



Epicor ERP
Epicor Service Connect 10 for
Epicor ERP Course
10.0.700

Disclaimer

This document is for informational purposes only and is subject to change without notice. This document and its contents, including the viewpoints, dates and functional content expressed herein are believed to be accurate as of its date of publication. However, Epicor Software Corporation makes no guarantee, representations or warranties with regard to the enclosed information and specifically disclaims any applicable implied warranties, such as fitness for a particular purpose, merchantability, satisfactory quality or reasonable skill and care. As each user of Epicor software is likely to be unique in their requirements in the use of such software and their business processes, users of this document are always advised to discuss the content of this document with their Epicor account manager. All information contained herein is subject to change without notice and changes to this document since printing and other important information about the software product are made or published in release notes, and you are urged to obtain the current release notes for the software product. We welcome user comments and reserve the right to revise this publication and/or make improvements or changes to the products or programs described in this publication at any time, without notice. The usage of any Epicor software shall be pursuant to an Epicor end user license agreement and the performance of any consulting services by Epicor personnel shall be pursuant to Epicor's standard services terms and conditions. Usage of the solution(s) described in this document with other Epicor software or third party products may require the purchase of licenses for such other products. Where any software is expressed to be compliant with local laws or requirements in this document, such compliance is not a warranty and is based solely on Epicor's current understanding of such laws and requirements. All laws and requirements are subject to varying interpretations as well as to change and accordingly Epicor cannot guarantee that the software will be compliant and up to date with such changes. All statements of platform and product compatibility in this document shall be considered individually in relation to the products referred to in the relevant statement, i.e., where any Epicor software is stated to be compatible with one product and also stated to be compatible with another product, it should not be interpreted that such Epicor software is compatible with both of the products running at the same time on the same platform or environment. Additionally platform or product compatibility may require the application of Epicor or third-party updates, patches and/or service packs and Epicor has no responsibility for compatibility issues which may be caused by updates, patches and/or service packs released by third parties after the date of publication of this document. Epicor® is a registered trademark and/or trademark of Epicor Software Corporation in the United States, certain other countries and/or the EU. All other trademarks mentioned are the property of their respective owners. Copyright © Epicor Software Corporation 2014. All rights reserved. No part of this publication may be reproduced in any form without the prior written consent of Epicor Software Corporation.

ED8932905

90521-10-9244-58310700

10.0.700

Revision: June 08, 2014 5:33 p.m.

Total pages: 87

course.ditaval

Contents

Epicor Service Connect for Epicor ERP Course.....	6
Before You Begin.....	7
Audience.....	7
Prerequisites.....	7
Environment Setup.....	7
Task Monitor.....	8
.....	9
Workshop Constraints.....	10
Overview.....	13
Implementation.....	17
Epicor Service Connect Environment.....	18
Epicor Service Connect Administration Console.....	18
Activities.....	19
Companies.....	19
Document Tracking.....	19
Languages.....	20
Tasks.....	20
Connectivity.....	20
Backup and Restore.....	22
Security.....	23
Events.....	23
Locks.....	23
Licensing.....	23
Epicor .NET Business Objects.....	24
References.....	24
Workflow Designer.....	25
Generate Schema from Sample Data.....	30
Message Extensions and Process Variables.....	30
Internal Envelope.....	31
SharePoint Integration.....	32
Workflow Foundation Integration.....	33
Task Monitor.....	33
Service Connect Services.....	34
Daily Processing.....	36
.NET References Import.....	36
Workshop - Import .NET Reference.....	37
Log in to the ESC Administration Console.....	37
Add a .NET Reference.....	37
Enter .NET Reference Information.....	38
Enter Epicor Logon Settings.....	38

Workflow Basics.....	39
Workshop - Create a Basic Workflow.....	39
Create Parts.....	39
Create Folders and Sample Data.....	40
Add a Message Type.....	40
Add a Sender.....	41
Add an Input Channel.....	41
Create the Workflow.....	42
Generate a Schema.....	43
View the Generated Schema.....	44
Add a Conversion to the Workflow.....	44
Define the Conversion.....	45
Add a .NET Call.....	46
Connect the Workflow Elements.....	46
Add a Message Map.....	47
Validate the Workflow.....	47
Run the Workflow.....	48
Verify the Results.....	48
Verify Part Maintenance.....	48
Common Workflow Techniques.....	49
Workshop - Use Common Workflow Techniques.....	49
Create Sample Data.....	49
Update the Schema.....	50
Add a Message Extension.....	51
Add a Process Variable.....	51
Revise the Update Type Conversion.....	52
Add a Choice to Test for Errors.....	53
Define the Choice Properties.....	54
Configure the Restore Original Data Conversion.....	55
Define the Conversion.....	55
Configure the Resolve Errors Task.....	56
Run the Workflow.....	57
Check the Workflow Progress.....	57
Use the Task Monitor.....	58
Verify the Results.....	58
Document Publication from a Workflow.....	59
Workshop - Send an E-Mail from a Workflow.....	59
Create an Output Channel.....	59
Add a Poster to the Workflow.....	60
Set the Poster Properties.....	60
Create an E-Mail Template.....	61
Run the Workflow and Verify the E-Mail.....	62
Functoids.....	63
Workshop - Use a Value Conversion Functoid.....	63
Create Sample Data.....	63

Add the Functoid to the Conversion.....	63
Run the Workflow and Verify the Data.....	65
Sub-Workflow Calls.....	65
Workshop - Use a Sub-Workflow Call.....	66
Create Sample Data.....	66
Make a Copy of the Original Workflow.....	66
Update the Namespaces.....	66
Update the default namespace in the input channel configuration.....	67
Update the namespace in the incoming document schema.....	68
Create the Master Workflow.....	68
Add the Sub-Workflow to the Master Workflow.....	69
Create a Sub-Workflow Schema.....	70
Convert the Message Extension to Use the Sub-Workflow Schema.....	70
Revise the Update Type Conversion.....	71
Rebuild the Update Type Conversion XSLT.....	71
Revise the Restore Original Data Conversion.....	72
Update the Message Map.....	73
Run the Workflow and Verify the Results.....	73
Database Operations.....	74
Workshop - Use DBOperation Workflow Element.....	74
Create Folders and Sample Data.....	74
Add a Message Type.....	75
Add a Sender.....	75
Add an Input Channel.....	76
Add an Output Channel.....	77
Import .Net Reference.....	77
Create the Workflow.....	78
Add Elements to the Workflow.....	79
Generate a Schema.....	79
Define the Conversion.....	80
Edit the Conversion.....	81
Define the .NET Call.....	82
Define the DBOperation Workflow Element.....	82
Build the Statement.....	83
Define the Poster.....	84
Add a Message Map.....	84
Run the Workflow.....	85
Verify the New Customer Record.....	85
View the Database Output.....	85
Conclusion.....	86

Epicor Service Connect for Epicor ERP Course

This course focuses on the main features of Epicor Service Connect and how to integrate it with the Epicor application. Service Connect is a powerful development tool that allows you to build workflows which can automate processes within an application or connect different business entities, applications, or users. It harnesses the power of XML and other open standards.

Service Connect workflows use XML documents as the primary interface. The workflows are designed to send documents to, and use documents from exposed services of other applications, such as .NET Business Objects Epicor 10. Service Connect workflows primarily map structures to convert this data so information is ready to use by other processes.

By using the document as the interface, you can quickly integrate applications and processes into a **loosely coupled** environment. To insert a new routine or new piece of information into the process, it is quicker and easier to incorporate into a loosely coupled environment than into an environment built on traditional programming practices.

Service Connect is designed to fully leverage the Service Oriented Architecture (SOA) of other applications. You can set up Service Connect workflows themselves as services designed to consume and return information on demand. In that sense, you can use Service Connect to create its own SOA environment.

Upon successful completion of this course, you will be able to:

- Understand how Service Connect is built to support processes that connect different business entities, applications, and users.
- Understand data conversions and the purpose of an internal envelope.
- Become familiar with Service Connect services.
- Work with the Epicor Service Connect Administration Console to define connectivity processing options for Service Connect.
- Use the Workflow Designer to define workflows.
- Create conversions and use .NET methods to transform a set of data from a specific source format to a specific target format.
- Use different workflow elements, such as choice, poster, or task.
- Work with the Task Monitor.
- Become comfortable with the mapper tool to generate XSLT code the conversions use.
- Process an external file, such as a comma separated (.csv) text file or Microsoft® Office® Excel® (.xls) file, as input to Service Connect.
- Import .NET assembly references.
- Use the DBOperation element to run statements against the application database.

Before You Begin

Read this topic for information you should know in order to successfully complete this course.

Audience

Specific audiences will benefit from this course.

- System Administrator
- IT/Technical Staff

Prerequisites

To complete the workshops in this course, the necessary modules must be licensed and operating in your training environment. For more information on the modules available, contact your Epicor Customer Account Manager at EpicorCAM@epicor.com. It is also important you understand the prerequisite knowledge contained in other valuable courses.

- **Navigation Course** - This course introduces navigational aspects of the Epicor application's user interface. Designed for a hands-on environment, general navigation principles and techniques available in two user interface modes - **Classic Menu** and **Modern Shell Menu**. Workshops focus on each of these modes and guide you through each navigational principle introduced.
- **Knowledge Camp Course** - This course provides a high level overview of the quote to cash flow through the Epicor application. You begin with how to create a quote, process it as an order, and fill the order across production planning and purchasing. The course also covers the manufacturing plan and shipment of parts to a customer, as well as how to process invoices, enter cash receipts, and generate supplier payments.
- **Database Concepts Course** - This course reviews the table and field name identification process using Field Help, Customization Tools, and the Data Dictionary Viewer functionality. It also describes table linking procedures and requirements as well as join type definitions and specifications.

The following industry knowledge is recommended:


- Basic understanding of the application file structure and how to manage data in your Epicor application.
- Experience with .NET assemblies, XML, XSLT, and SQL Server.
- Fundamental knowledge of relational database concepts such as table relationships, records, and field types.

Environment Setup

The environment setup steps and potential workshop constraints must be reviewed in order to successfully complete the workshops in this course.

Your Epicor training environment, in which the Epicor demonstration database is found, enables you to experience Epicor functionality in action but does not affect data in your live, production environment.

The following steps must be taken to successfully complete the workshops in this course.

1. Verify the following or ask your system administrator to verify for you:
 - **Your Epicor training icon (or web address if you are using Epicor Web Access) points to your Epicor training environment with the Epicor demonstration database installed.** Do not complete the course workshops in your live, production environment.
-  **Note** It is recommended that multiple Epicor demonstration databases are installed. Contact Support or Systems Consulting for billable assistance.
- **The Epicor demonstration database is at the same service pack and patch as the Epicor application.** Epicor's education team updates the Epicor demonstration database for each service pack and patch. If your system administrator upgrades your Epicor application to a new service pack or patch, he or she must also download the corresponding Epicor demonstration database from EPICweb > Support > Epicor > Downloads and install it. If this is not performed, unexpected results can occur when completing the course workshops.
 - **Your system administrator restored (refreshed) the Epicor demonstration database prior to starting this course.** The Epicor demonstration database comes standard with parts, customers, sales orders, and so on, already defined. If the Epicor demonstration database is shared with multiple users (that is, the database is located on a server and users access the same data, much like your live, production environment) and is not periodically refreshed, unexpected results can occur. For example, if a course workshop requires you to ship a sales order that came standard in the Epicor demonstration database, but a different user already completed this workshop and the Epicor demonstration database was not restored (refreshed), then you will not be able to ship the sales order. Epicor's education team has written the course workshops to minimize situations like this from occurring, but Epicor cannot prevent users from manipulating the data in your installation of the Epicor demonstration database.
2. Log in to the training environment using the credentials **manager/manager**. If you are logged in to your training environment as a different user, from the Options menu, select Change User.
 3. From the Main menu, select the company **Epicor Education (EPIC06)**.
 4. From the Main menu, select the **Main site**.
 5. **Epicor Service Connect 10** must be installed in your environment.

Task Monitor

To be able to work with the Task Monitor, perform the following additional setup:

- Add scsghost to the trusted sites.
 - Setup automatic logon with current user name and password
 - Update the registry
1. Open the Internet Explorer.
 2. From the **Tools** menu, select **Internet options**.
 3. On the **Security** tab, select **Trusted sites**.
 4. Click the **Sites** button.
 5. In the **Add this website to the zone** field, type **http://scsghost**.

6. Click **Add**.
7. Click **Close**.
8. On the **Security** tab, click the **Custom level** button.
9. In the **Security Settings - Trusted Sites Zone** window, scroll down to the **User Authentication > Logon** group.
10. Select **Automatic logon with current user name an password**.
11. Click **OK**.
12. To the confirmation message, click **Yes**.
13. In the **Internet options** window , click **OK**.

Now update the registry.



Note

Serious problems might occur if you modify the registry incorrectly. Therefore, make sure that you follow the following procedure carefully.

Epicor recommends you to back up the registry before you modify it. After editing, you can restore the registry if a problem occurs. For more information about how to back up and restore the registry, refer to the [Microsoft Knowledge Base](#).

1. Click **Start > Run**.
2. Type **regedit**, and click **OK**
3. In the **Registry Editor**, navigate to the following registry key:
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\MSV1_0
4. Create the following value of Multi-String type:
BackConnectionHostNames
5. Add 'scshost' value to **BackConnectionHostNames**.
6. Click **OK**.
7. Click the following registry key:
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa
8. Right-click **Lsa**, point to **New**, and then click **DWORD Value**.
9. Type **DisableLoopbackCheck**, and then press ENTER.
10. Right-click **DisableLoopbackCheck**, and then click **Modify** .
11. In the **Value data** box, type **1**, and then click **OK**.
12. Quit Registry Editor.

13. Restart your computer.

Workshop Constraints

The following workshops should only be performed once in your Epicor Service Connect environment.

- **.NET References Import - Workshop - Import .NET Reference**
- **Database Operations - Workshop - DBOperation Workflow Element - Import .Net Reference**

To complete the **Workshop - Use Common Workflow Techniques - Use the Task Monitor**, the configuration steps described above in the Task Monitor section must be performed.

To complete the **Workshop - Use DBOperation Workflow Element**, OLE DB connection to the application database must be set up in your environment.

To complete the **Workshop - Import .NET Reference** and **Workshop - Use DBOperation Workflow Element - Import .Net Reference**, the Epicor server must be available and at least one Epicor license must be available.

Hardware Requirements

The following hardware configuration is required:

- Intel® Core 2 Duo starting from 2.4 GHz or better
- 4GB RAM or more
- Gigabit network connection

Software Requirements

Epicor Service Connect requires specific software to run optimally.

Epicor Service Connect Server Software Requirements

The following server operating systems are supported:

- Microsoft® Windows® Server 2008 Standard Edition SP2
- Microsoft® Windows® Server 2008 Enterprise Edition SP2
- Microsoft® Windows® Server 2008 R2 Standard Edition SP1
- Microsoft® Windows® Server 2008 R2 Enterprise Edition SP1
- Microsoft® Windows® Server 2012 Standard Edition
- Microsoft® Windows® Server 2012 R2 Standard Edition

The following Database Management Systems (DBMS) are supported:

- Microsoft® SQL Server® 2008 Standard Edition SP3
- Microsoft® SQL Server® 2008 Enterprise Edition SP3
- Microsoft® SQL Server® 2008 Express Edition SP3
- Microsoft® SQL Server® 2008 R2 Standard Edition SP2
- Microsoft® SQL Server® 2008 R2 Enterprise Edition SP2
- Microsoft® SQL Server® 2008 R2 Express Edition SP2
- Microsoft® SQL Server® 2012 Standard Edition SP1
- Microsoft® SQL Server® 2012 Enterprise Edition SP1
- Microsoft® SQL Server® 2012 Business Intelligence Edition SP1
- Microsoft® SQL Server® 2012 Express Edition SP1

- Microsoft® SQL Server® 2012 SP1 Native Client

The required redistributable package is provided in ESC Setup package. ESC setup installs appropriate package, if it is not found on the workstation.

- Microsoft® SQL Server® 2014

Epicor Service Connect supports any configuration of Microsoft SQL Server:

- SQL Server 2008 x64 and x86 are supported
- SQL Server 2012 x64 and x86 are supported
- SQL Server 2014 x64 and x86 are supported
- All collation types for the SQL server are supported: any dictionary order, character set, case-sensitive and case-insensitive search.

The following software is required as well:

- Microsoft Internet Information Services (IIS) 7.0, 7.5, or 8.0 (common files and WWW Server are required; SMTP and FTP are optional)
- Microsoft Message Queuing Services (MSMQ) 2.0 (optional)
- IBM® WebSphere® MQ Server or Client 5.1 or above (English version only, optional)
- SonicMQ v7.6 CSharp Client for Windows (optional)
- Windows Service Bus 1.0 (optional)
- Microsoft .NET Framework 4.5 (full)

Epicor Service Connect setup does not install the package if it is not found on the system.

- Microsoft Web Services Enhancements (WSE) 2.0 SP3 is a prerequisite for Epicor Service Connect. It ships with Epicor Service Connect and is installed, if necessary, at the beginning of installation.
- Microsoft XML Parser (MSXML) 6.0 SP1. The required MSXML 6.0 SP1 redistributable package is provided in ESC Setup package. ESC setup installs appropriate package if it is not found on the system.
- One of the following internet browsers:
 - Microsoft Internet Explorer 7.0, 8.0, 9.0, 10 or 11. Typical set of components.
 - FireFox 25.0
 - Chrome 30.0.1599.101m
 - Safari 5.1.7

Integration between Service Connect and Windows SharePoint Services 2013, SharePoint Foundation 2013 and older versions are supported.



Note Epicor Service Connect installer does not add https binding in IIS. If there is no https binding, add it manually.

It is required to work with SCIntegration service (WCF version).

Epicor Service Connect Client Software Requirements

The following workstation operating systems are supported:

- Microsoft Windows Vista (Business, Enterprise or Ultimate Edition) SP2
- Microsoft Windows 7 (Professional, Enterprise or Ultimate Edition) SP1
- Microsoft Windows 8 (Professional or Enterprise)
- Microsoft Windows 8.1 (Professional, Enterprise)

The following software should be installed on the workstation:

- One of the following internet browsers:
 - Microsoft Internet Explorer 7.0, 8.0, 9.0, 10 or 11, typical set of components.
 - FireFox 25.0
 - Chrome 30.0.1599.101m
 - Safari 5.1.7
- Microsoft SQL Server 2012 SP1 Native Client. The required redistributable package is provided in ESC Setup package. ESC setup installs appropriate package, if it is not found on the workstation.
- Microsoft XML Parser (MSXML) 6.0 SP1. The required MSXML 6.0 SP1 redistributable package is provided in ESC Setup package. ESC setup installs appropriate package, if it is not found on the workstation.
- Microsoft .NET Framework 4.5 (full) must be installed. ESC setup does not install Microsoft .NET Framework, if it is not found on the workstation.
- To develop workflows using Windows Workflow Foundation, Visual Studio 2010 should be used; refer to Visual Studio 2010 documentation to learn more on how to develop WWF workflows.



Tip For the latest Service Connect software requirements refer to the Epicor Service Connect Installation and Implementation Guide.



Important .NET extensions must be allowed for Service Connect to work correctly. After you install ESC, make sure .NET extensions are allowed.

To allow .NET extensions, in the Internet Information Services (IIS) Manager, open the ISAPI and CGI Restrictions feature. On the ISAPI and CGI Restrictions page, for each .NET extension, set the Restriction to Allowed.

Overview

Epicor Service Connect is a business integration platform for secure workflow orchestrations within Epicor applications or for external connectivity to Epicor and non-Epicor applications. You can recompose business components represented as .NET business objects or web services outside the application within Service Connect to eliminate non-value-added steps or potentially speed up any business process.

Epicor Service Connect:

- Highlights the value of a Service Oriented Architecture
- Automates tasks and processes in the application to promote lean principles, continuous performance initiatives, and quality for an organization or across the supply chain
- Allows employees to focus on value-added activities and management by exception, instead of repetitive data re-entry tasks

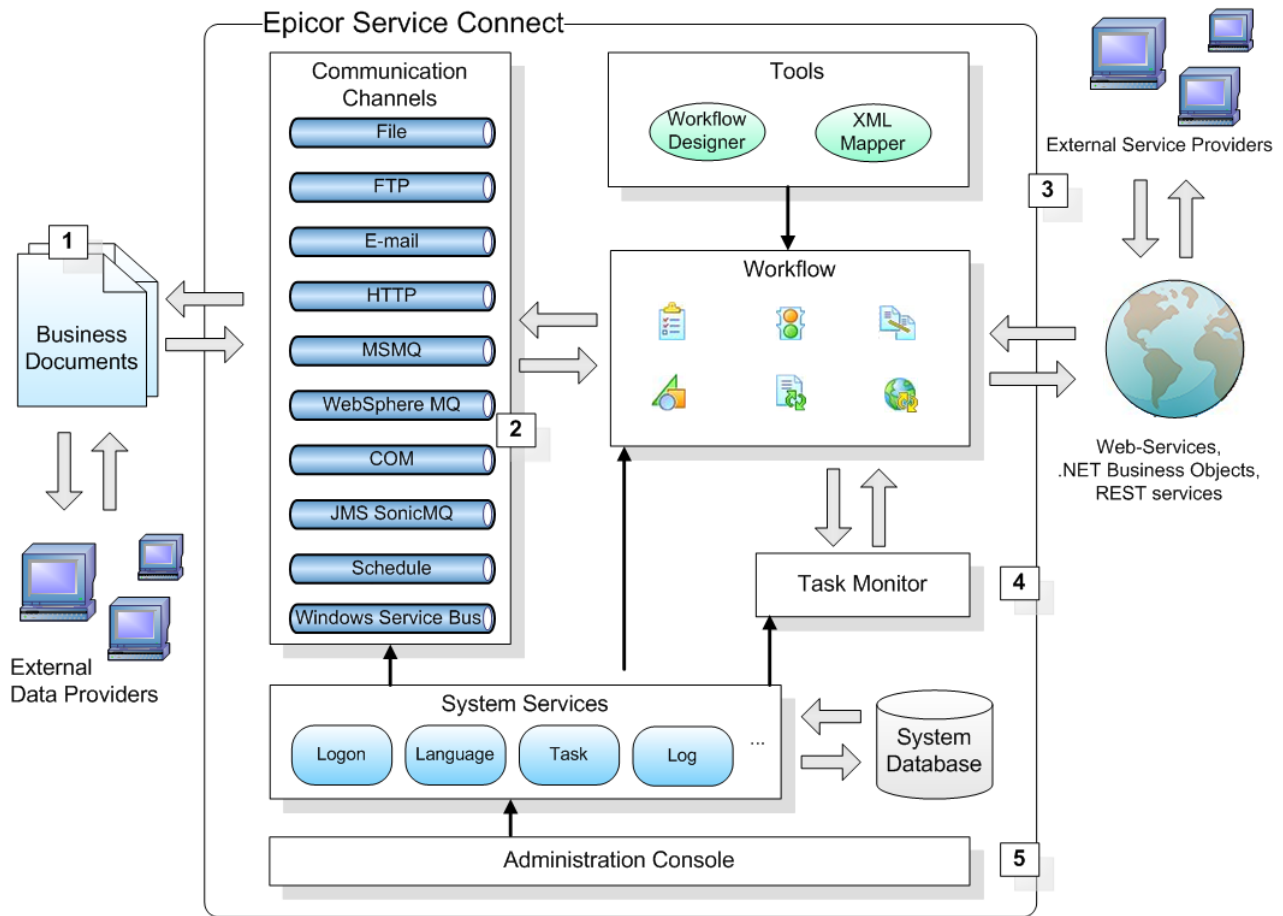
Epicor Service Connect Architecture Overview

Epicor Service Connect consists of four major parts:

- The **Epicor Service Connect Administration Console** is a Microsoft Management Console and is used to configure and maintain settings for Service Connect Framework services.
- The **Service Connect Framework** is a collection of services that supports all Service Connect functions, such as running communication channels, executing workflows, and tracking documents and system events.
- The **Task Monitor** is a web interface to view, track, and process user tasks assigned during workflow execution.
- The **Workflow Designer** is an environment to create and edit workflows.

Use these components to create and run custom, flexible workflows that accept different data formats and to use various .NET assemblies and services available with web service technology to process data.

The following graphic shows an architectural overview of Epicor Service Connect:



1. A document is dropped into a predefined area. The area can be a file folder, email inbox, or message queue. A Service Connect input channel monitors the area for new documents. When it receives a document, it converts it to XML and adds message attributes to it. The attributes indicate a document source and document type.
2. Service Connect reads the document attributes and verifies whether there is a message map with the corresponding attributes. If so, Service Connect routes the document to the workflow on the map or to another service registered in the Epicor Service Connect Administration Console.



Example You can use the Part .NET assembly to add or edit a part, and it does not necessarily require a workflow to tell the .NET assembly how to do this. Service Connect reads the document received on the input channel and converts it into action items based on the message type (schema).



Note If you call a Service Connect workflow directly from a Business Process Management directive, the document is sent straight to the workflow. There is no need for an input channel, message attributes, or message map.

3. Once the document is in a workflow, you can use it in several workflow elements that:
 - Convert the document to other formats.
 - Call .NET assembly.
 - Send email notifications or perform different output action.
 - Create tasks that require a user response.
 - Call other workflows to perform additional processing.

- Automate routing based on the information in the document.
- Perform the operation against the application database.



Example One example where Service Connect is helpful is when you process sales orders. This typically involves multiple availability inquiries, reviews, inventory release decisions, and so on. Orchestrating this process within the Service Connect Workflow Designer can eliminate many of these steps by routing processes to automated tasks, such as order-submit-direct-to-pick for specific inventory items or order fulfillment for a user's best customers.

Other examples include product lifecycle management (PLM) integration without entering change orders within a process workbench in the application, processing email attachments for automatic data input, and using task assignments with digital signatures to follow corporate governance of best practices.

Another example is sending an email to the customer's sales account manager when a credit hold is placed on a customer.

Service Connect Technology

Service Connect supports specific, collaborative processes, connecting different business entities, applications, or users. Open, industry-wide standards and technology are used to enable businesses to deploy solutions quickly.

Service Connect uses a Microsoft Distributed interNet Architecture (DNA) approach supporting .NET concepts.

Service Connect includes:

- Support for business documents based on open XML standards.
- Listeners and Posters that can connect to a number of communication protocols including COM, FILE, FTP, HTTP, email (SMTP), IMAP, IBMMQ, POP3, SonicMQ files, Microsoft Message Queue and Windows Service Bus.
- The Data Exchange Server (DES) and the Service Connect Access Server that links processes.

Business Documents and the Internal Message Envelope

Service Connect typically receives business documents through an input channel and processes them in a workflow. To process a document, Service Connect can generate new documents and distribute information to external recipients.

- Service Connect can receive and send documents in several formats, but it uses only XML documents internally.
- If an incoming document is in XML format, the XML is stored in the internal message envelope, which is an XML structure Service Connect recognizes. You must convert any documents that are not in XML format to XML and then wrap them in the internal message envelope.
- Service Connect provides plug-ins that can transform the following document formats to the internal message envelope format:
 - CSV (Comma Separated Value)
 - XML
 - Microsoft Office Excel (xls andxlsx)



Note

Service Connect only processes Excel files without protection.

- You can convert documents sent out from a workflow from the internal message envelope to CSV or Excel if necessary.
- To support conversions to other document formats, you can create custom plug-ins in XSLT, VB, or C#.

Use Communication Channels to send and receive documents. These channels leverage various communication protocols to enable document exchange between Service Connect and other applications. When a communication channel receives an incoming document, Service Connect converts it to XML and wraps it in the internal envelope format. Incoming document information is stored in the business data node of the internal message envelope. The business data node (**dta** node) contains the incoming document data.

Other nodes in the internal message envelope store system information, such as details about who sent the document to the input channel or the type of action taken on the document. The **RefID** node holds a reference to the message that initiated the exchange. The **MsgID** node stores the ID of the current message in the set of linked messages. The **Ack** node contains the flag that specifies whether an acknowledgment of sending is required.

Implementation

The following provides an overview of how to implement Service Connect.

Import References

Use the Epicor Service Connect Administration Console to import .NET, service, or REST web service references. Imported references, such as .NET assembly references are available for use in workflows and automatically register business documents message exchanges used between the .NET object methods and Service Connect.

Set Up Message Attributes

Message attributes are added to documents submitted to an input channel. They include message types and senders. Message types classify the different documents Service Connect can process. Senders define the origin of documents sent to Service Connect.

Message maps determine where to route a document once the document is received in an input channel based on the message attributes on the document.

Set Up Communication Channels

Use communication channels to send and receive documents. You must configure channels before you use them. You can define the following options when you configure channels:

- Communication protocol and protocol specific settings, such as the folder path for a file channel
- Message encoding and formatting
- Channel schedule
- Default message attributes

Create Workflows

A workflow is a diagram that tells the application how to process a business document. Use the Workflow Designer to create and edit workflows and define a sequence of automatic actions and user-performed tasks. General activities include:

- Document transformation
- User task assignments
- Business specific actions incorporated into workflows by importing services that provide required functionality

Workflows are organized in packages, which provide a convenient way to group workflows. You can create packages in the Epicor Service Connect Administration Console and the Workflow Designer.

Use the Task Monitor

Tasks are workflow elements that can halt a workflow under certain circumstances and send data to the Task Monitor website where an authorized user takes action and selects how to route the information. Two typical examples when the Task element is used are in authorization scenarios (for example, establish a credit limit) or error handling (for example, continue processing or halt processing).

Epicor Service Connect Environment

Epicor Service Connect (ESC) is a workflow and application integration environment.

The environment consists of:

- Epicor Service Connect Administration Console
- Workflow Designer
- Task Monitor
- Service Connect Windows services

The following are important Epicor Service Connect terms:

- **Document** - A document is information sent to, from, or processed by a Service Connect workflow. Documents can be .xml files, .csv (comma separated values) files, or .xls (Microsoft Excel) files. Use custom conversion plug-ins to support other document formats. Service Connect converts all incoming documents to a standard XML format referred to as the internal envelope. The parts of an XML document are referred to as nodes.
- **Workflow** - Workflows orchestrate automated processes. Workflows manipulate documents from an application in order to pass along the data to another application. You can manipulate a document to call out to another application in the middle of the workflow or to produce a new document by the end of the workflow.
- **Channel** - Channels are similar to telephone lines that receive or send information. Channels are configured in the Epicor Service Connect Administration Console to receive documents for workflow processing. Optionally, you can configure channels to send out documents to other applications. Channels that receive documents are referred to as **input channels**, and channels that send documents are referred to as **output channels**.
- **Message Attribute** - Message attributes are added to documents submitted to an input channel. There are two message attributes:
 - **Message Type** - Use message types to classify different documents you plan to process in Service Connect.
 - **Sender** - Use senders to define the origin of documents sent to Service Connect.
- **Message Map** - Message maps route incoming documents to the appropriate workflow.
- **Reference** - ESC uses three types of references: .NET References, RESTful References and Service References. References are configured links to .NET assemblies, RESTful web services, published web services or Windows Communication Foundation® (WCF) services outside Service Connect. After you add a reference in Service Connect, you can call .NET object methods, RESTful service methods or web service methods from inside workflows.
- **Message Extension** - Also known as containers, message extensions store data so it is available to workflow elements after a .NET call. Data can be information passed into the workflow or derived from other workflow activities, such as a Conversion or a prior .NET call.

Epicor Service Connect Administration Console

The **Epicor Service Connect Administration Console** is a console where you can manage one or more Service Connect server installations.

Start > Programs > Epicor Software > Epicor Service Connect > Service Connect Administration Console

The console is organized into several expandable and collapsible nodes. Each node manages a particular part of a Epicor Service Connect installation. The following topics discuss the Epicor Service Connect Administration Console nodes.

Activities

Use **Activities** to view users who are logged onto the Epicor Service Connect Administration Console or Workflow Designer to see their status and log them off if necessary.

Right-click a user to view the menu options.

The available options include:

- Logoff User
- Logoff All Users
- Refresh
- Help

Companies

Use **Companies** to set up a company information in Epicor Service Connect.



Important Only iScala Service Connect implementations currently use companies.

When adding a company information, enter a valid **Company code** and a **Company name**.

Document Tracking

Use **Document Tracking** to trace the progress of Epicor Service Connect workflows. You can view certain document metadata and the message data at various execution points, including each element within a workflow. This can be valuable for troubleshooting during your workflow development and also in your production environment.

During the process, the document passes through various Service Connect components, such as input and output channels, workflows, tasks, and so on. Each Service Connect component can change the document when performing a certain activity.

The document tracking system supports the export and import functionality. To export a trace, in the grid, right-click the trace and select **Export Activity**. You can also make a multiple selection and export more documents at once. To export all messages within a folder, right-click a folder, for example, Inbound Messages and select **Export All Activities**. To import trace(s) into the ESC Administration Console, select the folder, for example, Inbound Messages, right-click in the grid and select **Import Activities**.



Example You can export several traces, load them locally and evaluate them. This functionality is beneficial for the technical support purposes.

Double-click the trace to display the **Activity Progress** window. This window displays the detailed information about the activity. If the activity is paused, you can select an execution point and click the **Resume Activity** button. Use the **Delete Activity** button to delete the selected activity.

To quickly identify the reason of the error and examine a complex workflow, in the **Activity Progress** window, use the **Find** button.

If an incoming document fails to comply with a schema, it may be difficult to identify the error, generated by the workflow. To validate the incoming message against the schema, double-click an execution point to display the **Trace Details** window. Navigate to the **Processing Info** sheet, click the **Validate** button and view the information.

Languages

Use the **Languages** node to set up a language for your Epicor Service Connect installation.

The Epicor Language Server allows you to:

- Set language properties for a particular installation.
- Change the language order and inheritance.
- Import extra languages.

Tasks

Tasks are workflow elements that can halt a workflow under certain circumstances, and then send data to the Task Monitor web site, where an authorized user can take action and possibly select how to route the information.

You can assign tasks to individual users or groups.

To view the **Task Details** window, double-click a specific active or a completed task.

Use the **Attributes** node to set up a new task attribute.

Connectivity

The **Connectivity** node holds site-specific and user-generated information for a particular Epicor Service Connect (ESC) installation.

The following sections describe the connectivity components.

Asynchronous Pools

Use asynchronous pools to improve the flexibility for the threads you use for the workflow execution. You can manage several customer thread pools for asynchronous execution and assign them to the workflows. It results in availability of real load balancing configuration.

Workflow Packages

The Workflow packages node displays workflows for the local ESC installation.

Message Attributes

Add attributes to documents that enter ESC through an input channel. You can also assign these attributes to message maps, so when the document attributes match a map, ESC can route the document to a workflow or other service.

Within the Message attributes node, specify the following information:

- **Message Types** - Define document types sent to ESC and the expected action for the document. For example, you can define a Sales Order message type with an Update action.
- **Senders** - Define who is sending the document to ESC. For example, you can define a Customer sender and then add a sub-sender for each customer who submits documents.

Message Map

Message map is a combination of message attributes associated with a workflow. When a document enters an input channel, ESC looks for a map with the same message type, action, and sender on the document and routes the document to the appropriate workflow.

.NET References

.NET References are configured links to specific .NET assemblies. After you add a .NET reference in the ESC Administration Console, you can call .NET object methods within the Workflow Designer.

When you add a new .NET reference, first select the **Assembly type**. The available options are:

- Generic assembly
- Epicor 9.05 assembly
- Epicor 10 assembly

Then specify the **Assembly path** and the **Reference name**.

When you add an Epicor 10 assembly reference, the following are the required fields you must specify to successfully add the reference:

- The path to the Epicor client configuration file
- User
- Password
- Company
- Plant

When you access an Epicor Business Object and open Epicor session, the following Debug entry is added to the Event log.

```
Epicor session is open
  user:<UserName>
  server:<Server>
  port: <PortNumber>
  company:<Company>
  site:<site>
  license:<License Type>
```

In the list of .NET references in Epicor Service Connect Administration Console, references to Epicor 9.05 .NET business objects have a simple icon. References to Epicor 10 .NET business objects have an icon with an E letter on it.

Service References

Service references are configured links to the published application services, such as web services or Windows Communication Foundation® (WCF) services, which are outside of Service Connect. Adding service references in the ESC Administration Console allows you to pass and receive datasets to and from the web methods from inside workflows.

RESTful References

You can import RESTful web services to call them from inside workflows.

RESTful web services do not have a standard functionality to expose metadata describing their API. Epicor recommends that you use documentation provided by the service developer in order to discover an API and supported data formats. A part of this documentation is sometimes exposed as a help page, usually available by /help relative URI. The API documentation is the only way to discover exposed resources, their URI templates, HTTP methods available on them, and their representation formats unless metadata is exposed.

You can import a RESTful Reference manually. In this case you define all resources and operations available on the service. If a RESTful web service exposes metadata, you can use it to add the service automatically. RESTful web services expose metadata as a simple xml file which describes the whole API structure: schemas, parameters, and so on.

Schemas

Use the Schemas node in the Epicor Service Connect Administration Console to perform schema-related tasks.

You can perform the following tasks:

- Browse through XSD schema files and folders.
- Create or delete schema files and folders.
- Rename schema files and folders.
- Generate an XSD schema based on the XML provided.
- Generate an XML based on the selected schema, registered in the system.



Note Schema snap-in supports multi-site SC installations.

Communication Setup

Use the Communication Setup node to set up the channel information. Channels are the entry and exit points for documents going into and out of Service Connect.

The following are the nodes that display within the Communication Setup node:

- **Channels** - Use the Channels node to set up Input and Output channels.

The following Input channel types are available: COM, File, FTP, IBMMQ, IMAP, POP3, MSMQ, IBMMQ (.NET) Schedule, SonicMQ, and Windows Service Bus.

The following Output channel types are available: COM, File, FTP, HTTP, IBMMQ, SMTP, MSMQ, IBMMQ (.NET), SonicMQ and Windows Service Bus.



Tip You can configure the channel as **high-loaded**. The system tries to allocate more resources (separate thread per channel) for a high-loaded channel to improve performance and to balance the overall load.

- **Conversions** - These XSLT transforms, COM Objects and .NET classes convert incoming documents to the internal envelope message format and convert outgoing documents to an external format. You can also register new conversions in this node.
- **Failed Input** - This node holds documents that failed to enter Service Connect through an input channel.
An input channel was unable to handle these documents after a certain number of attempts.
- **Failed Output** - This node holds documents that failed to leave Service Connect through an output channel.
An output channel was unable to handle these documents after a certain number of attempts.

Backup and Restore

Use the **Backup Connectivity Settings** window to backup the Service Connect data, such as workflow, schemas, service references and so on.

To access the **Backup** and **Restore** options, right-click the Connectivity node.

Use the Backup option to backup and save all or part of the Connectivity settings.

Use the Restore option to copy the settings to another server or to the same server. You can overwrite existing settings with the same name. The restore is flexible, so you can restore and reconfigure specific information.



Tip After you re-import a Web Service or .NET reference, you must use parameters specified during the first import to regenerate it. This task is necessary when a Web Service or .NET reference changes its interface. Use the Run Re-import Service References wizard upon restore completion check box in the Restore Connectivity Settings wizard to automatically regenerate service references.

Security

Use the **Security** node to maintain the general security settings for Epicor Service Connect

The available options include:

- Users
- Enterprise Organigram
- Granted Permissions
- Role Assignment

The following BuiltIn roles are provided for Epicor Service Connect users to simplify user rights assignment:

- System Administrators
- Workflow Administrators

Events

The Events node includes event logging functionality setup, such as the start up and services closing.

The information displays in the following three nodes:

- All logs
- Administration
- Archive



Tip Use the information provided by event logs while debugging.

Locks

Use the **Locks** node to identify locked files.

The Logon ID, Session ID, Resource ID, and Key data are included in the lock information.

Locking is required to keep business data consistent. Objects like tables can be locked automatically in the event of system failure to prevent data loss.

The following actions are available to you:

- Customize the Lock list view.
- Check or specify the SQL server and database name.
- Select Persistent Lock Settings from the context menu.
- Export the Lock list (save the list as a separate file).
- Unlock selected objects.



Tip To release a locked object, select a locked record in the right pane, and then select Delete from the context menu.

Licensing

Use the **Licenses** node to import and manage Epicor Service Connect (ESC) licenses.

Licenses include **System** and **Business Features** licensed in the ESC application.

To import a license, right-click the **Licenses > Licenses** node and select **Add License**.

To easily see what functional items are licensed, under the **Features > System Features** or **Features > Business Features** node, sort by the Description column. For example, you need to find out if Workflow Element: DB Operation is licensed. If it is not licensed or is expired, it will not appear in the list or its Expiry Date would be in the past.

Epicor .NET Business Objects

A business object contains the code that runs a business process. There are hundreds of .NET business objects that control the Epicor 10 application functionality.

.NET assemblies are platform independent. You can use a wide range of customizations and create your custom functionality in any .NET language. This saves a lot of time and effort in exploiting existing infrastructures. Microsoft .NET Framework provides a powerful means to work with .NET assemblies.

You can call Epicor 9.05 and Epicor 10 .NET assemblies from Service Connect. Calling .NET assemblies improves performance because in this case you call Epicor .NET object directly.

No additional installation steps are required to use Epicor 10 .NET Business Objects. To call Epicor 10 .NET Business Objects methods in Epicor Service Connect workflows, add .NET references in ESC Administration Console.

In the list of .NET references in Epicor Service Connect Administration Console, references to Epicor 9.05 .NET business objects have a simple icon, and references to Epicor 10 .NET business objects have an icon with an E letter on it.

References

References are configured links to .NET business objects, REST services, published web services (WSE) or Windows Communication Foundation® (WCF) services, which are outside of Epicor Service Connect (ESC). Add .NET references, RESTful references or Service references in the ESC Administration Console, and then you can pass and receive datasets to and from their methods from inside workflows. A wizard helps to import references, configure security and handle other miscellaneous details regarding the exchange of datasets.

When you create a .NET reference, RESTful reference, or Service reference, the reference name you enter is very important. The workflow uses this name to reference the .NET assembly, RESTful service, or web service. ESC provides the default name same as the .NET assembly, RESTful service, or web service name. You can assign any name to a reference, but it is recommended that you use the default name.

For any web service, .NET assembly or RESTful service you have the option to import all or just a subset of its methods. For each method, Service Connect creates two schemas: a request schema and a response schema. The request schema contains the document structure the method expects when it is called. The response schema contains the document structure the method returns.

To locate the request and response schemas for every method and to view the only location where imported references are stored, use the following path: C:\Program Files\Epicor Service Connect\SCS\Schema\.

After you add a reference, you can perform a connectivity check of its methods. To perform the test, in the Epicor Service Connect Administration Console, navigate to the Connectivity > Service References, or .NET References, or RESTful References node. Select the reference. From the list of its methods, right-click a method and select Test. This method uses the schema that was created during import to generate a sample request xml, which you can further analyze. This request xml is a message template. You must replace the placeholders with real data before you click the Test button to actually send the request.



Note If you are backing up and restoring the system or using workflow packages created somewhere else, ensure the service name in the Add Service References utility is exactly the same as the name used when the web service was created. The same applies to reference name in the **Add .NET Reference** and **Add REST Reference** utilities.

The handler selected during the Add Service References step provides the username and password web services use to access the Epicor application. In some cases, you must use the **Security** sheet to define the server credentials.

Workflow Designer

Use the Workflow Designer environment to create and edit workflows. Workflows are graphical representations of a set of interconnected data operations, known as workflow elements. Each workflow models and supports a real-life business process. A designated icon represents each element. Elements are linked together with connectors that show the operation sequence.

Start > Programs > Epicor Software > Epicor Service Connect > Workflow Designer

When you create a new blank workflow, the designer includes a workflow design surface with a Start element and a Finish element. All workflows are stored in groups called **workflow packages**. The Workflow Designer allows you to customize the appearance of the items in the diagram to make the diagram easy to understand.

The following sections explain the workflow elements you can use to create a workflow.

Pointer

Use the Pointer tool to select components already added to a workflow. To select more than one item, hold the **Ctrl** key while you click items. Alternately, click and hold the mouse button and drag the Pointer to select a group of items.

Connector

Use the Connector item to connect two workflow elements. You must connect all the workflow elements before the workflow will function. In the Connector Properties, you can add a caption, select a font, and select the line that connects elements. The caption appears as a connector label.

Splitter

Use the Splitter workflow element to simplify and clarify the workflow layout or to apply a schema to a document. Splitters can have more than one inbound connector and more than one outbound connector. The only functional property of the Splitter element is the schema.

Task

Use the Task workflow element to issue an assignment to a user in the Service Connect Task Monitor. In the Task Monitor, you can review incoming documents and edit them, if permitted. Then, you can route the document to one or multiple outbound connectors.

Poster

Use the Poster workflow element to publish XML documents from a workflow. You can post a document through several channels, such as a message queue, file system, FTP site, or email message based on the Output Channel(s) assigned to the Poster. If the document is posted to a location a Service Connect Input Channel monitors, you can use the document to trigger a separate workflow. If an Output Channel assigned to the Poster is an SMTP channel, you can use the document in an email message to notify users about events that occur within a workflow.



Note If you add a Poster element to the workflow and modify the channel configuration in the Workflow Designer, the customized poster icon displays instead of the default one.

Sub-Workflow as a Workflow Call

Use this workflow element to call a workflow as a subroutine for another workflow. You can set the sub-workflow to run asynchronously (the main workflow continues to execute) or synchronously (the main workflow pauses until the sub-workflow finishes). When a sub-workflow is set to execute synchronously, the Sub-workflow results are available in the following workflow element of the main workflow.

You can set the sub-workflow to execute once or to cycle through specific nodes in a document.



Example If a document contains a sales order, you can set up a sub-workflow to cycle through each sales order line item.

If necessary, you can send the data stored in message extensions for use in the sub-workflow. To improve the performance of such business processes, run such sub-workflows as parallel threads, in cases when the business logic permits. System administrators can select if this functionality should be used or not, and it will depend on the underlying business process and hardware configuration.

You can use sub-workflows to validate information sent to a workflow or to retrieve information required for future workflow operations.

You have the ability to break the sub-workflow process based on an exception raised in one of the looping processes without having to complete all the process loops. For example, complete processing of an input file or a child record set. This can save time and resources in case of cycles with many iterations.

WF Workflow element as a Sub-Workflow Call

Service Connect is integrated with Windows Workflow Foundation®. Windows Workflow Foundation is a Microsoft technology that defines, executes, and manages workflows. Integration of Service Connect and Windows Workflow Foundation provides the capability to use workflows, created by Visual Studio® 2010 as sub-workflows of Service Connect workflows.

This technology is part of .NET Framework 3.0. Windows Workflow Foundation is the programming model, engine, and tool used to quickly build workflow enabled applications in Windows. It consists of a .NET Framework version 3.0 (formerly WinFX) namespace, in-process workflow engine, and designers for Visual Studio 2010. Windows Workflow Foundation includes support for both system workflows and human workflows across a wide range of scenarios including:

- Line-of-business application workflows
- User interface page-flow
- Document-centric workflows
- Human workflows
- Composite workflows for service oriented applications
- Business rule driven workflows
- Systems management workflows

For more details on Windows Workflow Foundation, refer to the Microsoft website (<http://msdn.microsoft.com>).

Requester

Use the Requester workflow element to support request and response communication with an external system during workflow processing. The external system is usually another application, but it can also be another workflow. The Requester workflow element forwards the incoming message to output communication channels in the same way as a Poster and then waits for a response. The response returns to the workflow through an input channel and is routed to the Correlation Manager according to its message map settings. When the response is received or the timeout expires, the suspended workflow resumes.

A Requester relies on two components - the Requester workflow element and the Correlation Manager. A Requester posts a message similarly to a Poster and adds attributes to the message that identify the Requester and the workflow that posted the message. When a response returns to the system, the input channel that receives the message is unaware it requires special handling. To route the message back to the waiting Requester, the message is sent to the Correlation Manager. The Correlation Manager analyzes the attributes the Requester added and resumes the workflow from the Requester workflow element.



Note If you add a Requester element to the workflow and modify the channel configuration in the Workflow Designer, the customized requester icon displays instead of the default one.

This workflow element is rarely used. If you need information from another application, it is usually provided by a .NET business object, web service or another service registered in the ESC Administration Console as a Reference.

.NET Call

Use the .NET Call workflow element to call any method of any .NET object registered in a workflow using standard workflow designer engine. This feature allows you to extend the scope of functionality within workflows and it gives you freedom to create functionality in any .NET language.



Note If you add a .Net Call element to the workflow and modify the Epicor logon configuration in the Workflow Designer, the customized .NET Call icon displays instead of the default one.

Web Method

Use the Web Method element to call a Web Service method. You must import the method as a Service Reference in the ESC Administration Console before you can use it in a Web Method workflow element. Web Methods are the primary method to communicate with target databases.

When you add a web method, view the handlers in the **Default** configuration field of the web method **Properties** window to override service credentials.



Note If you add a Web Method element to the workflow and modify the handlers configuration in the Workflow Designer, the customized Web Method icon displays instead of the default one.

Conversion

Use the Conversion workflow element to convert a document from one format to another and to store values in message extensions or variables. The Conversion defines an Input Schema (the format of the incoming document) and an Output Schema (the format of the document passed to the outbound Connector). After you set the Input and Output Schemas, use the **XML Mapper**, a visual interface to map elements from the Input Schema to the Output Schema, to create the conversion.

To convert the file, map nodes from the incoming document to nodes in the target document. The XML Mapper shows both the incoming and target documents as expandable trees. Mappings between nodes are represented as lines that connect the nodes in the incoming document to the nodes in the target document. The XML Mapper interface is similar to many other data mapping tools. In addition to the graphical interface, the XML Mapper also has an XSLT view, where you can view and edit the source code directly.

To open the XML Mapper, click **Edit** next to the **Conversion** field in the conversion **Properties** dialog.

There are four basic types of nodes:

- **Simple** - Represent XML elements that contain parsed character data (PCDATA)
- **Complex** - Represent XML elements that can contain PCDATA and other element
- **Attribute** - Provide additional information about an element node, appear as child nodes to their respective elements
- **Fake** - nodes are nodes in the target document mapped to a node in the incoming document, but the target node no longer exists

To view only the fake nodes, in the XML Mapper View menu select Show only fake nodes.

Simple nodes are typically mapped to each other using a one-to-one relationship, or many-to-many if the nodes represent collections.

Complex nodes are typically mapped to each other using a deep copy, which means all the child node values of the complex node in the incoming document are automatically copied to the child nodes of the complex node in the target document. For a deep copy to function, the structure of the complex nodes must be the same in both documents.

There are two types of complex nodes:

- **Single complex node** - This element can appear only once within its current context. If the element has child nodes, you can expand and collapse it.
- **Complex collection node** - This element can appear multiple times within its current context. XML Mapper provides functionality to add an arbitrary number of collection items to the output schema and set values for them.

If the element has child nodes, you can expand and collapse it.

**Tip**

As a rule, you create for-each loop only between collection. If either left or right node is not a collection then the for-each is not directly available because of the following reasons:

- If the left node is not a collection then only one node can be evaluated and then this looping makes no sense
- If the right node is not a collection then looping creates several output nodes in place where only one is available. Such output xml does not correspond to scheme and can be rejected.

To create a for-each loop between nodes that are not collections, drag the link with SHIFT key pressed. The Mapper displays a warning notifying that the selected source schema node is not a collection.



Tip To only display nodes mapped between the incoming and target document and nodes holding literal values, in the **XML Mapper** tool, from the **View** menu select **Show only mapped nodes**.

You can enter **Functoids** in the center pane of the XML Mapper window. Functoids are a collection of XPath functions and extension functions that can perform a variety of tasks, such as comparisons, mathematical operations, data type conversions, and others. You can use the Functoid Palette to add Functoids to a Conversion. Some functoids require one or more nodes be mapped to it from the incoming document. You can map the functoid result to another functoid or to a node in the target document.

Condition

Use the Condition workflow element to test an incoming document against an automated processing rule and potentially halt the document path if it does not satisfy the rule. If the document satisfies the processing rule, the Condition passes it to the outbound Connectors. Although you can attach the Condition to more than one outbound Connector, the document is evaluated against only one rule. To evaluate a document against more than one rule, use a Choice workflow element.

A Condition rule is an XPath expression evaluated against an incoming document as true or false. If the document satisfies the rule, the system passes the document through the outbound Connector(s). If the document does not satisfy the rule, the document path is stopped, which can stop the entire workflow.

Choice

Use the Choice workflow element to route a document based on automated processing rules. This workflow element has one inbound Connector but can have several outbound Connectors. Each outbound Connector is associated with a processing rule. A document that enters the Choice element is evaluated against the rules to

determine which outbound Connector the document follows. You can configure Choice workflow elements to send the incoming document along multiple outbound Connectors or just one.

Each rule is an XPath expression evaluated against an incoming document as true or false. Documents can progress through each outbound Connector for which they satisfy the rule. To restrict the passage of a document to only one outbound Connector, clear the **Allow multiple exits** check box.

Rules are applied to the incoming document in the order they appear on the **Rules** sheet of the Choice **Properties** dialog. Thus, in the instance where the incoming document satisfies more than one rule but the document can pass through only one outbound Connector, the document progresses through the first Connector that satisfies the rule. To change the rules order, select a row in the grid and use the **Move Up** or **Move Down** button.

All Choice elements have a Default case rule. If an incoming document does not satisfy any of the rules, the system passes the document to the outbound Connector associated with the Default case rule, which allows the workflow to continue.



Note Similar to the Conversion element, **Choice** and **Condition** elements display a wizard when you browse for a schema.

To specify the incoming document schema, on the **Schema** sheet, click the **Browse** button. You can switch between Web-Services schemas, .NET Reference schemas, REST-Services schemas and User schemas generated in the Schemas node of the Service Connect Administration Console, or in the Generate Schema from Sample Data tool in the Workflow Designer, or the schemas generated in the Workflow Designer when you click the Create sub-workflow schema button on the Cycling tab of Sub-workflow properties dialog.

In the **Root element** field, select the schema root element. In the **Multiplicity** field, select **Single** to use just one element under the dta element, or **Multiple** to utilize a collection under the dta element.

DBOperation

Use the DBOperation workflow element to perform SQL statements against the application database. First, set up the OLE DB Provider to connect to the application database. Once you establish the connection, create and run the SELECT, UPDATE, INSERT and DELETE operations against the database.

If you validate the workflow or save it, and ESC finds an empty statement inside the DBOperation, the validation result dialog displays a warning message. If during input message processing ESC encounters a DBOperation with an empty statement, a warning message is added to the Events log.

Break

Use the Break element in a Sub-Workflow to specify how exactly the main workflow execution should be altered. In the Break element properties, you define an ordered set of rules that are executed against incoming document in order to decide if the looping should continue or not. The Break element passes one of following break codes to the main workflow:

- **Completed**

The sub-workflow looping is interrupted and the trace status is set to Complete.

- **In progress**

The sub-workflow looping is interrupted and the trace status is set to In Progress.

- **Abortive**

The sub-workflow looping is interrupted and the trace status is set to Abortive.

- **Ignore**

The sub-workflow looping continues.

The Break element doesn't stop the execution of the sub-workflow immediately, instead the break code is remembered in process context and execution continues. Thus the Break element should not necessary be followed by the Finish element, but can be placed anywhere the workflow logics requires.

In case you divide execution into pseudo-parallel branches, all these branches should reach the Finish element. If several branches reach the Break element (the same one or several different elements), the "worst" break status code is selected as the resulting one. For example, if the first reached Break element set the Completed status code, and the second Break element set the In progress status, the resulting break code is In progress. But if the first break code was Abortive, than it is not modified to Completed by the following Break elements.

Generate Schema from Sample Data

Use the Generate Schema from Sample Data option to generate a schema directly from Workflow Designer using a sample document.

You can use the following data types:

- **well-formed XML**
- **Comma Separated Value** - (CSV) file
- **Excel** - (XLS, XLSX, XLSM, XLSB) file



Note

Service Connect only processes Excel files without protection.

For more information on how to generate a schema for XML, Comma Separated Input and Excel files, see ESC Help.

Message Extensions and Process Variables

Message Extensions

Workflow Designer > File > Process Properties > Message Extensions

Use Message Extensions, also known as Containers, to store data you want to use several times within one workflow. The data can be information passed into the workflow or derived from other workflow operations, such as a Conversion or prior .NET Call.

Message Extensions are useful because the Input Schema for a .NET Call must always be the request schema for the .NET method, and the Output Schema for a .NET Call must always be the response schema for the .NET method. These schemas only have nodes for the information sent to and from the .NET method, so you must use a container to propagate any other data required later in the workflow.

The nodes in a message extension are available from the **wfl > usr** node of the internal envelope.

Process Variables

Workflow Designer > File > Process Properties > Process Variables

Process variables are similar to message extensions. They store data in the **wfl > usr** node of the internal envelope, so you can access the information when necessary. While message extensions are designed to hold complex data structures based on a schema, variables are designed to hold a single value. In addition, you can assign a data type and a default value to process variables.



Tip When you configure a Conversion element and drop nodes to the <usr> section, XML Mapper creates Process variables and Message extensions automatically. When you map a simple type node to the <usr> section, a new Process variable is created and mapped. When you map a complex type node under the source <dta> section to the target <usr> section, a new message extension in source schema is created.

You can alternatively right-click the right pane and select New process variable or New message extension to create variables or containers.

If you want to select a workflow as a source for Message Extensions and Process Variables, click the Change button next to the Workflow to inherit Message Extensions and Process Variables field. In the Select workflow to inherit Message Extensions and Process Variables window, select a workflow and click Open.



Note The Validate workflow functionality checks for name conflicts between inherited Extensions/Variables and local ones specified in a workflow. In case such conflicts are found, a warning displays. If an inherited variable/extension has the same name as a local one, the local variable/extension will be seen in XML Mapper, XPath builder and runtime.

You can pass Process Variables only in one direction: from the Main workflow to a Sub-workflow. Process variables from a Sub-workflow are not passed back to the Main workflow.

Internal Envelope

Once a document is submitted to Service Connect (SC), and until it leaves SC, you must convert data to an internally recognizable format before SC can process it. To meet this requirement, incoming documents are placed in the SC internal message envelope. The internal message envelope is an XML document that encapsulates the original document in a single XML node called the business data (dta) node. The other message nodes contain data for routing, error handling, and tracing.

Internal Message

Documents are wrapped in the internal envelope at a workflow entry point: an input channel or a workflow exposed as a web service. Generally, most integration scenarios use input channels because they are less complex.

Input channels are configured in the Epicor Service Connect Administration Console. When you configure the input channel, besides selecting the transfer protocol of the channel (such as Microsoft Message Queuing) and its connection details, select a conversion type appropriate for the document being received. Conversion plug-ins available include standard XML to SC internal message, CSV file to SC internal message, XLS file to SC internal message, and SC external message to SC internal message.

If the document is already in the internal format, you do not need to select a conversion plug-in. Input channel configuration also involves entering values used to populate the internal message metadata. Setting this metadata (such as Sender or Message Type) allows Service Connect to route an incoming document to the appropriate workflow. Values for this metadata are defined in the Message attributes node of the Epicor Service Connect Administration Console, so you can select them during input channel configuration.

Internal Envelope Structure

The following table displays the main nodes in the internal envelope structure of a Service Connect document.

Node	Description
msg	This is the top level node of the internal envelop XML document.
req	This is the request node. It contains the dta , ers , wfl , and ctx nodes.
dta	This is the business data node. It holds the data used for most processing, such as stock items or sales orders. Information that enters Service Connect is stored in this node. The data displays as child nodes. Schemas available for Service Connect workflow elements are located at the following location: http://scshost/schemas/ .

Node	Description
ers	This is the error messages node. If Service Connect, a .NET business object, a web service, or a RESTful service returns an error, the error number and a description display as child nodes.
wfl	This is the workflow data node. It contains the usr node, plus internal information, such as the message type and sender, which is used for document processing.
usr	This is the user node. It contains message extensions and process variables. Each message extension and process variable displays as a child node. Message extensions and process variables are custom data containers you can use to store values in a workflow until the information is ready to use as part of a business process.
ctx	This is the element configuration node. Values in this node are defined by the incoming document or process properties settings.
cfg	This is the second configuration node. It can contain the same configuration data as the ctx node but has lower precedence.
trc	This is the tracing information node. It is used for internal purposes.

External Message Envelope

The Service Connect external message envelope is a schema available to format documents before they enter or after they leave Service Connect. Service Connect does not require external schema conformance, either for incoming or more commonly, outgoing documents. The external schema is a suggestion to supply or receive the Service Connect data that would be exposed by using it. Conversion plug-ins that convert external messages to internal messages and vice versa are available.

The envelope contains service data necessary to process messages by the ScaDES Router service. The envelope format is validated according to <Service Connect Installation folder>\SCS\Schemas\epicor\ScalaMessage.xsd.

Most envelope nodes are optional, as they display only in responses (msg:ers). The only node used for business data is **msg:dta**.

SharePoint Integration

Service Connect supports integration with Windows SharePoint® Services. You can use Service Connect to extend, or as a substitute for, SharePoint workflows.

The Service Connect SharePoint integration component allows Service Connect to:

- Extend Windows SharePoint Services workflows.
- Use a SharePoint document library as an input channel.
- Use an output channel to publish documents to SharePoint document libraries.

Unlike other listeners, the SharePoint input channel does not always consume a document when it is added to the library. Instead, you can leave the document in the library and initiate a workflow when you add or modify a document.

Refer to the Service Connect Installation Guide for system requirements and how to install the Service Connect SharePoint Integration component. After the integration is installed, the process to consume documents from a SharePoint document library and use them in a workflow is as follows:

1. Set up an input channel to monitor the SharePoint document library.

2. Create a Service Connect workflow.
3. Add a message map to direct documents you receive in the input channel and route them to the workflow.
4. Define a SharePoint workflow to use the integration.
5. (Optional) Define an output channel to publish documents from the Service Connect workflow to a SharePoint document library.
6. (If you performed the previous step) Use the output channel with a Poster workflow element in the Service Connect workflow.

For more information, refer to the Service Connect Installation Guide and the Application Help.

Workflow Foundation Integration

Use WFWorkflow Call workflow element to call a Windows Workflow Foundation (WF) workflow as a subroutine of a Service Connect workflow.

Windows WF is a Microsoft technology that defines, executes, and manages workflows. This technology is available starting from .NET Framework 3.0.

Windows Workflow Foundation is the programming model, engine and tools for quickly building workflow enabled applications on Windows. It consists of in-process workflow engine, and designers for Visual Studio 2010. WF includes support for both system workflow and human workflow across a wide range of scenarios including: workflow within line of business applications, user interface page-flow, document-centric workflow, human workflow, composite workflow for service oriented applications, business rule driven workflow and workflow for systems management.

For details about WF, refer to the Microsoft website.

The following is the procedure to use Windows WF with Service Connect:

1. Create a Project in Visual Studio 2010.
2. Add Service Connect activities to the Visual Studio Toolbox.
3. Register the WF workflow in Service Connect.
4. Call the WF workflow from a Service Connect workflow.

For more information, review the Application Help.

Task Monitor

The **Task Monitor** is a web interface to administer workflow tasks. In the Task monitor you can view, track, and process Tasks assigned during workflow execution.

Tasks are workflow elements that can halt a workflow under certain circumstances and then send data to the Task Monitor website, where an authorized user can take action and possibly choose how to route the information. Task workflow elements are typically used in authorization scenarios (for example, establish a credit limit) or error handling (for example, continue processing or halt processing).

Start > All Programs > Epicor Software > Epicor Service Connect > Task Monitor

The following are the main features of the Task Monitor:

- **Quick Filter** - Use the Quick Filter options at the top of the form and click **Apply** to use a filter to narrow the task list.
- **Show XML** - Select a task and click the Show XML button to edit or process the XML message sent to the task.
- **Layout** - Use this option to set up the Task Monitor layout.
- **Process** - To process the message as is, select a task and click **Process**. Otherwise, edit the message data, click **Save**, and click **Process**.
- **List of common exits** - Select an exit (there may be only one) and click **Process**.

Service Connect Services

Use the **Service Manager** utility to manage the services associated with Service Connect.

To start the Service Manager:

1. On the computer where Service Connect is installed, open Windows Explorer and navigate to the folder where Service Connect is installed.

2. Browse to the **ScaServiceManager** folder.

The default menu path is:

C:\Program Files\Epicor Service Connect\Tools\ScaServiceManager

3. Double-click **ScaServiceManager.exe**.

At the end of the Windows taskbar, in the System Tray, a new icon displays.

4. Right-click the Service Manager icon to access its menu options.



Tip You can also access ScaServiceManager using the **Start** menu:



Example Start > All Programs > Epicor Software > Epicor Service Connect > SC Service Manager

In the menu option window, you can:

- Start/Stop Service Connect Services.
- Check the services status.
- Restart Service Connect Windows services.
- Set Service Manager utility options.

Below is a list of Service Connect services you can control using Start > Administrative Tools > Services

- **ScaLogonSrv** - This service manages user credential validation and authentication and provides access to company-specific information and configuration data.
- **ScaLogSrv** - This service enables event logging issued by product components.
- **ScaLicenceSrv** - This service manages license information and provides security functionality.

- **ScaLanSrv** - This service provides multilingual graphical user interface support.
- **ScaLockSrv** - This service allows users to work simultaneously with shared data.
- **ScaUserProfileSrv** - This service provides persistence of user-specific settings and makes users' personal settings available on any client workstation.
- **ScaTrackSrv** - This service tracks business documents.
- **ScaTaskSrv** - This service manages user tasks in business process workflows.
- **ScaDESRouter** - This service provides routing capabilities for message interchange and receives and dispatches XML messages.
- **ScaMessengerSrv** - This service manages communication channels.



Note The default settings for some virus protection programs block port 25 to prevent mass mailing worms from sending mail. If a workflow does not send an email as designed, check your virus protection program's port blocking rules. If this port is blocked on the server where Service Connect is installed, add **ScaMessengerSrv.exe** to the **exception** program list so you can send emails from workflows.

Daily Processing

This section reviews key Service Connect features and contains workshops that explain how to integrate Service Connect with your Epicor solution.

.NET References Import

You can make ESC workflow engine call methods of .NET business objects.

Calling .NET assemblies improves performance because you call Epicor .NET business object directly.

With Service Connect you can call .NET object methods from inside workflows. The following object methods can be used in Service Connect:

- Public methods of public classes that have default constructor
- Public static methods of public classes
- Epicor Business Objects

Only the following Epicor 9.05 assembly types can be imported:

- Epicor.Mfg.BO
- Epicor.Mfg.Proc
- Epicor.Mfg.Rpt

Only the following Epicor 10 assembly types can be imported:

- Erp.Contracts.BO
- Erp.Contracts.Proc
- Erp.Contracts.Rpt
- Ice.Contracts.BO
- Ice.Contracts.Proc
- Ice.Contracts.Rpt

For Generic assembly import, there are no restrictions on class names.

The first step to use Epicor .NET object in Service Connect workflow is to register the .NET assembly in the Epicor Service Connect Administration Console as a .NET Reference. After the import, in the list of .NET references in Epicor Service Connect Administration Console, references to Epicor 9.05 .NET business objects have a simple icon, and references to Epicor 10 .NET business objects have an icon with an E letter on it.

In the import wizard, the following **Assembly types** are available:

- **Generic Assembly** - This type supports import method rules of all public static methods for all public classes. It also supports all public methods for all public classes with default constructor; all input parameters must have default constructors.
- **Epicor 9.05 Assembly** - This type supports import method rules of all public methods for all public classes.

If you select this option, in the following screen, specify the following fields:

- **AppServer** - This is the Epicor application server.
- **Port** - This is the server port, for example, 9001.

- **User** - Enter a valid user, for example, manager.
- **Password** - Enter a valid password for the user, for example, manager.
- **Company** - Enter a company you want to interact with, for example, EPIC06.
- **Plant** - Specify the site within the company, for example, MfgSys.
- **Epicor 10 assembly** - This type supports import method rules of all public methods for all public classes.

If you select this option, in the following screen, specify the following fields:

- **Client config** - The path to the Epicor client configuration file
- **User** - Enter a valid user, for example, manager.
- **Password** - Enter a valid password for the user, for example, manager.
- **Company** - Enter a company you want to interact with, for example, EPIC06.
- **Plant** - Specify the site within the company, for example, MfgSys.

In this course, you will work with Epicor 10 assemblies.

Once you register the .NET Reference, you can call it within Epicor Service Connect Workflow Designer using the **.NET Call** workflow element.

Workshop - Import .NET Reference

Import an Epicor 10 .NET Business Object into the Epicor Service Connect Administration Console as a .NET Reference. After that Service Connect can call the .NET object methods from a workflow.

.NET References are configured links to .NET Business Objects. After you add .NET references in the Administration Console, you can pass and receive datasets to and from .NET Business Objects methods from inside Service Connect workflows.

For any .NET Business Object, you can import all, or just a subset of its methods.



Note If the .NET Business Objects for the outside application are updated, or if you install a newer version of Service Connect, you must re-import the .NET references into the Administration Console.



Important If the **Part** .NET reference is already imported in your environment, skip this workshop. If necessary, contact your system administrator for help.

Log in to the ESC Administration Console

1. Navigate to the **Epicor Service Connect Administration Console** using the following path:
Start > Programs > Epicor Software > Epicor Service Connect > Service Connect Administration Console
The Login window displays.
2. In the **User Name** field, enter **admin** and leave the **Password** field blank.
3. Click **OK**.
The **Epicor Service Connect Administration Console** displays.
If necessary, contact your system administrator for help.

Add a .NET Reference

1. In the **Epicor Service Connect Administration Console**, in the tree view, expand the **Connectivity** node.

2. Right-click the **.NET References** node and select **Add Reference**.
The **Add .NET Reference** window displays.
3. On the welcome screen, click **Next**.

Enter .NET Reference Information

1. In the **Add .NET Reference - .NET reference information** window, in the **Assembly type** field, select **Epicor 10 assembly**.
2. Next to the **Assembly path** field, click the ... (browse) button.
The **Select assembly** window displays.
3. Navigate to the client folder on the server, for example C:\Epicor\ERP10\ERP10.0.300\ClientDeployment\Client.
If necessary, contact your system administrator for help.



Note Browse for an assembly location on the Epicor Service Connect server, not client hard drives.

4. Select the **Erp.Contracts.BO.Part.dll** and click **OK**.
The path is now displayed in the **Assembly path** field.
5. In the **Reference Name** field, the default **Part** name is displayed.
6. Verify the **Import all methods** check box is selected.
7. Click **Next**.
The **Logon to Epicor** window displays.

Enter Epicor Logon Settings

1. In the **Client config** field, specify Epicor client configuration file.
For example, C:\Epicor\ERP10\ERP10.0.300\ClientDeployment\Client\Config\ERP10.sysconfig.
You can alternative click the browse button, navigate to the configuration file, select it and click OK.
2. In the **User** field, enter a valid user, for example, **manager**.
3. In the **Password** field, enter a valid password for the user, for example, **manager**.
4. In the **Company** field, enter **EPIC06**.
5. In the **Plant** field, enter **mfgsys**.
6. Clear the **Use service license** and **Import UD fields** check boxes.
7. Click **Next**.
8. In the **Import .NET reference** window, verify the information and click **Next**.
The .NET assembly import process can take few minutes.

9. Once complete, at the bottom of the **Add .NET Reference** window, the **Reference is successfully imported** message displays.
10. Click **Finish**.

Workflow Basics

A workflow is a series of activities (also referred to as workflow elements) defined in the **Service Connect Workflow Designer**.

Before you create a workflow, define several items such as message attributes and input and output channels in the **Epicor Service Connect Administration Console**. These items receive documents into Service Connect and route them to the appropriate workflow. After you define the administrative pieces, you can create the workflow and add workflow elements to it.

At the end of the following workshop, you will be able to:

- Understand message attributes.
- Set up an input channel that consumes an Excel spreadsheet.
- Route the spreadsheet to a workflow.
- Create a schema for the spreadsheet.
- Convert the spreadsheet data into an XML document that a .NET business object can consume.
- Call a .NET business object method from a workflow to update the target database.
- Verify the progress of the workflow from within the Epicor Service Connect Administration Console.

Workshop - Create a Basic Workflow

In this workshop, create a workflow that updates the Type code for several parts at one time. The workflow accepts an Excel spreadsheet as input.

Create Parts

In the Epicor application, create three simple purchased parts.

Navigate to **Part Maintenance**.

Menu Path: Material Management > Inventory Management > Setup > Part

1. Click **New** and select **New Part**.
2. In the **Part** field, enter **00C1-XXX** (where XXX are your initials) and press **Tab**.
3. In the **Description** field, enter **Purchased Part**.
4. In the **Type** field, verify **Purchased** displays.
5. Accept all other defaults and click **Save**.
6. Repeat steps 1-5 and enter the following two parts:

Field	Data
Part	00C2-XXX (where XXX are your initials)

Field	Data
Description	Purchased Part
Type	Purchased

Field	Data
Part	00C3-XXX (where XXX are your initials)
Description	Purchased Part
Type	Purchased

7. Exit Part Maintenance.
8. Remain in the Epicor application.

Create Folders and Sample Data

1. On your local drive, create the following folders:
C:\ESCSamples\XXX_PartTypeUpdate\IN (where XXX are your initials)
2. In the **XXX_PartTypeUpdate** (where XXX are your initials) folder, create a spreadsheet in Microsoft Excel® with the sample data in the table below.

Part	Description	CurrentType	NewType	Company
00C1-XXX (where XXX are your initials)	Updated type P to M - ESC	P	M	EPIC06
00C2-XXX (where XXX are your initials)	Updated type P to M - ESC	P	M	EPIC06
00C3-XXX (where XXX are your initials)	Updated type P to M - ESC	P	M	EPIC06



Note When you work with data from a spreadsheet or comma separated data, it is important that the first row of information always contains the textual items that represent the nodes in XML. When the file is converted on the input channel, the converter looks at the first row to get the data it needs for the XML nodes. Rows 2-n can contain the data.

3. Save the spreadsheet as **PartTypeUpdate.xls**.



Important Verify the file is saved in the **XXX_PartTypeUpdate** (where XXX are your initials) folder and not in the **IN** folder.

4. Exit Microsoft Excel.

Add a Message Type

Message types indicate the type of information coming into Service Connect.

The information you define in this task is added to documents that enter Service Connect through an input channel.

1. In the **Epicor Service Connect Administration Console**, expand the **Connectivity > Message attributes** nodes.
2. Right-click **Message Types** and select **Add new Message Type**.
The **Add New Message Type** window displays.
3. In the **Message type name** and **Message type description** fields, enter **XXX_Parts** (where XXX are your initials).
4. Click the **Add** button.
The **New Action** window displays.
5. In the **Action name** field, enter **UpdateType**.
6. In the **Action description** field, enter **Update Part Type Code**.
7. In the **New Action** window, click **OK**.
8. In the **Add New Message Type** window, click **OK**.
Notice the new message type is now displayed in the list in the right pane.

Add a Sender

Senders indicate who is sending information to Service Connect. Along with a Message Type, the Sender can be added to a document as it enters Service Connect to determine which workflow should process the information.

1. In the Epicor Service Connect Administration Console, under Connectivity > Message attributes node, right-click **Senders** and select **Add New Sender**.
The **Add New Sender** window displays.
2. In the **Sender name** and **Sender description** fields, enter **XXX_Internal** (where XXX are your initials).
3. Click the **Add** button.
The **New Sub-sender** window displays.
4. In the **Sub-sender name** field, enter **XXX_InvMgr** (where XXX are your initials).
5. In the **Sub-sender description** field, enter **Inventory Manager**.
6. In the **New Sub-sender** window, click **OK**.
7. In the **Add New Sender** window, click **OK**.
Notice, the new sender is now displayed in the list in the right pane.

Add an Input Channel

In this workflow, the input channel is a file folder Service Connect monitors for an Excel spreadsheet. Configure the input channel to add the message attributes created earlier and to convert the .xls file to XML.

1. In the **Epicor Service Connect Administration Console**, navigate to **Connectivity > Communication Setup > your server name > Channels > Input Channels**.
2. Right-click **Input Channels** and select **Add New**.
The **Channel properties** window displays.
3. In the **Channel Name** field, enter **XXX_PartTypeUpdate** (where XXX are your initials).
4. In the **Listener type** field, select **FILE**.
5. Select the **Use scan interval** check box and accept the default value of **1 seconds**.
6. Optionally, select the **High-loaded channel** option to move the channel to a separate free thread as soon as possible to guarantee the maximum performance of the channel.
The system tries to allocate more resources (separate thread) for a high-loaded channel to improve performance and to balance the overall load.
7. Click the **Configure** button.
The **Channel configuration** window displays.
8. Verify the **Message properties** sheet displays and select the following information:

Field	Value
SenderName	XXX_Internal (where XXX are your initials)
SenderSubName	XXX_InvMgr (where XXX are your initials)
MsgType	XXX_ Parts (where XXX are your initials)
Action	UpdateType



Tip Click on each of these fields to see an arrow button to open a window where you can edit or select the value.

9. Navigate to the **Communicator properties** sheet and enter the following information.

Field	Value
File path	C:\ESCSamples\XXX_PartTypeUpdate\IN (where XXX are your initials)
Mask	*.xls
Conversion	excel2xml.dll

10. In the **Channel configuration** and **Channel properties** windows, click **OK**.

The folder you entered in the File path field is now **hot**, meaning that Service Connect consumes any .xls file you add to it. You cannot retrieve a file once it is consumed.


Create the Workflow

1. Navigate to the **Epicor Service Connect Workflow Designer** using the following path:
Start > Programs > Epicor Software > Epicor Service Connect > Workflow Designer

2. Log in as **admin\<no password>**.
If necessary, contact your system administrator for help.
3. From the **File** menu, select **New Process**.
The **New Process** window displays.
4. Verify the **Blank process** option is selected and click **OK**.
5. On the Standard toolbar, click **Save**.
6. To the **Workflow Designer** message, click **Yes**.
The message indicates that the workflow will not function unless you connect any element to the Finish point. Epicor recommends you save frequently when you develop workflows, which means you might see this message until the workflow is complete or nearly complete.
7. In the **Save New Workflow** window, next to the **Package** field, click the **New** button.
The **Create Package** window displays.
8. In the **Package name** field, enter **XXX_SCCourse** (where XXX are your initials) and click **OK**.
The **Save New Workflow** window displays.
9. In the **Save workflow as** field, enter **XXX_PartTypeUpdate** (where XXX are your initials) and click **Save**.

Generate a Schema

The input channel you created earlier converts the spreadsheet to XML. To use the XML inside of the workflow, you must generate a schema for it.

1. In the **Epicor Service Connect Administration Console**, expand the **Connectivity > Schemas** nodes.
2. Right-click the **User Schemas** node and select **Generate Schema**.
The **Generate Schema** window displays.
3. Next to the **Sample data file** field, click the browse button (...), navigate to and select the **PartTypeUpdate.xls** file found in C:\ESCSamples\XXX_PartTypeUpdate (where XXX are your initials).
4. Click **Open**.
5. Next to the **Generated schema** field, click the browse button (...).
The **Save Schema** window displays.
6. In the **File Name** field, verify the **PartTypeUpdate.xsd** schema name defaults and click **OK**.
7. Verify the **Use conversion settings from input channel** option is selected.
 **Tip** Use this option if you plan to use standard conversion defined on the channel, in this case - excel2xml.dll.
8. Click **Next**.
9. From the list of available channels, select the channel **XXX_PartTypeUpdate** (where XXX are your initials).

10. Click **Next**.

11. Select the **Open generated schema in Schema Viewer** check box.

12. Click **Finish**.

The generated schema id displayed.

Remain in the schema window.

You can as well generate schema in Workflow Designer. From the **Tools** menu, select **Generate Schema from Sample Data** utility. Its interface is similar to schema generation from Administration Console.

View the Generated Schema

You can view the schema you generated for Excel input, and any other schema.

1. In the schema window, notice the targetNamespace attribute.
targetNamespace indicates the XML namespace used for all the data nodes in the document.
targetNamespace can help you easily allocate the document source.
2. Close the schema window.
3. You can view any other schema registered in Service Connect. In the Epicor Service Connect Administration Console, expand the **Connectivity > Schemas** nodes.
4. Expand one of the schema folders for example, **ImportedAssemblies > APIInvoiceDLL**.
5. In the right pane, right-click one of the schemas, for example, **Erp_Proxy_BO_APIInvoiceImpl_Close_Request.xsd** and select **Open**.
6. View the schema.
7. Close the schema window.

Add a Conversion to the Workflow

The Conversion workflow element takes the document entering the workflow and maps the data to a .NET Call to update the database.

1. In the **Items** toolbar on the left, click the **Conversion** button.
2. To the right of the **Start** workflow element, click in the workflow design area.
The **Properties** window displays.
3. On the **General** sheet, next to the **Input schema** field, click the **Browse** button.
The **Select a schema** window displays.
4. In the left pane, select **User Schemas**.
5. In the right pane, select the **PartTypeUpdate.xsd** schema and click **Next**.
6. In the **Specify the root element** window, accept the default values and click **Finish**.
7. Next to the **Output schema** field, click the **Browse** button.

The **Select a schema** window displays.

8. In the left pane, select **.NET Reference Schemas**.
9. In the right pane, double-click the **Part** folder.
10. Select **Erp_Proxy_BO_PartImpl_UpdateExt_Request.xsd** and click **Next**.

Make sure that you select request schema for the **UpdateExt** method.



Important These schemas are responsible for .NET business object transactions. For each .NET business object method, there are two schemas: a request schema and a response schema.

11. In the **Specify the root element** window, accept the default values and click **Finish**.

Remain in the **Properties** window.

Define the Conversion

1. Next to the **Conversion** field, click the **Edit** button.
The **New Transformation - Conversion type** window displays.
2. On the left side of the **New Transformation** window, expand the following nodes: msg > req > dta > table > row.
3. On the right, expand the following nodes: msg > req > dta > Erp_Proxy_BO_PartImpl_UpdateExt_Request > ds > UpdateExtPartDataSet > Part.
4. Map the **Company** node on the left to the **CompanyID** node on the right (it displays under Erp_Proxy_BO_PartImpl_UpdateExt_Request).
To create the mapping, click the Company node and hold the left mouse button while you drag the mouse to the CompanyID node on the right. Release the mouse button to create the mapping.
5. Create the following mappings. All of the right nodes display under **Part**.

Left node	Right node
Company	Company
Part	PartNum
Description	PartDescription
NewType	TypeCode
row	Part

6. In the Standard toolbar, click **Save**.
The **Save Transformation File** window displays.
7. In the **Save Transformation File** window, enter **XXX_UpdateType.xslt** (where XXX are your initials) and click **OK**.
8. Exit the XML Mapper.

9. In the **Properties** window, navigate to the **Appearance** sheet.
10. In the **Caption** field, enter **Update Type** and click **OK**.
11. Save your workflow.

Add a .NET Call

The .NET Call workflow element takes the document from the Conversion and uses it to call a .NET business object that updates the target database.

1. In the **Items** toolbar, click the **.NET Call** button.
2. To the right of the **Update Type** conversion, click in the workflow design area.
The **Properties** window displays.
3. On the **General** sheet, next to the **Request ID** field, click **Select**.
The **Request ID** window displays.
4. Expand the **Part > Erp.Proxy.BO.PartImpl** node.
5. Select **PartImpl.UpdateExt** and click **OK**.
6. Navigate to the **Appearance** sheet.
7. In the **Caption** field, enter **Part Update**.
8. In the **Properties** window, click **OK**.

Connect the Workflow Elements

The Connection links workflow elements and shows the order of operations. You must connect all elements.

1. In the **Items** toolbar, click the **Connection** button.
2. Connect **Start** to the **Update Type** conversion.



Tip Click in the center of the Start element, hold the mouse button, drag the cursor to the center of the Update Type conversion, and release the button. Do not try to draw the connection from the edge of Start to the edge of the conversion. Service Connect will place the line automatically with an arrow that points to the conversion element.

3. Make the following connections:
 - **Update Type** to **Part Update**
 - **Part Update** to **Finish**
4. On the Standard toolbar, click **Save**.

Add a Message Map

A message map uses the message attributes added to the document when it enters the input channel and routes it to a workflow for processing.

1. In the **Epicor Service Connect Administration Console**, navigate to **Connectivity > Message Map**.
2. Right-click **Message Map** and select **Add new Request**.
The **New Request ID** window displays.
3. In the **Sender name** field, select **XXX_Internal** (where XXX are your initials).
4. In the **Sender subname** field, select **XXX_InvMgr** (where XXX are your initials).
5. In the **Message type** field, select **XXX_Parts** (where XXX are your initials).
6. In the **Message action** field, select **UpdateType**.
7. Next to the **Request ID** field, click the **Select** button.
8. In the **Request ID** window, clear the **Channels**, **Web Methods** and **.NET Methods** check boxes.
9. In the **RequestID** column, select the row with **XXX_SCCourse\XXX_PartTypeUpdate** (where XXX are your initials) and click **OK**.
10. Click **OK**.

Validate the Workflow

Check the workflow you created for consistency.

1. In the Workflow Designer, from the **File** menu, select **Validate Process**.
2. In the **Validate process** window, view the validation result.
The validation functionality will show a warning message if one of the following conditions is NOT met:
 - The process has one and only Start element.
 - The process has one and only Finish element.
 - There is one and only link from the Start element and no links to it.
 - There is at least one link to the Finish element and no links from it.
 - Each element has at least one link to it and one link from it.
 - Each Sub-Workflow element references an existing process.
 - Each Conversion element contains wfl/SubWF mapping in order to continue workflow execution after Task.
 - Inherited message extensions and process variables are not overwritten by the local ones specified in this workflow.
 - DBOperation element has no empty statements.
3. If you want to validate your workflows every time you save them, select the **Always validate process before saving** check box.

4. Close the **Validate process** window.

Run the Workflow

1. In Windows Explorer, navigate to C:\ESCSamples\XXX_PartTypeUpdate.
2. Right-click **PartTypeUpdate.xls** and select **Copy**.



Important Copy and paste the file into the folder monitored by the Service Connect input channel. If you cut and paste, or if you move the file, you will lose it when Service Connect consumes it in the workflow.

3. Navigate to the C:\ESCSamples\XXX_PartTypeUpdate\IN folder.
4. Right-click and select **Paste**.
The file disappears when it is accepted into Service Connect. It should take about one second.

Verify the Results

Verify the workflow results in the Epicor Service Connect Administration Console and in the Epicor application.

1. In the Epicor Service Connect Administration Console, navigate to Document Tracking > Inbound Messages
2. Check for a message where the Execution point is XXX_PartTypeUpdate (where XXX are your initials) and the timestamp is current.
3. Double-click the status to view the details.
4. In the **Activity Progress** window, double-click an execution point to view the trace details of that activity.



Example Double-click the **Part.Erp.Proxy.BO.PartImpl.UpdateExt** execution point. In the Trace Details window, open the Message Data sheet. Here you can view the XML dataset the .NET business object returns. The dataset should include information for the three parts the workflow updated.

5. Click **OK** until you exit all dialog boxes.

Verify Part Maintenance

Navigate to the Epicor application.

Navigate to **Part Maintenance**.

Menu Path: Material Management > Inventory Management > Setup > Part

1. In the **Part** field, search for and select **00C1-XXX** (where XXX are your initials).
2. In the **Type** field, verify **Manufactured** displays.
3. In the **Description** field, view the new description for the part.
4. Repeat steps 1-3 and verify the two remaining parts **00C2-XXX** and **00C3-XXX** (where XXX are your initials).
5. Exit Part Maintenance.

6. Remain in the Epicor application.

Common Workflow Techniques

The workflow from the previous workshop illustrates the basic settings and techniques required to accept a document into Service Connect, process the document, and use the information to update an application database. However, there are several other workflow elements and techniques commonly used in workflows.

Use the Choice element to route a document along different paths in a workflow based on automatic processing rules, and to test for .NET business object errors. If a .NET business object returns an error, you can route the document to a Task workflow element. The Task pauses the workflow so you can make a choice about how the workflow should proceed. The user which you assign to the task can log in to the web-based Task Monitor to make the decision. Use the Task Monitor to edit the document and possibly resubmit it for a database update.

In addition to using the Choice and Task workflow elements in a workflow, use message extensions and process variables to store information. Message extensions are particularly useful to store the data originally sent into the workflow in the event it must be recovered for further processing. You can use message extensions to store information returned from .NET business object until it is ready to be used.

At the end of this workshop, you will be able to:

- Create and populate a message extension.
- Define a process variable and set its default value.
- Use a Choice workflow element to test for .NET business object errors and route information appropriately.
- Restore the original data from the message extension.
- Add a Task that pauses the workflow so that a flawed document can be fixed and resubmitted.

Workshop - Use Common Workflow Techniques

In this workshop, enhance the workflow created previously to add error checking and document routing capabilities. The new features require you to define a message extension and variable to the workflow and to add a Choice and Task workflow elements.

Create Sample Data

1. Navigate to the C:\ESC\Samples\XXX_PartTypeUpdate folder.
2. In the **XXX_PartTypeUpdate** (where XXX are your initials) folder, create a spreadsheet in Microsoft Excel® with the sample data in the table below.

Part	Description	CurrentType	NewType
	00C1 Common Techs Workshop	M	P
00C2-XXX (where XXX are your initials)	00C2 Common Techs Workshop	M	P
00C3-XXX (where XXX are your initials)	00C3 Common Techs Workshop	M	P

Notice the Company column has been removed. Create a process variable to supply the company ID to the .NET Call. Also notice the spreadsheet is missing key data (the first part ID is blank), which causes the .NET Call update to return an error.

3. Save the spreadsheet as **XXX_PartTypeUpdate_CommonTechs.xls** (where XXX are your initials).
4. Exit Microsoft Excel.

Update the Schema

Since the spreadsheet is changed (Company column is removed), update the schema for it so Epicor Service Connect recognizes the new format.

1. In the **Epicor Service Connect Administration Console**, expand the **Connectivity > Schemas** nodes.
2. Right-click the **User Schemas** node and select **Generate Schema**.
The **Generate Schema** window displays.
3. Next to the **Sample data file** field, click the browse button (...).
4. In the **Open** window, navigate to C:\ESCSamples\XXX_PartTypeUpdate (where XXX are your initials) folder and select the **XXX_PartTypeUpdate_CommonTechs.xls** (where XXX are your initials) sample file.
5. Click **Open**.
6. In the **Generate Schema - General** window, next to the **Generated schema** field, click the browse button (...).
7. In the **Save Schema** window, navigate to the **PartTypeUpdate.xsd** and select it.
8. Click **OK**.
9. To the **Schema with specified name already exists. Do you want to overwrite the existing schema?** message, click **Yes**.
10. Select the **Select and configure conversion plug-in** option.
11. Click **Next**.
12. In the **Generate Schema - Conversion Plug-in** window, select **excel2xml.dll**.
13. Click **Next**.
14. Click **Next**.
15. To the **Validation Result** warning, click **OK**.
16. In the **Generate Schema - Generate Schema** window, verify the **Open generated schema in Schema Viewer** check box is selected.
17. Click **Finish**.
The new schema is generated and displayed.
18. In the schema window review the schema.
Notice, this schema has no **Company** element.
19. Close the schema window.

You can as well update the schema in the Workflow Designer. From the **Tools** menu, select **Generate Schema from Sample Data**. The functionality interface is similar to schema generation in Administration Console.

Add a Message Extension

Save the original data sent into the workflow to be able to restore it after the .NET Call workflow element, so that the data can be fixed and resubmitted for update.

1. In the Workflow Designer, verify the **XXX_SCCourse\XXX_PartTypeUpdate** workflow is open.
2. From the **File** menu, select **Process Properties**.
The **Process properties** window displays.
3. Navigate to the **Message Extensions** sheet.
4. Right-click **msg:usr** and select **Add Container**.
The **Add Container** window displays.
5. In the **Name** field, enter **OriginalData**.
6. Next to the **Schema** field, click the browse button.
The **Select a schema** window displays.
7. In the left pane, click **User Schemas**.
8. Select **PartTypeUpdate.xsd** and click **Next**.
9. Accept the default values and click **Finish**.
10. In the **Add Container** window, click **OK**.
A node labeled **OriginalData** displays beneath **msg:usr**. You can expand the nodes to see the columns from the incoming spreadsheet beneath the row node.
11. Remain in the **Process properties** window.

Add a Process Variable

When you updated the xls file, the Company column was removed from the spreadsheet. Now create a process variable to hold the company code.

1. In the **Process properties** window, navigate to the **Process Variables** sheet.
2. Click the **Add** button.
The **New process variable** window displays.
3. In the **Name** field, enter **CompanyID**.
4. In the **Type** field, verify **String** displays.
5. In the **Default Value** field, enter **EPIC06**.

6. In the **New process variable** window, click **OK**.

7. In the **Process properties** window, click **OK**.

The message extension and process variable are now available in workflow elements, such as Conversions.

The OriginalData message extension displays in the internal envelope at msg/req/wfl/usr/OriginalData.

The process variable displays at msg/req/wfl/usr/ProcessVariables/CompanyID.

Revise the Update Type Conversion

Revise the **Update Type** Conversion in the workflow to map the new document structure to the .NET business object request document. Also, store the incoming data in the OriginalData message extension.

1. Right-click the **Update Type** Conversion and select **Properties**.

The **Properties** window displays.

2. Next to the **Conversion** field, click **Edit**.

The **Conversion type** window displays.

3. In the **Conversion type** window, on the left side, expand the following nodes: req > wfl > usr > ProcessVariables

4. On the right, expand the following nodes: msg > req > dta > Erp_Proxy_BO_PartImpl_UpdateExt_Request > ds > UpdExtPartDataSet > Part.

Also expand wfl > usr.

5. Click the green line between the two **usr** nodes.



Tip You must click either on the right or left side of the mapper window. Clicking in the center does not select the line.

6. Right-click the line and select **Delete selected link(s)**.

7. To the **Do you want to delete selected link?** message, click **Yes**.

8. Map the **dta** node on the left to the **OriginalData** node on the right.

9. To the message, click **No**.

10. Map the **ProcessVariables** node on the left to the **ProcessVariables** node on the right.

11. Map the **CompanyID** node on the left (under ProcessVariables) to the **CompanyID** (under Erp_Proxy_BO_PartImpl_UpdateExt_Request) and **Company** (under UpdExtPartDataSet > Part) nodes on the right.

You are replacing the broken links.

12. To the confirmation message, click **Yes**.

13. In the right pane, select and delete the broken links to the following nodes that display under **Part**.

- PartNum
- PartDescription
- TypeCode

The broken links display in red.

14. Recreate the following mappings.

Left node	Right node
Part	PartNum
Description	PartDescription
NewType	TypeCode
row	Part

15. Click **Save**.

16. Exit the **Conversion type** window.

17. In the **Properties** window, click **OK**.

Add a Choice to Test for Errors

The Choice workflow element analyzes the .NET business object results and routes the document along one of two possible paths based on whether the .NET business object returns an error.

1. In the **XXX_PartTypeUpdate** (where XXX are your initials) workflow, click the connection between the **Part Update** .NET Call and **Finish** and press **Delete**.
2. In the **Items** toolbar, click the **Choice** button.
3. In the workflow, click to the right of the **Part Update** .NET Call.
The **Properties** window displays.
4. In the **Properties** window, click **OK**.
5. In the **Items** toolbar, click the **Conversion** button.
6. In the workflow, click to the right of the **Choice** item.
The **Properties** window displays.
7. In the **Properties** window, click **OK**.
8. In the **Items** toolbar, click the **Task** button.
9. In the workflow, click above and to the right of the **Choice** item.
The **Properties** window displays.
10. In the **Properties** window, click **OK**.
11. In the **Items** toolbar, click the **Connection** button.
12. Make the following connections:
 - **Part Update** to **Choice**

- **Choice** to **Finish**
- **Choice** to **Conversion**
- **Conversion** to **Task**
- **Task** to **Update Type** Conversion
- **Task** to **Finish**

13. Right-click the Choice to Conversion connection and select **Properties**.
The **Properties** window displays.

14. In the **Caption** field, enter **Error**.

15. In the **Properties** window, click **OK**.

16. Right-click the Choice to Finish connection and select **Properties**.
The **Properties** window displays.

17. In the **Caption** field, enter **Success**.

18. In the **Properties** window, click **OK**.

Define the Choice Properties

1. Double-click the **Choice** workflow element.
The **Properties** window displays.

2. Navigate to the **Appearance** sheet.

3. In the **Caption** field, enter **Test for Error**.

4. Navigate to the **Rules** sheet.

5. In the **Default case** field, select **Error**.

Now by default after the Choice element, the workflow is directed to the Conversion element that restores the original data.

6. Clear the **Allow multiple exits** check box.

7. In the grid, select the **Success** row.

8. Click the **Configure** button.

The **Rule Assistant** window displays.

9. Expand the following nodes: msg > req > dta > Erp_Proxy_BO_PartImpl_UpdateExt_Response.

10. Map the **errorsOccurred** node to the **XPath** marker on the right.

11. Navigate to the **XPath expression** sheet.

12. Edit the expression so it displays as follows:

/msg:msg/msg:req/msg:dta/dta:Erp_Proxy_BO_PartImpl_UpdateExt_Response[dta:errorsOccurred = "false"]

This is the simplified XPath expression form; namespaces are omitted and only the xml elements names are displayed.

You described the case when no errors occurred when the .NET Call processed the converted incoming message.

13. Verify the **Admit Value** field displays **true**.

If the condition you described (no errors occurred during the .NET Call work) is true, the workflow is directed to the Finish element.

14. In the **Rule Assistant** window, click **OK**.

15. In the **Properties** window, click **OK**.

Configure the Restore Original Data Conversion

To make the data available to edit in the Task workflow element, restore the original data to the dta node of the internal envelope.

1. Double-click the **Conversion** that follows the Choice workflow element.
The **Properties** window displays.
2. Navigate to the **Appearance** sheet.
3. In the **Caption** field, enter **Restore Original Data**.
4. Click **Apply**.
5. Navigate to the **General** sheet.
6. In the **Input schema** field, verify the **Erp_Proxy_BO_PartImpl_UpdateExt_Response.xsd** schema is displayed.
The schema is automatically inherited from the previous workflow element. Inherited schemas are displayed in gray text.
7. Next to the **Output schema** field, click **Browse**.
The **Select a schema** window displays.
8. In the left pane, click **User Schemas**.
9. Select the **PartTypeUpdate.xsd** schema and click **Next**.
10. Accept the default values and click **Finish**.
11. Remain in the **Properties** window.

Define the Conversion

1. Next to the **Conversion** field, click **Edit**.
The **New Transformation - Conversion type** window displays.

2. On the left side, verify the following nodes are expanded: msg > req > wfl > usr
3. On the right, verify the following nodes are expanded: msg > req
4. Map the **OriginalData** node on the left to the **dta** node on the right.
5. To the message, click **No**.
6. On the Standard toolbar, click **Save**.
The **Save Transformation File** window displays.
7. As the transformation file name, enter **XXX_RestoreOrigData.xslt** (where XXX are your initials) and click **OK**.
8. Exit the **XXX_RestoreOrigData.xslt - Conversion type** (where XXX are your initials) window.
9. In the **Properties** window, click **OK**.

Configure the Resolve Errors Task

The Task pauses the workflow so you can decide how to route a document. This Task also allows you to edit and resubmit the document to the .NET business object.

1. Right-click the connection between the **Task** and **Update Type** and select **Properties**.
The **Properties** window displays.
2. In the **Caption** field, enter **Retry** and click **OK**.
3. Right-click the connection between the **Task** and **Finish** and select **Properties**.
The **Properties** window displays.
4. In the **Caption** field, enter **Cancel** and click **OK**.
5. Double-click the **Task**.
The **Properties** window displays.
6. Verify the **Schema** sheet displays.
7. Next to the **XSD Schema** field, click the **Browse** button.
The **Open Schema** window displays.
8. In the left pane, click **User Schemas**.
9. Select the **PartTypeUpdate.xsd** schema and click **OK**.
10. Navigate to the **General** sheet.
11. In the **Task name** field, enter **Resolve Errors**.
12. Navigate to the **Users** sheet.
13. Click the **Add** button.
The **Add User/Group** window displays.

14. Expand the **ADMINS** node.
15. Select **Admin** and click **OK**.
16. Navigate to the **Appearance** sheet.
17. In the **Caption** field, enter **Resolve Errors** and click **OK**.
18. On the Standard toolbar, click **Save**.

Run the Workflow

1. In Windows Explorer, navigate to C:\ESCSamples\XXX_PartTypeUpdate (where XXX are your initials) folder.
2. Right-click **XXX_PartTypeUpdate_CommonTechs.xls** (where XXX are your initials) and select **Copy**.



Important Copy and paste the file into the folder monitored by the Service Connect input channel. If you cut and paste, or if you move the file, you will lose it when Service Connect consumes it in the workflow.

3. Navigate to the C:\ESCSamples\XXX_PartTypeUpdate\IN folder.
4. Right-click and select **Paste**.
The file disappears when Epicor Service Connect accepts it. It should take about one second.

Check the Workflow Progress

Verify the workflow progress in the Epicor Service Connect Administration Console.

1. In the Epicor Service Connect Administration Console, navigate to Document Tracking > Inbound Messages.
2. Check for a message where the Execution point is **XXX_PartTypeUpdate** (where XXX are your initials), the Status is **Pending**, and the timestamp is current.
3. Double-click the message.
The **Activity Progress** window displays.
You can see that the workflow progress stopped at the **Resolve Errors** Task workflow element.
4. Double-click **Part.Erp.Proxy.BO.PartImpl.UpdateExt** to view the execution details.
The **Trace Details** window displays.
5. Verify the **Message Data** sheet displays.
6. Locate the **msg > req > dta > Erp_Proxy_BO_PartImpl_UpdateExt_Response > errorsOccurred** node.
Its value is true.
7. Under the **Erp_Proxy_BO_PartImpl_UpdateExt_Response** node, expand the **result > BOUpdErrorDataSet > BOUpdError** node.
The **ErrorLevel** node displays **Error**, and the **ErrorText** node displays **Part is required**.

8. In the **Trace Details** window, click **OK**.
9. In the **Activity Progress** window, click **OK**.

Use the Task Monitor

Log into the Task Monitor to review the document, supply the missing part ID, and retry the update.

Review the **Environment Setup - Task Monitor** section for the Task Monitor setup requirements.

1. Use the following path to navigate to the Task Monitor: Start > Programs > Epicor Software > Epicor Service Connect > Task Monitor
2. Log in as **admin\<no password>**.
This is the user to whom you assigned the task.
3. Select the **Resolve Errors** task and click **Show Xml Editor**.
The **Xml Editor** window displays.
4. Click the **table** link.
5. Click the first **row** link.
View the **Value** field for the **Part** row is blank.
6. In the **Part** row, click the **Edit** icon.
7. In the **Value** field, enter **00C1-XXX** (where XXX are your initials).
8. In the **Part** row, click the **Save** icon.
9. In the **Xml Editor** window, click **Save**.
10. In the **Xml Editor** window, click **Process**.
The **Process** window displays.
11. From the **List of common exits**, select **Retry** and click **Process**.
12. Exit the web browser.

Verify the Results

1. In the Epicor Service Connect Administration Console, navigate to Document Tracking > Inbound Messages to verify the continued progress of the workflow.
You might have to wait for some time before the workflow completes.
2. On the Standard toolbar, click the **Refresh** button.
The status should now be **Complete**. If you view the processing details, you can verify the document was sent from the Task to the Update Type conversion and resubmitted to the .NET business object, which succeeded on the second attempt.
3. Minimize the Epicor Service Connect Administration Console.

4. Navigate to **Part Maintenance**.

Menu Path: Material Management > Inventory Management > Setup > Part

5. In the **Part** field, search for and select **00C1-XXX** (where XXX are your initials).

6. Verify the **Type** and **Description** fields updated appropriately.

7. Repeat steps 5-6 to verify the two remaining parts **00C2-XXX** and **00C3-XXX** (where XXX are your initials).

8. Exit Part Maintenance.

Document Publication from a Workflow

Use Epicor Service Connect to send documents out of a workflow. Workflows can post files to message queues, file folders, and FTP sites; and also send e-mail through an SMTP server.

Two components are required to publish information from a workflow:

- Output channels define the communication method and can supply any required credentials.
- Posters are workflow elements you can place into a workflow diagram to send a document out of the workflow using one or more output channels.

The workshop in this section explains how to send an e-mail from the previously created workflow. The e-mail alerts the users about the task waiting for them in the Task Monitor.

At the end of the following workshop, you will be able to:

- Define an output channel that uses an SMTP server.
- Add a Poster workflow element to the workflow that uses the output channel.
- Develop an e-mail template that includes information from the workflow.

Workshop - Send an E-Mail from a Workflow

To enhance the previously created workflow, add an e-mail notification on a Task creation.

Create an Output Channel

The output channel in this workshop uses an SMTP server.

1. In the Epicor Service Connect Administration Console, navigate to Connectivity > Communication Setup > <your server name> > Channels > Output Channels
2. Right-click **Output Channels** and select **Add New**.
The **Channel properties** window displays.
3. In the **Channel name** field, enter **XXX_NETErrorNotice** (where XXX are your initials).
4. In the **Speaker type** field, select **SMTP**.
5. Click the **Configure** button.
The **Channel configuration** window displays.

6. Navigate to the **Communicator properties** sheet.
7. In the **URL** field, enter **your smtp server**.
8. In the **Channel configuration** window, click **OK**.
9. In the **Channel Properties** window, click **OK**.

Add a Poster to the Workflow

The Poster workflow element uses the output channel to send an e-mail notification about the part update failure.

1. Navigate to the Workflow Designer and verify the **XXX_SCCourse\XXX_PartTypeUpdate** (where XXX are your initials) workflow is open.
2. In the **Items** toolbar, click the **Poster** button.
3. In the workflow design area, click between the **Test for Error** Choice and the **Restore Original Data** Conversion.
The **Properties** window displays.
4. In the **Properties** window, click **OK**.
5. Place the **Poster** on the **Error** connection.
6. Select the **Error** connection.
7. Drag the handle on the arrow tip from the **Restore Original Data** Conversion to the **Poster** to connect the **Test for Error** Choice to the **Poster**.
You might need to adjust the caption of the Error connection.
8. In the **Items** toolbar, click the **Connection** button.
9. Connect the **Poster** to the **Restore Original Data** Conversion.
10. On the Standard toolbar, click **Save**.

Set the Poster Properties

In this part of the workshop, select the schema the Poster uses to supply information into the e-mail message and add the output channel created earlier.

1. Right-click the **Poster** and select **Properties**.
The **Properties** window displays.
2. Navigate to the **Appearance** sheet.
3. In the **Caption** field, enter **Send E-mail**.
4. Navigate to the **Schema** sheet.
5. Notice the **XSD Schema** field displays the schema inherited from Choice.

Verify it is the

http://scsghost/schemas/ImportedAssemblies/Part/Erp_Proxy_BO_PartImpl_UpdateExt_Response.xsd
schema.

6. In the **Properties** window, navigate to the **Fixed Channels** sheet.
7. Click the **Add** button.
The **Select Channels** window displays.
8. Select **XXX_NETErrorNotice** (where XXX are your initials) and click **OK**.
9. In the **Fixed Channels** sheet, select the row with the **XXX_NETErrorNotice** (where XXX are your initials) channel.
10. Click the **Configure** button.
The **Output Channel Configuration** window displays.
11. Verify the **Message Properties** sheet displays.
12. In the **From** field, enter **scnotify@epicor.com**.
13. In the **To** field, enter **your email address**.
14. Remain in the **Output Channel Configuration** window.

Create an E-Mail Template

The e-mail template contains text that you enter, a link to the Task Monitor, and details about the .NET Business Object error.

1. In the **Email template name** field, click the far right down arrow button to open the e-mail editor.
The **Email Template** window displays.

2. From the toolbar, click down arrow next to the **Mode** icon and verify **HTML** mode is selected.



Tip You can create email templates in the following editing modes: Plain Text, HTML, XSLT to Plain Text and XSLT to HTML. Use the toolbar to select formatting options, such as font, size and color.

3. In the **Subject** field, enter **SC Task Waiting**.
4. In the body, enter the following text:
A .NET Business Object request failed. Please log into the Task Manager to review the document.
<http://scsghost/TaskMonitor/>
Error details:
5. Press **Enter** to skip to the new line.
6. Right-click in the new line and select **Insert simple link**.
The **Insert simple link** window displays.
7. In the **Link name** field, enter **ErrorDesc**.

8. In the link editor, expand the following nodes: **msg > req > dta > Erp_Proxy_BO_PartImpl_UpdateExt_Response > result > BOUpdErrorDataSet/BOUpdError**.
9. Map the **ErrorText** node to the **XPath** marker on the right.
10. In the **Separator** field, select **CRLF**.
11. In the **Insert simple link** window, click **OK**.
12. On the Email Template Toolbar, click **Save**.
The **Select Email Template** window displays.
13. In the name field, enter **XXX_NETErrorNotice** (where XXX are your initials) and click **OK**.
14. In the **Email Template** window, click **OK**.
15. In the **Output Channel Configuration** window, click **OK**.
16. In the **Properties** window, click **OK**.



Tip As you modified the output channel configuration previously set up in the Administration Console, note the customized poster icon displays instead of the default one. Customized icons indicate the default information was overridden in elements such as Poster, Requester, Web Method or .Net Call.

17. On the Standard toolbar, click **Save**.

Run the Workflow and Verify the E-Mail

Use the same document as in the previous workshop to test the e-mail notification.

1. In Windows Explorer, navigate to the **C:\ESCSamples\XXX_PartTypeUpdate** (where XXX are your initials) folder.
2. Right-click **XXX_PartTypeUpdate_CommonTechs.xls** (where XXX are your initials) and select **Copy**.
3. Navigate to the **C:\ESCSamples\XXX_PartTypeUpdate\IN** folder.
4. Right-click and select **Paste**.
5. Open your e-mail client and verify you received the e-mail with the error details.
It may take a while for the e-mail to arrive.
6. In the ESC Administration Console, navigate to Document Tracking > Inbound Messages.
7. Right-click the Pending process status and select **Delete Whole Activity**.
8. To the confirmation message, click **Yes**.

Functoids

Functoids are a collection of XPath and extension functions you can add to the XML Mapper center pane when you create a conversion to perform a variety of in-transformation tasks, such as comparisons, mathematical operations, data type conversions, and so on.



Example The Sum Functoid can total the values of a document that contains several rows of numeric data.

Use the Functoid Palette to add Functoids to a conversion. Some Functoids require one or more nodes to be mapped to them from the incoming document. You can map the result of a Functoid to another Functoid or to a node in the target document. More than fifty Functoids are available. To learn more about them, review the Epicor Service Connect application help or the Epicor Service Connect User Guide.

At the end of the following workshop, you will be able to:

- Use a Value Conversion Functoid to transform information in the incoming document into values that the target database accepts.

Workshop - Use a Value Conversion Functoid

In this workshop, add a Value Conversion Functoid to the Update Type conversion in the PartTypeUpdate workflow.

Create Sample Data

1. Navigate to the C:\ESC\Samples\XXX_PartTypeUpdate (where XXX are your initials) folder.
2. In the **XXX_PartTypeUpdate** (where XXX are your initials) folder, create a spreadsheet in Microsoft Excel® with the sample data in the table below.

Part	Description	CurrentType	NewType
00C1-XXX (where XXX are your initials)	00C1 Functoids	Purchased	Manufactured
00C2-XXX (where XXX are your initials)	00C2 Functoids	Purchased	Manufactured
00C3-XXX (where XXX are your initials)	00C3 Functoids	Purchased	Manufactured

Notice the CurrentType and NewType values have been changed from **P** and **M** to **Purchased** and **Manufactured**. These values must be converted back to the Type codes the target database expects.

3. Save the spreadsheet as **XXX_PartTypeUpdate_Functoids.xls** (where XXX are your initials).
4. Exit Microsoft Excel.

Add the Functoid to the Conversion

1. In the Workflow Designer, verify the **XXX_SCCourse\XXX_PartTypeUpdate** (where XXX are your initials) workflow is open.

2. Double-click the **Update Type** Conversion.
The **Properties** window displays.
3. Verify the **General** sheet displays.
4. Next to the **Conversion** field, click the **Edit** button.
The **Conversion type** window displays.
5. On the left side, expand the following nodes: msg > req > dta > table > row
6. On the right side, expand the following nodes: msg > req > dta > Erp_Proxy_BO_PartImpl_UpdateExt_Request > ds > UpdExtPartDataSet > Part
7. Delete the mapping between the **NewType** node and the **TypeCode** node.
8. To the confirmation message, click **Yes**.
9. On the Standard toolbar, click the **Functoid Palette** button.
The **Functoid palette** window displays.
10. In the **Functoid palette** window, navigate to the **Special** sheet.
11. Click the **Value Conversion** icon (second from the left), drag the cursor to the center pane of the window, and release the mouse button.
12. From the left pane, map the **NewType** node to the **source = <empty>** on the **Functoid**.
13. Double-click the **Functoid**.
The **Value Conversion Functoid** window displays.
14. Click the **Add** button.
15. In the **Source Value** column, enter **Manufactured**.
16. In the **Target Value** column, enter **M**.
17. Click the **Add** button to add the second conversion.
18. In the **Source Value** column, enter **Purchased**.
19. In the **Target Value** column, enter **P**.
20. In the **Value Conversion Functoid** window, click **OK**.
21. In the **Functoid** header, click the words **value conversion** and drag the cursor to the **TypeCode** node in the right pane.
This maps the Functoid output to the target document.
22. Close the Functoid palette window.
23. On the Standard toolbar, click **Save**.
24. Exit the **Conversion type** window.

25. In the **Properties** window, click **OK**.
26. On the Standard toolbar, click **Save**.

Run the Workflow and Verify the Data

1. In Windows Explorer, navigate to the C:\ESC\Samples\XXX_PartTypeUpdate (where XXX are your initials) folder.
2. Right-click **XXX_PartTypeUpdate_Functoids.xls** (where XXX are your initials) and select **Copy**.
3. Navigate to the C:\ESC\Samples\XXX_PartTypeUpdate\IN folder.
4. Right-click and select **Paste**.
5. In the Epicor Service Connect Administration Console, navigate to Document Tracking > Inbound Messages.
6. Check for a message where the Execution point is XXX_PartTypeUpdate (where XXX are your initials), and the timestamp is current.
7. Double-click the message to access the **Activity Progress** window.
8. Double-click the Start workflow element.
In the **Trace Details** window, look at the message data details for the Start workflow element. Notice that the type codes submitted to the workflow are spelled out.
9. Click the down arrow button to move to the trace details for the next workflow element.
The trace details for the **Update Type** Conversion are displayed. Notice that, when the part update request is passed to the .NET Call, the values have been converted to the expected type codes.
10. In the **Trace Details** window, click **OK**.
11. In the **Activity Progress** window, click **OK**.

Sub-Workflow Calls

Use Sub-workflow Call workflow element to call a workflow as a subroutine for another workflow.

You can set the Sub-workflow to run asynchronously (the main workflow continues to execute) or synchronously (the main workflow pauses until the Sub-workflow finishes). When a Sub-workflow is set to execute synchronously, the Sub-workflow results are available in the following workflow element of the main workflow.

Also, you can set the Sub-workflow to execute once or to cycle through specific nodes in a document. For example, if a document contains a sales order, you can set up a Sub-workflow to cycle through each sales order line item. If necessary, you can send the data stored in message extensions for use in the Sub-workflow.

At the end of the following workshop, you will be able to:

- Create a copy of an existing workflow.
- Set up a Sub-workflow workflow element that calls another Service Connect workflow as a subroutine.
- Define the cycle settings so the incoming document is processed one line at a time.
- Build a Sub-workflow schema based on the cycle settings.

- Apply the Sub-workflow schema in the workflow used as a subroutine.

Workshop - Use a Sub-Workflow Call

In this workshop, create a master workflow that calls a copy of the previously created workflow as a subroutine. Instead of attempting to update three part records at once, the PartTypeUpdate workflow updates a single record at a time. Thus, if the incoming document has one bad row of data, the two good rows are updated and only the bad row fails.

Create Sample Data

1. Navigate to the C:\ESC\Samples\XXX_PartTypeUpdate (where XXX are your initials) folder.
2. In the **XXX_PartTypeUpdate** (where XXX are your initials) folder, create a spreadsheet in Microsoft Excel® with the sample data in the table below.

Part	Description	CurrentType	NewType
00C1-XXX (where XXX are your initials)	00C1 Subworkflow	Manufactured	Purchased
	00C2 Subworkflow	Manufactured	Purchased
00C3-XXX (where XXX are your initials)	00C3 Subworkflow	Manufactured	Purchased

This is basically the same data used for the Functoids workshop, except the NewType is switched back to Purchased and the part ID is missing for the second row.

3. Save the spreadsheet as **XXX_PartTypeUpdate_Subworkflow.xls** (where XXX are your initials).
4. Exit Microsoft Excel.

Make a Copy of the Original Workflow

1. In the Workflow Designer, verify the **XXX_SCCourse\XXX_PartTypeUpdate** (where XXX are your initials) workflow is open.
2. From the **File** menu, select **Save Process As**.
The **Save Workflow As** window displays.
3. In the **Save workflow as** field, enter **XXX_PartTypeUpdateAsSub** (where XXX are your initials) and click **Save**.

Update the Namespaces

The general rule is that if you use a Sub-Workflow that includes .NET references or Service references, you must update namespaces in the following entities used in this Sub-Workflow:

- input channel configuration
- incoming document schema

Both must use the same namespace the .NET method or Service method uses. It is important to match the namespace entries to the .NET or Service schema definition.



Important Update the namespaces before you create the Sub-Workflow schema.

In the workflows used in this workshop, find the namespace the Part Update .NET Call uses, then assign this namespace to the XXX_PartTypeUpdate (where XXX are your initials) channel and to the PartTypeUpdate.xsd schema.

Find out the .NET method namespace

1. In the **Epicor Service Connect Administration Console** tree, expand the **Connectivity > Schemas > ImportedAssemblies** nodes.
2. Click the **Part** node.
3. Double-click **Erp_Proxy_BO_PartImpl_UpdateExt_Request.xsd**.
4. On the **Erp_Proxy_BO_PartImpl_UpdateExt_Request** tab, locate the **targetNamespace** attribute.
The attribute value is **http://Epicor.com/Part/Erp_Proxy_BO_PartImpl_UpdateExt_Request**.
5. Copy the targetNamespace attribute value to a text editor, for example, to Notepad.
Now update the namespace for the input channel and the incoming message schema.
6. Close the schema window.

Update the default namespace in the input channel configuration

1. In the Epicor Service Connect Administration Console, navigate to **Connectivity > Communication Setup > your server name > Channels > Input Channels**.
2. Double-click the **XXX_PartTypeUpdate** (where XXX are your initials).
3. In the **Channel properties** window, click the **Configure** button.
4. In the **Channel Configuration** window, open the **Communicator properties** sheet.
5. In the **ConversionCfg** field, in the **Value** column, click the arrow button.
6. In the **Configuration** window, open the **Text** tab and click the **Default** button.
7. Add the following line to the configuration. It is important that it is entered exactly as shown.
<DefaultNamespace>http://Epicor.com/Part/Erp_Proxy_BO_PartImpl_UpdateExt_Request</DefaultNamespace>.
DefaultNamespace indicates the XML namespace used for all the data nodes in the document.
8. Click **OK**.
9. To the validation message, click **OK**.
10. In the **Channel Configuration** window, click **OK**.
11. In the **Channel properties** window, click **OK**.

Update the namespace in the incoming document schema

1. In the **Epicor Service Connect Administration Console**, expand the **Connectivity > Schemas** nodes.
2. Right-click the **User Schemas** node and select **Generate Schema**.
The **Generate Schema** window displays.
3. Next to the **Sample data file** field, click the browse button (...).
4. In the **Open** window, navigate to **C:\ESCSamples\XXX_PartTypeUpdate** (where XXX are your initials) folder and select the **XXX_PartTypeUpdate_Subworkflow.xls** (where XXX are your initials) sample file.
5. Click **Open**.
6. In the **Generate Schema - General** window, next to the **Generated schema** field, click the browse button (...).
7. In the **Save Schema** window, navigate to the **PartTypeUpdate.xsd** and select it.
8. Click **OK**.
9. To the **Schema with specified name already exists. Do you want to overwrite the existing schema?** message, click **Yes**.
10. Select the **Select and configure conversion plug-in option**.
11. Click **Next**.
12. In the **Generate Schema - Conversion Plug-in** window, select **excel2xml.dll**.
13. Click **Next**.
14. In the **Conversion Plug-in Configuration** window, open the **Text** tab and click the **Default** button.
15. Add the following line to the configuration. It is important that it is entered exactly as shown.
`<DefaultNamespace>http://Epicor.com/Part/Erp_Proxy_BO_PartImpl_UpdateExt_Request</DefaultNamespace>`.
16. Click **Next**.
17. To the validation message, click **OK**.
18. In the **Generate Schema** window, click **Finish**.
19. In the schema window, notice the **targetNamespace** parameter is now set to **http://Epicor.com/Part/Erp_Proxy_BO_PartImpl_UpdateExt_Request**.
20. Close the schema window.

Create the Master Workflow

The master workflow contains a Sub-Workflow call to the XXX_PartTypeUpdateAsSub workflow.

1. From the **File** menu, select **New Process**.

2. In the **New Process** window, click **OK**.
3. On the Standard toolbar, click **Save**.
4. To the **Workflow Designer** message, click **Yes**.
The **Save New Workflow** window displays.
5. In the **Package** field, select the **XXX_SCCourse** (where XXX are your initials) package.
6. In the **Save workflow as** field, enter **XXX_PartTypeUpdateMaster** (where XXX are your initials).
7. In the **Save New Workflow** window, click **Save**.
8. To the confirmation message, click **Yes**.

Add the Sub-Workflow to the Master Workflow

Take the following steps to add the Sub-Workflow workflow element:

- Set the schema for the incoming document (PartTypeUpdate.xsd).
 - Select which workflow to call as a subroutine (XXX_PartTypeUpdateAsSub).
 - Define the synchronization settings.
 - Select the data node in the schema to use for cycling through the data rows.
 - Create a schema based on the cycle settings.
1. In the **Items** toolbar, click the **Workflow call** button.
 2. In the workflow, click between **Start** and **Finish**.
The **Properties** window displays.
 3. Verify the **Schema** sheet displays.
 4. Click the **Browse** button.
The **Open Schema** window displays.
 5. Click **User Schemas**.
 6. Select **PartTypeUpdate.xsd** and click **OK**.
You can alternatively add this schema on the Start element. In this case, this same schema will be automatically inherited by the Sub-Workflow.
 7. Navigate to the **General** sheet.
 8. Click the **Select** button.
The **Select workflow** window displays.
 9. Expand the **XXX_SCCourse** (where XXX are your initials) node.
 10. Select the **XXX_PartTypeUpdateAsSub** (where XXX are your initials) workflow and click **Open**.
 11. Clear the **Pass msg:usr section** check box.

Use this setting when you need to pass message extensions and process variables from the main workflow to the sub-workflow.

12. Navigate to the **Cycling** sheet.
13. Select the **Enable cycling execution** check box.
14. Click the **Build XPath** button.
The **Rule Assistant** window displays.
15. Expand msg > req > dta > table > row
16. Map the **row** node to the **XPath** marker on the right.
17. In the **Rule Assistant** window, click **OK**.
18. Remain in the **Properties** window.

Create a Sub-Workflow Schema

1. In the **Properties** window, click the **Create sub-workflow schema** button.
The **Save Schema** window displays.
2. In the **File name** field, enter **PartTypeUpdate.SubWF.xsd** and click **OK**.
3. Navigate to the **Appearance** sheet.
4. In the **Caption** field, enter **PartTypeUpdateAsSub**.
5. In the **Properties** window, click **OK**.
6. Use the **Connection** tool to connect **Start** to the **Sub-Workflow** call and to connect the **Sub-Workflow** call to **Finish**.
7. On the Standard toolbar, click **Save**.

Convert the Message Extension to Use the Sub-Workflow Schema

The sub-workflow schema contains only one row of data. Update OriginalData message extension to match the new structure.

1. In the Workflow Designer, from the **Window** menu, select **XXX_PartTypeUpdateAsSub** (where XXX are your initials) workflow.
2. From the **File** menu, select **Process Properties**.
The **Process properties** window displays.
3. Navigate to the **Message Extensions** sheet.
4. Expand the **msg:usr** node.
5. Right-click **OriginalData** and select **Edit Container**.

The **Container Properties** window displays.

6. Next to the **Schema** field, click the browse button.

The **Select a schema** window displays.

7. In the left pane, click **User Schemas**.

8. Select the **PartTypeUpdate.SubWF.xsd** schema and click **Next**.

9. Accept the default values and click **Finish**.

10. In the **Container Properties** window, click **OK**.

The message extension now conforms to the schema developed for the sub-workflow, which represents the format of the data sent from the sub-workflow into the master workflow.

You can expand the nodes to review the new, more compact structure.

11. In the **Process properties** window, click **OK**.

Revise the Update Type Conversion

Modify the Update Type Conversion to use the sub-workflow schema as the incoming document schema.

1. In the Workflow Designer, double-click the **Update Type** Conversion.

The **Properties** window displays.

2. Verify the **General** sheet displays.

3. Next to the **Input schema** field, click **Browse**.

The **Select a schema** window displays.

4. In the left pane, click **User Schemas**.

5. In the right pane, select the **PartTypeUpdate.SubWF.xsd** schema and click **Next**.

6. Accept the default values and click **Finish**.

7. Remain in the **Properties** window.

Rebuild the Update Type Conversion XSLT

The conversion now accepts a different incoming document. So you must modify the Update Type Conversion mapping.

1. Next to the **Conversion** field, click **Edit**.

The **Conversion type** window displays.

2. Press **Ctrl +A** to select all the links.

3. Press **Delete**.

When you have updated the same conversion mapping several times, it is safer to remove all the links and start from scratch.

4. To the XML Mapper message click **Yes**.
5. To the message that states system links were not removed, click **OK**.
6. In the left pane, expand the following nodes:
 - msg > req > dta > row
 - wfl > usr > Process Variables
7. In the right pane, expand the following nodes: msg > req > dta > Erp_Proxy_BO_PartImpl_UpdateExt_Request > ds > UpdateExtPartDataSet > Part
8. Recreate the following mappings:

Left node	Right node
...dta > row > Part	...Part > PartNum
...dta > row > Description	...Part > PartDescription
...dta > row > NewType	Map to Value Conversion functoid argument. If no functoid arguments are displayed, right-click the functoid and select Expand Functoid to be able to map the NewType node to the part that says source = <empty> . Map the Functoid header to the ...Part > TypeCode node in the right pane.
... wfl > usr > ProcessVariables > CompanyID	...Part > Company; ...Erp_Proxy_BO_PartImpl_UpdateExt_Request > CompanyID
dta	...wfl > usr > OriginalData To the message, click No .
ProcessVariables	ProcessVariables

9. On the Standard toolbar, click **Save**.
10. Exit the **Conversion type** window.
11. In the **Properties** window, click **OK**.
12. Save your workflow.

Revise the Restore Original Data Conversion

Update the output schema for the Restore Original Data Conversion to the sub-workflow schema.

1. Double-click the **Restore Original Data** Conversion.
The **Properties** window displays.
2. Next to the **Output schema** field, click **Browse**.
The **Select a schema** window displays.

3. Verify the **User Schemas** option is selected in the left pane.
4. Select **PartTypeUpdate.SubWF.xsd** and click **Next**.
5. Accept the default settings and click **Finish**.
6. In the **Properties** window, click **OK**.
The conversion XSLT does not need to be revised.
7. Save your workflow.

Update the Message Map

Update the message map that routes the documents from the input channel to the workflow to call the master workflow instead of the sub-workflow.

1. In the Epicor Service Connect Administration Console, navigate to Connectivity > Message Map.
2. Right-click the message map with the following attributes and select **Properties**.
 - Sender Name = XXX_Internal (where XXX are your initials)
 - Sender Subname = XXX_InvMgr (where XXX are your initials)
 - Message Type = XXX_Parts (where XXX are your initials)
 - Message Action = UpdateTypeThe **Map Properties** window displays.
3. Click the **Select** button.
The **RequestID** window displays.
4. Clear the **Channels**, **Web Methods** and **.NET Methods** check boxes.
5. Expand the **RequestID** column.
6. Select the **XXX_SCCourse\XXX_PartTypeUpdateMaster** (where XXX are your initials) row and click **OK**.
Verify **Request ID is valid** displays beneath the **Request ID** field.
7. In the **Map Properties** window, click **OK**.

Run the Workflow and Verify the Results

1. In Windows Explorer, navigate to the C:\ESC\Samples\XXX_PartTypeUpdate (where XXX are your initials) folder.
2. Right-click **XXX_PartTypeUpdate_SubWorkflow.xls** (where XXX are your initials) and select **Copy**.
3. Navigate to the C:\ESC\Samples\XXX_PartTypeUpdate\IN folder.
4. Right-click and select **Paste**.
The following should occur:
 - You receive an e-mail titled SC Task Waiting to announce the failure of the second row of data.

- In the activity progress details (Epicor Service Connect Administration Console > Document Tracking > Inbound Messages), you should see the PartTypeUpdateAsSub workflow called three times. The second time it is called, it will show that the .NET object returned an error and that a Task was created.
- A Task is created for the Admin user in the Task Monitor. The Task contains only the part with the missing ID. You can add the missing data or opt to cancel the operation and then refresh the activity progress details to track the flow of information. You may use the Workshop - Use Common Workflow Techniques as a guide to perform actions in the Task Monitor.

5. Navigate to **Part Maintenance**.

Menu Path: Material Management > Inventory Management > Setup > Part

6. In the **Part** field, search for parts that start with **00C**.

7. Verify the **Type** and **Description** fields for the parts you work with are updated appropriately.

Database Operations

In Service Connect, you can run SQL statements against a database.

Use the **DBOperation** workflow element to run one or several SQL statements (SELECT, INSERT, DELETE, UPDATE, or EXEC) against one or several databases.

You can use DBOperation against any database that has ODBC driver to perform the following actions:

- Read, write, and update third party databases during ESC workflow execution; this is required for data synchronization.
- Read values from application databases "on the fly"; this can be useful if one or several fields are not returned by the standard Business Object methods.
- Read additional Business Object values from custom user tables.

You can use part of an XML document in the SQL statements to update or select database data. You can as well put the SELECT statement result into an XML document for further workflow processing.

Only the first data retrieval statement within the SQL commands batch marked as output returns data from a single DBOperation.

If you validate the workflow or save it, and ESC finds an empty statement inside the DBOperation, the validation result dialog displays a warning message.

When you execute SQL statements, the DB Operation element logs each SQL statement and connection string that executes against a database into the ESC Events log. If during input message processing ESC encounters a DBOperation with an empty statement, a warning message is added to ESC Events log.

Workshop - Use DBOperation Workflow Element

In this workshop, use the .NET Call to create a new customer record in the Epicor application. Once the new customer record is created, use the DBOperation workflow element to run a simple query against the database to retrieve data. As a result, an .xml file containing all customer records will be dropped in the output folder you specify.

Create Folders and Sample Data

1. On your local drive, create the following folders:

C:\ESCSamples\XXX_CustomerUpdate\IN (where XXX are your initials)

C:\ESCSamples\XXX_CustomerUpdate\OUT

2. In the **XXX_CustomerUpdate** (where XXX are your initials) folder, create a spreadsheet in Microsoft Excel® with the sample data from the table below.

CustID	CustName	Currency	TerritoryID
XXX_Cust (where XXX are your initials)	XXX Customer (where XXX are your initials)	USD	US MID

3. Save the spreadsheet as **XXX_CustomerUpdate.xls** (where XXX are your initials).



Important Verify the file is saved in the **XXX_CustomerUpdate** (where XXX are your initials) folder and not inside the **IN** or **OUT** folder.

4. Exit Microsoft Excel.

Add a Message Type

1. In the **Epicor Service Connect Administration Console**, navigate to **Connectivity > Message attributes**.
2. Right-click **Message Types** and select **Add new Message Type**.
The **Add New Message Type** window displays.
3. In the **Message type name** and **Message type description** fields, enter **XXX_Customer** (where XXX are your initials).
4. Click the **Add** button.
The **New Action** window displays.
5. In the **Action name** field, enter **UpdateCustomer**.
6. In the **Action description** field, enter **Update Customer**.
7. In the **New Action** window, click **OK**.
8. In the **Add New Message Type** window, click **OK**.

Add a Sender

1. Right-click **Senders** and select **Add New Sender**.
The **Add New Sender** window displays.
2. In the **Sender name** and **Sender description** fields, enter **XXX_CRM** (where XXX are your initials).
3. Click the **Add** button.
The **New Sub-sender** window displays.
4. In the **Sub-sender name** field, enter **XXX_AccountMgr** (where XXX are your initials).

5. In the **Sub-sender description** field, enter **Account Manager**.
6. In the **New Sub-sender** window, click **OK**.
7. In the **Add New Sender** window, click **OK**.

Add an Input Channel

1. In the **Epicor Service Connect Administration Console**, navigate to **Connectivity > Communication Setup > your server name > Channels > Input Channels**.
2. Right-click **Input Channels** and select **Add New**.
The **Channel properties** window displays.
3. In the **Channel name** field, enter **XXX_CustomerUpdate** (where XXX are your initials).
4. In the **Listener type** field, select **FILE**.
5. Select the **Use scan interval** check box and accept the default value of **1 seconds**.
6. Click the **Configure** button.
The **Channel configuration** window displays.
7. Verify the **Message properties** sheet displays and select the following information:

Field	Data
SenderName	XXX_CRM (where XXX are your initials)
SenderSubName	XXX_AccountMgr (where XXX are your initials)
MsgType	XXX_Customer (where XXX are your initials)
Action	UpdateCustomer

8. Navigate to the **Communicator properties** sheet and enter the following information.



Tip On many of these fields, there is a button to open a window where you can edit or select the value.

Field	Data
File path	C:\ESCSamples\XXX_CustomerUpdate\IN (where XXX are your initials)
Mask	*.xls
Conversion	excel2xml.dll

9. In the **Channel Configuration** and **Channel properties** window, click **OK**.

Add an Output Channel

1. In the **Epicor Service Connect Administration Console**, navigate to **Connectivity > Communication Setup > your server name > Channels > Output Channels**.
2. Right-click **Output Channels** and select **Add New**.
The **Channel properties** window displays.
3. In the **Channel name** field, enter **XXX_CustomerQuery** (where XXX are your initials).
4. In the **Speaker type** field, select **FILE**.
5. Click the **Configure** button.
The **Channel configuration** window displays.
6. Verify the **Message properties** sheet displays.
7. In the **Code page** row, in the **Value** field, select **OEM**.
8. Navigate to the **Communicator properties** sheet and enter the following information:

Field	Data
File Path	C:\ESCSamples\XXX_CustomerUpdate\OUT (where XXX are your initials)

9. Accept all other defaults and click **OK**.
10. In the **Channel properties** window, click **OK**.

Import .Net Reference



Important If the **Customer** assembly is already imported in your environment, skip this task and continue with the following task **Create the Workflow**.

1. In the **Epicor Service Connect Administration Console**, navigate to **Connectivity > .NET References**.
2. Right-click **.NET References** and select **Add Reference**.
The **Add .NET reference** wizard displays.
3. Click **Next**.
4. In the **Assembly type** field, select **Epicor 10 assembly**.
5. Next to the **Assembly path** field, click the ... (browse) button.
The **Select assembly** window displays.
6. Navigate to the client folder on the server, for example C:\Epicor\ERP10\ERP10.0.300\ClientDeployment\Client



Note If necessary, contact your system administrator for help.

7. In the **File type** field, select **Epicor 10 .NET Assemblies (Erp.Contracts.BO.*.dll)**.
8. In the **Select assembly** window, select **Erp.Contracts.BO.Customer.dll** and click **OK**.
This is the customer business object assembly that will update the customer table.
9. Notice, the **Reference name** field displays **Customer** by default.
10. In the **Add .NET Reference** window, click **Next**.
The **Logon to Epicor** screen displays.
11. Enter the following information:

Field	Data
Client config	C:\Epicor\ERP10\ERP10.0.300\ClientDeployment\Client\Config\ERP10.sysconfig
User	enter a valid user, for example, manager
Password	enter a valid password for the user, for example, manager
Company	enter a company, for example, EPIC06
Plant	enter a plant (site) name, for example, MfgSys

12. On the **Logon to Epicor** screen, click **Next**.
13. On the **Import .NET reference** screen, verify the information and click **Next**.
The import process begins and may take a while.
14. Once the **All methods of .NET assembly are imported. Reference is successfully imported** information displays, click **Finish**.

Create the Workflow

1. In the **Epicor Service Connect Administration Console**, navigate to **Connectivity > Workflow packages**.
2. Right-click **Workflow packages** and select **New package**.
3. In the **Package name** field, enter **XXX_Customer** (where XXX are your initials).
4. Click **OK**.
5. Right-click the **XXX_Customer** (where XXX are your initials) package and select **New**.
The **Epicor Service Connect Workflow Designer** displays.
6. On the Standard toolbar, click **Save**.
7. To the **Workflow Designer** message, click **Yes**.
The **Save New Workflow** window displays.
8. In the **Save workflow as** field, enter **XXX_Customer** (where XXX are your initials) and click **Save**.

Add Elements to the Workflow

1. In the **Items** toolbar, click the **Conversion** button.
2. To the right of the **Start** workflow element, click in the workflow design area.
The **Properties** window displays.
3. In the **Properties** window, click **OK**.
4. In the **Items** toolbar, click the **.NET Call** button.
5. To the right of the **Conversion**, click the workflow design area.
The **Properties** window displays.
6. In the **Properties** window, click **OK**.
7. In the **Items** toolbar, click the **DBOperation** button.
8. To the right of the **.NET Call**, click the workflow design area.
The **Properties** window displays.
9. In the **Properties** window, click **OK**.
10. In the **Items** toolbar, click the **Poster** button.
11. To the right of the **DBOperation** activity, click in the workflow design area.
The **Properties** window displays.
12. In the **Properties** window, click **OK**.
13. In the **Items** toolbar, click the **Connection** button.
14. Make the following connections:
 - **Start** to **Conversion**
 - **Conversion** to **NETCall**
 - **NETCall** to **DBOperation**
 - **DBOperation** to **Poster**
 - **Poster** to **Finish**
15. On the Standard toolbar, click **Save**.
16. To the **Workflow Designer** message, click **Yes**.

Generate a Schema

1. In the Workflow Designer, from the **Tools** menu, select **Generate Schema from Sample Data**.
The **Generate Schema - General** window displays.
2. Next to the **Sample data file** field, click the browse button (...).

3. In the **Open** window, navigate to the C:\ESCSamples\XXX_CustomerUpdate (where XXX are your initials) folder and select the **XXX_CustomerUpdate.xls** file.
4. Click **Open**.
5. In the **Generate Schema - General** window, next to the **Generated schema** field, click the browse button (...).
6. In the **Save Schema** window, in the **File name** field, verify **XXX_CustomerUpdate.xsd** (where XXX are your initials) is displayed.
7. Click **OK**.
8. Verify, the **Use conversion settings from input channel** option is selected.
9. Click **Next**.
10. In the **Generate Schema - Input Channel** window, select the **XXX_CustomerUpdate** (where XXX are your initials) channel.
11. Click **Next**.
12. In the **Generate Schema - Generate Schema** window, click **Finish**.

Define the Conversion

1. In the Workflow Designer, double-click the **Conversion** workflow element.
2. In the **Properties** window, verify the **General** sheet is displayed.
3. Next to the **Input Schema** field, click the **Browse** button.
The **Select a schema** window displays.
4. In the left pane, select **User Schemas**.
5. In the right pane, select the **XXX_CustomerUpdate.xsd** (where XXX are your initials) schema and click **Next**.
6. In the **Specify the root element** window, accept the default values and click **Finish**.
7. Next to the **Output schema** field, click the **Browse** button.
The **Select a schema** window displays.
8. In the left pane, click **.NET Reference Schemas**.
9. In the right pane, double-click the **Customer** folder.
10. Select **Erp_Proxy_BO_CustomerImpl_UpdateExt_Request.xsd** and click **Next**.
11. In the **Specify the root element** window, accept the default values and click **Finish**.
Remain in the **Properties** window.

Edit the Conversion

1. Next to the **Conversion** field, click the **Edit** button.
The **New Transformation - Conversion type** window displays.
2. On the left side of the **New Transformation** window, expand the following nodes: msg > req > dta > table > row
3. On the right, expand the following nodes: msg > req > dta > Erp_Proxy_BO_CustomerImpl_UpdateExt_Request > ds > UpdExtCustomerDataSet > Customer
4. Create the following mappings:

Left Node	Right Node
CustID	CustID
CustName	Name
TerritoryID	TerritoryID
Currency	Currency Code

The next step is to set literal values for the remaining required fields.



Note The required fields display + (plus) sign in their icons.

5. In the right pane, right-click the **Company** field and select **Set Literal value**.
The **Set Literal value** window displays.
6. In the **Literal value** field, enter **EPIC06** and click **OK**.
7. Assign the following literal values:

Field	Literal Value
CustNum	1
TermsCode	2/10
CustomerType	CUS
BillFrequency	W



Tip To locate a particular node use the standard Find (Ctrl+F) functionality.

8. On the Standard toolbar, click **Save**.
The **Save Transformation File** window displays.
9. In the **Save Transformation File** window, enter **XXX_CustomerUpdate.xslt** (where XXX are your initials) and click **OK**.
10. Exit the XML Mapper.

11. In the **Properties** window, navigate to the **Appearance** sheet.
12. In the **Caption** field, enter **Customer Update** and click **OK**.

Define the .NET Call

1. In the Workflow Designer, double-click the **NETCall** workflow element.
2. In the **Properties** window, verify the **General** sheet is displayed.
3. Next to the **Request ID** field, click the **Select** button.
4. Expand the **Customer > Erp.Proxy.BO.CustomerImpl** node.
5. From the list, select **CustomerImpl.UpdateExt** and click **OK**.
6. Navigate to the **Appearance** sheet.
7. In the **Caption** field, enter **Customer Update Call** and click **OK**.
8. On the Standard toolbar, click **Save**.
9. To the Workflow Designer message, click **Yes**.

Define the DBOperation Workflow Element

1. In the Workflow Designer, double-click **DBOperation**.
The **Properties** window displays.
2. In the **Input Schema** field, verify the inherited schema is displayed.
http://scshost/schemas/ImportedAssemblies/Customer/Erp_Proxy_BO_CustomerImpl_UpdateExt_Response.xsd
3. Navigate to the **General** sheet.
4. Click the **Add** button.
The **SQL Statement** window displays.
5. Under the **Connection string** section, click the ... (browse) button.
The **Data Link Properties** window displays.
6. On the **Provider** tab, select the **OLE DB Provider** you use to connect to the application database and click **Next**.
In this example, select **Microsoft OLE DB Provider for SQL Server**.
7. On the **Connection** tab, enter the data required to set up connection to the application database.
For a **SQL Server** database connection, enter the following data:
 - Select your server name.
 - Enter the user name and a password to connect to the server (or use the Windows NT authentication option).
 - Select the appropriate database.



Example For a **Progress** database connection, enter the following data:

- Select your data source name.
- Enter the user name and a password to connect to the server (for example, sysprogress/sysprogress).
- Select the Allow saving password check box.
- Select the appropriate database.

8. In the **Select or enter a server name** field, enter **localhost** or just a dot (.).
9. Select the **Use Windows NT Integral security** option.
10. Select the **Select the database on the server** option and select the **ERP10**.
11. Click the **Test Connection** button.
12. If the connection is successful, to the **Test connection succeeded** message, click **OK**.
13. In the **Data Link Properties** window, click **OK**.
14. If the **Warning message** displays, click **OK**.
Remain in the SQL Statement window.

Build the Statement

1. In the **SQL Statement** window, click the **Select** button.
The **Table selection** window displays.
2. From the **Tables** list, select **Customer**.
Notice the **Columns** section populates with all available fields within the table.
3. In the **Table selection** window, click **OK**.
Notice the **SQL Statement** field displays the query that returns all fields from the Customer table.



Important When you run the statement against the **Progress** database, delete the **SQL Statement** and enter the following statement: **SELECT * FROM "<your database name>". "<owner name>". "<table name>"**

Example: SELECT * FROM "MFGSYS"."PUB"."Customer"

This must be done due to syntax and database structure differences between MS SQL and Progress.

4. In the **SQL Statement** field, replace the list of fields with an asterisk (*) to make the statement more concise.
SELECT * FROM ERP10.Erp.Customer
WHERE (1 = 1)
5. Select the **Output statement** check box.
6. In the **SQL Statement** window, click **OK**.
7. To the **Output schema** message, click **OK**.

The **Save Schema** window displays.

8. Click **User Schemas**.
9. In the **File name** field, enter **XXX_CustomerQuery.xsd** (where XXX are your initials) and click **OK**.
10. In the **Properties** window, click **OK**.

Define the Poster

1. In the Workflow Designer, double-click the **Poster** workflow element.
The **Properties** window displays.
2. Navigate to the **Fixed Channels** tab.
3. Click the **Add** button.
The **Select Channels** window displays.
4. Select **XXX_CustomerQuery** (where XXX are your initials) and click **OK**.
5. In the **Properties** window, click **OK**.
6. On the Standard toolbar, click **Save**.
7. Exit the Workflow Designer.

Add a Message Map

1. In the **Epicor Service Connect Administration Console**, navigate to **Connectivity > Message Map**.
2. Right-click **Message Map** and select **Add new Request**.
The **New Request ID** window displays.
3. In the **Sender name** field, select **XXX_CRM** (where XXX are your initials).
4. In the **Sender subname** field, select **XXX_AccountMgr** (where XXX are your initials).
5. In the **Message type** field, select **XXX_Customer** (where XXX are your initials).
6. In the **Message action** field, verify **UpdateCustomer** displays.
7. Next to the **Request ID** field, click the **Select** button.
The **Request ID** window displays.
8. Clear the **Channels**, **Web Methods** and **.NET Methods** check boxes.
9. In the **RequestID** column, select the row with **XXX_Customer\XXX_Customer** (where XXX are your initials) and click **OK**.
10. In the **New Request ID** window, click **OK**.

Run the Workflow

1. On your local drive, browse to the following folder: C:\ESC\Samples\XXX_CustomerUpdate
2. Copy the **XXX_CustomerUpdate.xls** file into the C:\ESC\Samples\XXX_CustomerUpdate\IN folder (where XXX are your initials).
Notice the file disappears as Service Connect consumes it.
3. Navigate to the **Epicor Service Connect Administration Console**.
4. Expand the **Document Tracking > Inbound Messages** node.
5. View the **Processing status(es)** column and click the **Refresh** icon to refresh the information.
6. The status of your workflow execution should be **Complete**.
7. Double-click the status to view its details.
Address any issues if necessary.
8. In the **Activity Progress** window, click **OK**.

Verify the New Customer Record

Navigate to the Epicor application.

Navigate to **Customer Maintenance**.

Menu Path: Sales Management > Order Management > Setup > Customer



Tip The CRM menu path is: Customer Relationship Management > Order Management > Setup > Customer

1. In the **Customer** field, search for and select **XXX Customer** (where XXX are your initials) record.
2. Recall the source data from the spreadsheet and literal values you specified and view the record.
3. Exit Customer Maintenance.

View the Database Output

1. On your local drive, browse to the following folder: C:\ESC\Samples\XXX_CustomerUpdate\OUT (where XXX are your initials).
2. Open the .xml file in a text editor.
For example, in the Notepad.
3. View the response from the database.
4. Optionally, you can search for the **XXX_Cust** (where XXX are your initials) customer record you created.
5. Exit the .xml file.

Conclusion

Congratulations! You have completed the Epicor Service Connect for Epicor ERP course.



Additional information is available at the Education and Documentation areas of the EPICweb Customer Portal. To access this site, you need a Site ID and an EPICweb account. To create an account, go to <http://support.epicor.com>.