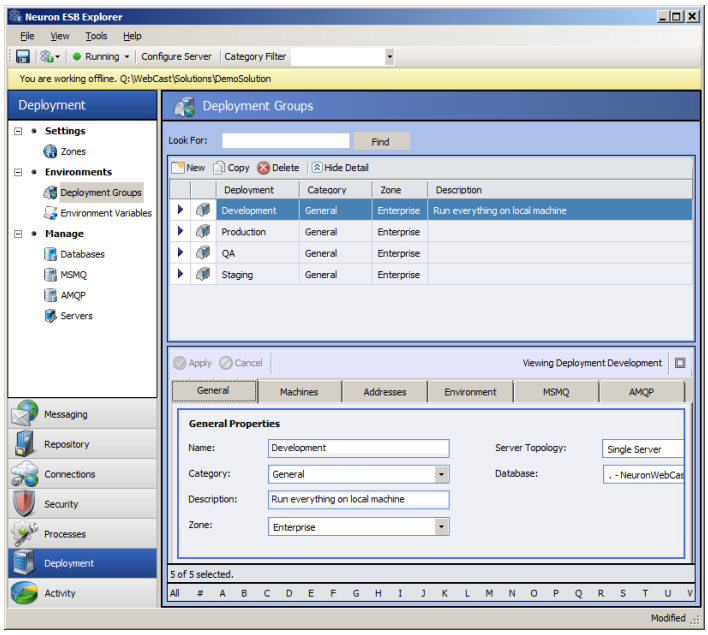
The nice thing about the ODBC Process Step, it automatically inherits all the great enhancements that we made to the ODBC Adapter. The ODBC Process Step allows organization to query or execute statements directly against any ODBC data source within a Business Process.

Environmental Variable support in Process Designer

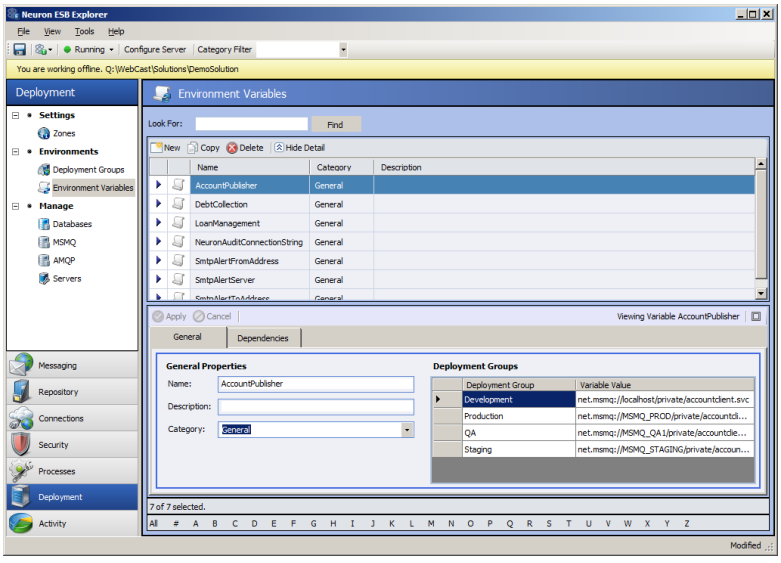
Neuron ESB ships with a great deployment story in the 3.0 release. We really wanted to make configuration and deployment something easy to do…and easy to manage between all the various environments an organization may. For instance, many organizations will have dedicated environments for development, QA, Test, Staging and Production. In each of these environments, the resources that an ESB Solution calls out to, accesses or needs to expose could be slightly different. For example, the service endpoint that a message is routed to could be entirely different in the QA environment than in the Production environment. In fact, the security model could be very different as well. In all of these cases, we thought a mechanism should be provided that allows customers to seamlessly port an ESB solution from one environment to another without intervention or rewrites of the solution.

When I was working for Microsoft, BizTalk Server used the concept of generating MSI packages, exporting all configurations to XML as a way of handling deployment to different environments. Users would have to duplicate the XML for each environment and manually find and edit the correct values (where applicable if supported) for all environments and then re-package those up in an MSI. After 6 years of working with customers and the field on BizTalk deployments, well…it was hard, costly, took a lot of effort and was complicated and troublesome to maintain.

With Neuron ESB, we wanted something far more intuitive and easier to use. So long ago we introduced the concept of Environmental Variables and Deployment Groups. Within every Neuron ESB Solution, users can define any number of Deployment Groups (pictured below).



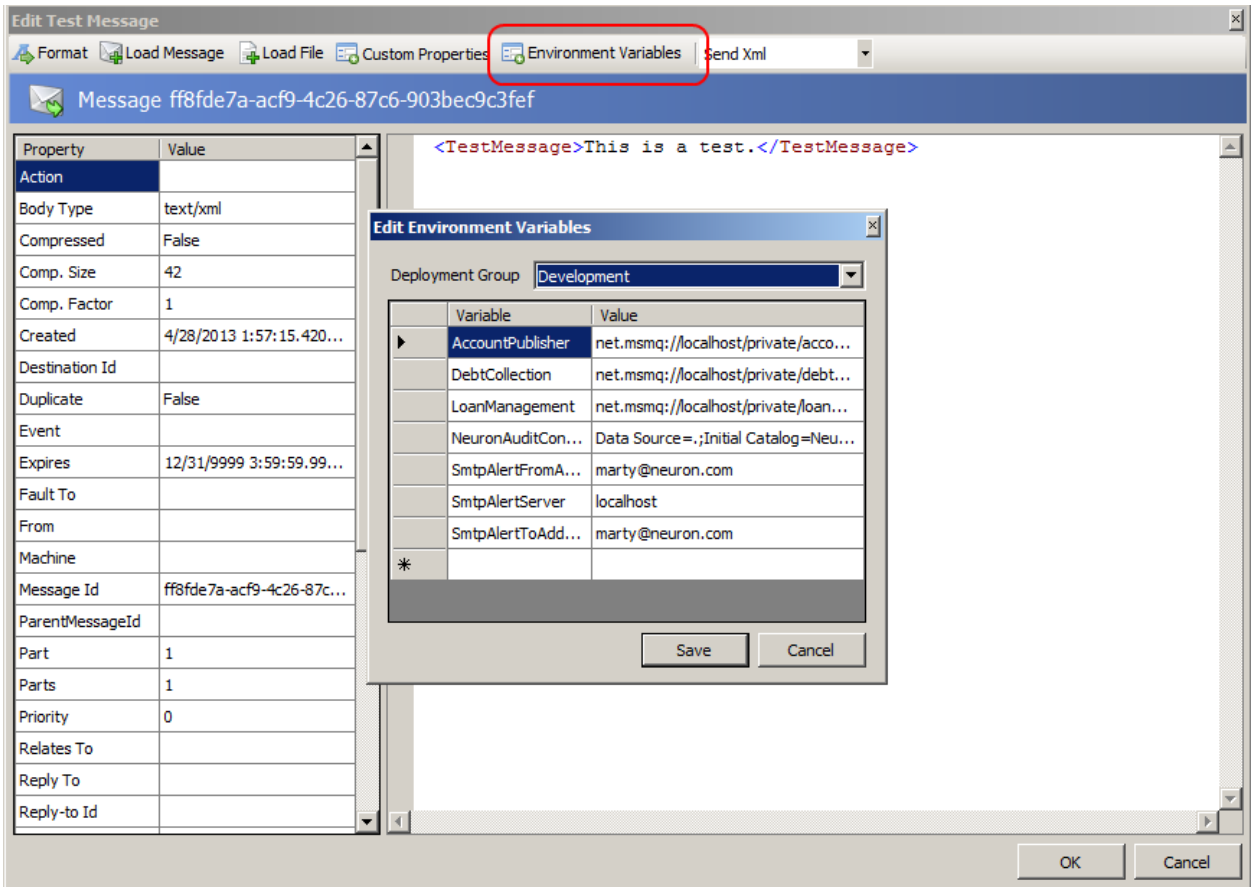
Following that, they can define what we call an “Environmental Variable”. This variable can be assigned a unique value per deployment group.



Then within the Neuron ESB Explorer User Interface, users can use Environmental Variables to configure any/most properties of Adapter Endpoints, Service Endpoints, Databases or even individual Process Steps within a Business Process. We found this approach infinitely simpler and easier to manage than what users are forced to work with in the BizTalk world.

However, there were things we could better. Users may (and really should) want to test their Business Processes or other logic against the Environmental Variable's values that they define for any deployment group. Previously, this was not available in our original release of Neuron ESB 3.0, but we added to the 3.0.2 release!

When testing a process within the Neuron ESB Explorer's Business Process Designer users can launch the "Edit Test Message" form and select the new "Environmental Variables" toolbar button. This will launch the "Edit Environmental Variable" dialog that allows users to select the deployment group's set of environmental variables that they require to be accessible at design time during their test as depicted below.



Additionally, users can access and inspect the values of their environmental variables directly within the Process Designer via a Code Process Step. In previously releases users could access and retrieve Environmental Variables, but ONLY at runtime using C# within a Code Process step similar to below:

```
System.Text.StringBuilder sb = new System.Text.StringBuilder();
foreach(string s in Neuron.Configuration.NeuronEnvironment.Current.VariableNames)
    sb.Append(s.ToString());

context.Data.FromString(sb.ToString());
```

In the 3.0.2 release, users can access Environmental Variables at either runtime, or within the Process Designer's design time environment during testing of any business process using C# within a Code Process step similar to below:

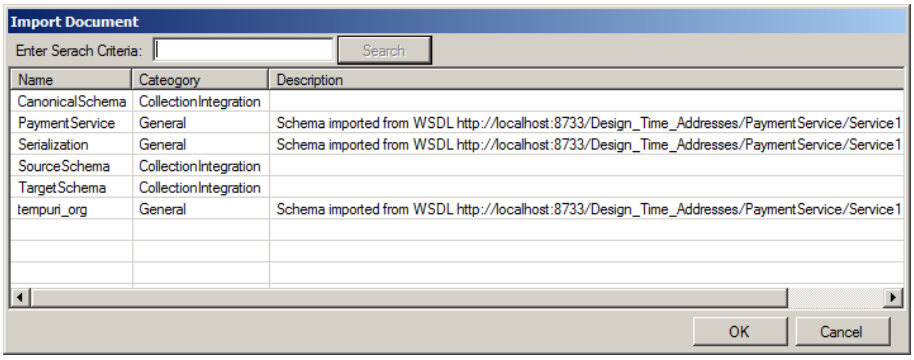
```
foreach(var s in context.EnvironmentVariables)
    context.Instance.TraceInformation(string.Format("key={0}, value={1}",s.Key,s.Value));
```

This means that any Process Step configured to use Environmental Variables, will work as expected against the selected Deployment Group's Environmental Variables within the Process Designer's design time environment while testing.

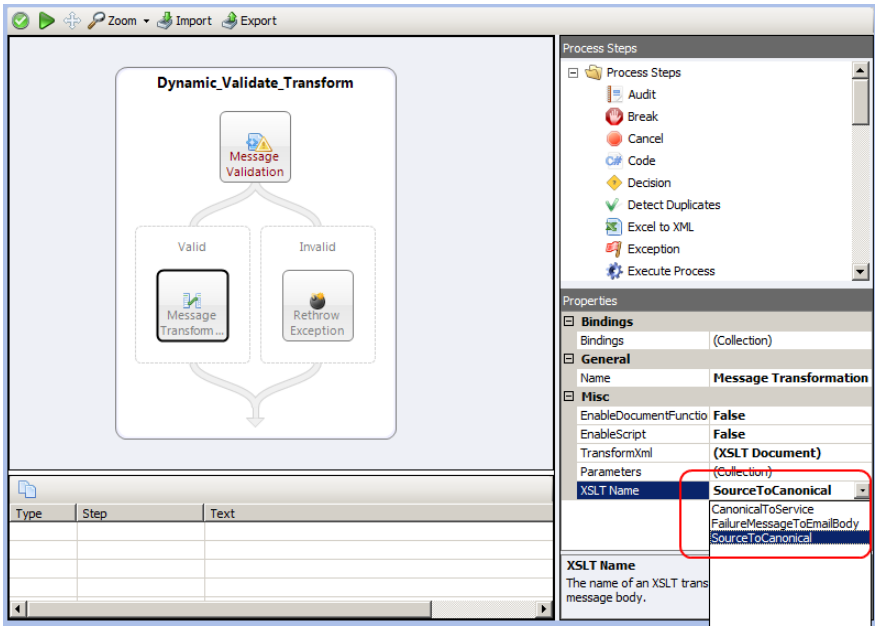
Environmental Variables have turned out to be very useful with our customer base, not just for configuring Neuron entities, but also to use for their own custom application solution needs.

Schema Validation and XSLT Transform Process Steps

In Neuron ESB configuring standard VETO (Validate, Extract, Transform, Operate) patterns has always been fairly easy to do. Users can create a Business Process, drag the Schema Validation and XSLT Transform process steps onto the surface and configure them using the property grid. We've always been pretty flexible. Users could either set the schema and transform properties at runtime using C# to accommodate dynamic resolution or, import schemas and XSLTs from disk or just select to import them in from the Neuron ESB Repository via the Import Document dialog below.

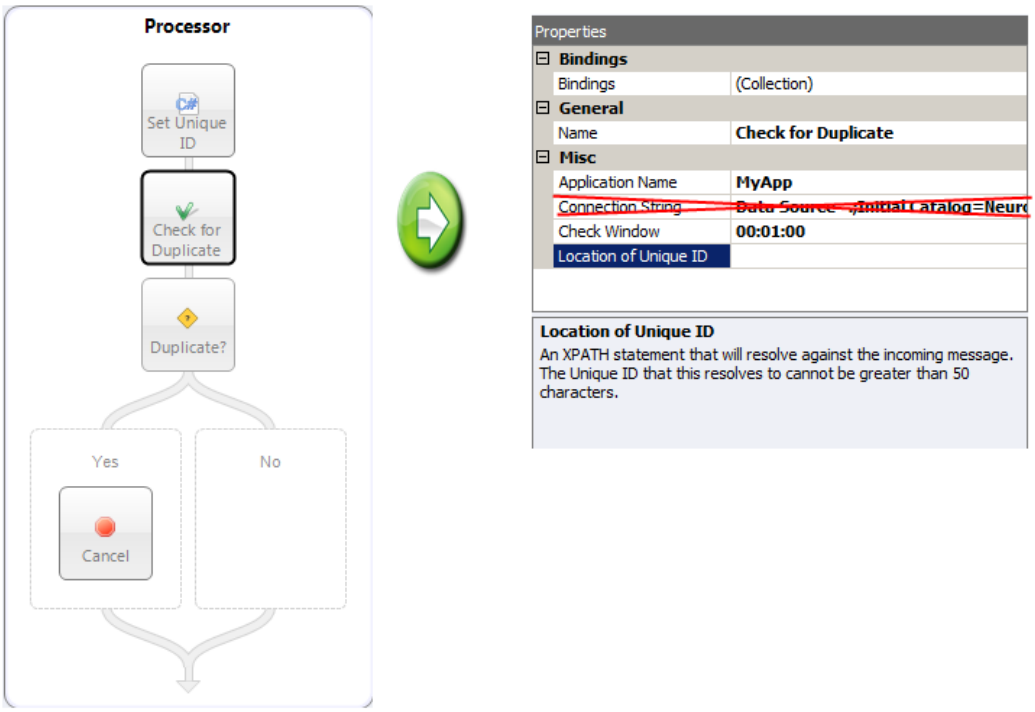


The only downside is that if users imported from the Repository, they were creating a copy of the documents to use within the process. This meant that if they later changed or updated the Schemas/XSLTs that existed in the Repository the process would be left unchanged at runtime and would continue to use the outdated Schemas/XSLTSs that were originally imported into the Process Steps. In Neuron ESB 3.0.2, we're providing a new option to configure these Process Steps by selecting what Schemas/XSLTs to use from the Repository…linking them at runtime/design time, rather than importing in a static copy. To be clear, users can now do both by accessing a list of the object names as depicted below!



Detect Duplicates Process Step

I added this Process Step in Neuron ESB 3.0, but it immediately drove me crazy that I had to configure the connection string to the Neuron Audit database as one of the properties. Well you guessed it. In Neuron ESB 3.0.2 we resolve this automatically at runtime.



For design time, if testing is required users can pass in a custom property when using the “Edit Test Message” dialog to submit a message to the Business Process Designer. The prefix for the custom

property is “dupcheck”; the property name to use is “ConnectionString”. The connection string to the Neuron Audit database can always be obtained by navigating to the Deployment->Manage-.>Databases section of the Neuron ESB Explorer. Select the database, click on the “Edit Connection String” command button which will launch the Connection Properties form. Clicking the “Copy” button which will copy the connection string to the clipboard.

Well that’s all for now. Looking forward to posting more information about our product in the future. Stay tuned!