

TIBCO Business Studio™

iProcess Implementation Guide

Software Release 3.5.1
February 2012

Important Information

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document contains confidential information that is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of TIBCO Software Inc.

TIBCO, The Power of Now, TIBCO ActiveMatrix, TIBCO Business Studio, TIBCO iProcess, TIBCO BusinessWorks, TIBCO PageBus are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

EJB, Java EE, J2EE, JMS and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

The BPMN logo is a trademark or registered trademark of the Object Management Group, Inc. in the United States and other countries.

XPDL and Powered by XPDL are trademarks of the Workflow Management Coalition.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

THIS SOFTWARE MAY BE AVAILABLE ON MULTIPLE OPERATING SYSTEMS. HOWEVER, NOT ALL OPERATING SYSTEM PLATFORMS FOR A SPECIFIC SOFTWARE VERSION ARE RELEASED AT THE SAME TIME. SEE THE README.TXT FILE FOR THE AVAILABILITY OF THIS SOFTWARE VERSION ON A SPECIFIC OPERATING SYSTEM PLATFORM.

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

Copyright © 2004-2012 TIBCO Software Inc. ALL RIGHTS RESERVED.

TIBCO Software Inc. Confidential Information

Contents

Preface	ix
Changes from the Previous Release of this Guide	x
Related Documentation	xi
TIBCO Business Studio Documentation	xi
Other TIBCO Product Documentation	xii
Third Party Documentation	xii
Typographical Conventions	xiii
Connecting with TIBCO Resources	xvi
How to Join TIBCOCommunity	xvi
How to Access All TIBCO Documentation	xvi
How to Contact TIBCO Support	xvi
 Chapter 1 Getting Started	 1
Who Should Use TIBCO Business Studio?	2
How TIBCO Business Studio Supports MDA	3
The Solution Design Capability	4
Labels and Names	4
Displaying Implementation Details	5
Implementation Approach	7
Hand Over from Business Analyst	7
BPM/SOA Implementation	7
Service Creation/Testing for Sub-Processes	8
Hand Off/Deployment	8
Process Testing	8
BPM/SOA Implementation Overview	9
User Tasks	9
Service Tasks	10
Script Tasks	14
Complete the Process Details	15
Set the Destination Environment	15
Deploying a Process	16
 Chapter 2 Tutorials	 17
iProcess Developer Samples Project	18
Tutorial 1: Working with Destination Environments	19

Tutorial 2: Elaborating a Process: User Tasks	24
Tutorial 3: Automatically Creating Forms	30
Tutorial 4: Elaborating a Process: Service Tasks	31
Calling Web Services	31
Creating an E-Mail Task	36
Tutorial 5: Adding a Script Task	39
Gather User Details and Apply Core Logic	40
Add Branching	42
Call iProcess Functions	42
Tutorial 6: Creating Dynamic Sub-Processes	45
Looking at the Example Process	45
Example Case	51
Tutorial 7: Deploying a Process	54
Tutorial 8: Advanced Deployment	59
Tutorial 9: Exporting to the TIBCO iProcess Modeler	68
Chapter 3 TIBCO Business Studio for iProcess Users	73
iProcess Examples in BPMN	74
Using the Advanced Tab	75
Invocation Styles	76
Configuring Web Services	78
Abstract and Concrete WSDL Files	79
Using Aliases	80
Using Scripts for Mapping	81
Data Transport Mechanisms	81
Configuring BusinessWorks Service Tasks	82
Implementing TIBCO iProcess Conductor Processes	84
Orchestrator Service Tasks	84
Order Service Tasks	85
Transformation Steps	85
Customizing the Audit Trail	87
Immediate Release	87
Delayed Release	88
Transaction Control Steps	89
Deferring New Transactions	89
Example 1	90
Example 2	91
User Tasks	93
Forms	93
Scripts	93
Priority and Step Permissions	94

Deadline Expressions	97
Withdraw Links	98
Compensation Events	102
Dynamic Sub-Procedures	103
Implementing this Example in TIBCO Business Studio	104
Graft Steps	106
Implementing this Example in TIBCO Business Studio	107
Reusable Sub-Process Call Task Validation	109
Groups and Roles	110
Public Steps and Events	111
Object Mapping	112
Project Objects	112
Chapter 4 Configuring User Tasks	119
Adding Parameters and Data Fields	120
Specifying a Form	122
Using Performer Data Fields	124
Converting a Participant into a Performer Data Field	126
Chapter 5 Configuring Service Tasks	127
Working with Service Registries	128
Adding a UDDI Registry	128
Viewing a Registry	128
Creating a Registry Search	128
Changing the Properties of a Registry or Search	130
Working with WSDL Files	131
WSDL File Requirements	131
Generating a New WSDL File	132
Copying a WSDL File	134
Dragging a WSDL File From the Registry	135
Creating a JMS Server (TIBCO BusinessWorks Only)	136
Connecting to a JMS Server (TIBCO BusinessWorks Only)	138
Importing a WSDL File/Service Description	139
Validating a WSDL File	144
Configuring a BusinessWorks or Web Services Service Task	145
BusinessWorks Endpoint Resolution	146
Web Services Endpoint Resolution	147
Using the Mapper	149
Array to Array Mapping	150
Applying Scripts to a Mapping	150

Deleting Mappings	152
Changing Mappings	152
Mapping Date and Time Parameters	152
Sending an Email	156
Making a Database Call	160
Creating a Database Connection Profile	160
Connecting to the Database	163
Working Offline	163
Configuring the Service Task	164
Configure the Advanced Connection Properties	165
Mapping Existing Process Data	166
Creating a Business Object Model From Returned Data	168
Calling Java Code	172
Complete the Parameter Mapping	174
Java Deployment	175
Configuring POJO in TIBCO Business Studio	177
Changing the Exception Mode	177
Changing Conversion Defaults Between iProcess and Java	177
Configuring TIBCO iProcess Conductor Service Tasks	180
Orchestrator Service Tasks	181
Order Service Tasks	182
Chapter 6 Working with Scripts	185
Creating a Script Task	186
Entering JavaScript	187
Transforming Data (XPath and XSLT)	188
Using iProcess Expressions and Functions	194
iProcess Script File Functions	194
Obtaining iProcess User Attributes	194
iProcess Functions With Optional Parameters	194
Using iProcess Array Functions	195
Using iProcess System Fields	195
Creating a Script for a User Task	199
Creating Audit Scripts	202
Associating a Script with a Conditional Flow	204
Timer Event Scripts	205
Customizing JavaScript Presentation Preferences	207
Customizing XPath Presentation Preferences	209
Chapter 7 Validating Processes	213
Working with Destination Environments	214

Destination Components	215
Correcting Validation Errors	217
Turning off Validation	218
Chapter 8 Deploying to the iProcess Engine	219
Deployment Overview	220
Preparing a Process for Deployment	222
Package the Process	222
Verify Package/Process Naming	222
Set the Destination and Correct Problems	223
Creating a New iProcess Server	225
Specify the Server Name and Runtime	225
Enter the Runtime Server Parameters	225
Enter the MBean Server Configuration Parameters (Optional)	227
Enter the Workspace (Browser) Server Configuration Details	229
Creating a New Workspace Server	231
Connecting to a Server	232
Troubleshooting	232
Deploying a Module	235
Drag and Drop Deployment	235
Deploying from the Server Menu	237
Starting a Case From Within TIBCO Business Studio	240
Opening the TIBCO iProcess Workspace (Browser)	242
Managing Deployed Modules	244
Changing Server Properties	245
Disconnecting from the Server	246
Chapter 9 Import and Export - iProcess Modeler	247
Integrating with the TIBCO iProcess Suite	248
Importing a TIBCO iProcess Modeler Package	250
Chapter 10 Reference	253
Overview	254
Properties View	255
Process Properties (Advanced Tab)	256
User Task Properties	259
Service Task Properties (Web Service/BusinessWorks Service)	261
Service Task Properties (EAI)	266
Service Task Properties (Email)	267

Service Task (Database) 269

Service Task (Java) 271

Service Task (TIBCO iProcess Conductor - Orchestrator) 272

Service Task (TIBCO iProcess Conductor - Order) 275

Embedded Sub-Process (Under Transaction Control) 277

Reusable Sub-Process Tasks (Calling Sub-Processes) 278

Reusable Sub-Process Tasks (Calling Process Interfaces) 279

Send and Receive Tasks 280

XOR Gateway (With Outgoing Conditional Flow) 281

Intermediate Events 282

TIBCO iProcess System Fields 283

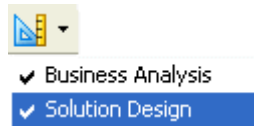
Index 285

Preface

This guide is aimed at the solution engineer who implements business processes designed by a business analyst. It describes how to elaborate a TIBCO Business Studio™ process with execution details and how to deploy it to the TIBCO iProcess™ Engine.



The features described in this guide are available in the Solution Design capability. To change between this capability and the Business Analysis capability, choose from the following menu on the toolbar:



There are also links in the Properties view for certain types of objects (for example, service tasks and script tasks) that allow you to view or hide the implementation details of the selected object:



Topics

- [Changes from the Previous Release of this Guide on page x](#)
- [Related Documentation on page xi](#)
- [Typographical Conventions, page xiii](#)
- [Connecting with TIBCO Resources, page xvi](#)

Changes from the Previous Release of this Guide

This section itemizes the major changes from the previous release of this guide.

- The section 'Mapping the Result Set' has been removed.
- A note has been added about support for array to array mapping and how to enable it. See [Array to Array Mapping on page 150](#).
- You cannot choose to import some processes in a process and sub-process family into TIBCO Business Studio, and then only partially re-deploy them. See the note on this in [Preparing a Process for Deployment on page 222](#).
- A helper function is provided to help you to translate iProcess script constructs to Javascript. See [Importing a TIBCO iProcess Modeler Package on page 250](#).
- There is support for iProcess custom EAI step types, and other standard iProcess EAI step types (such as EAI Java and the other EAI steps that are based on it) that were not previously supported in TIBCO Business Studio. See [Service Task Properties \(EAI\) on page 266](#).

Related Documentation

This section lists documentation resources you may find useful.

TIBCO Business Studio Documentation

The following documents form the TIBCO Business Studio documentation set:

- *TIBCO Business Studio Process Modeling User's Guide* Read this manual to learn how to create a model of a business process using Business Process Modeling Notation (BPMN).
- *TIBCO Business Studio Simulation User's Guide* Read this manual to learn how to simulate a process that has been developed in TIBCO Business Studio to identify areas of the process that can be improved.
- *TIBCO Business Studio Business Object Modeling User's Guide* Read this manual to learn how to create a set of business terms and relationships specific to your corporate environment (a Business Object Model).
- *TIBCO Business Studio Organizational Modeling User's Guide* Read this manual to learn how to create a model of the resources (people) who work for the organization and their structural relationships to one another.
- *TIBCO Business Studio Forms User's Guide* Read this manual for information about modeling and deploying forms.
- *TIBCO Business Studio iProcess Implementation Guide* Read this manual to learn how to elaborate a TIBCO Business Studio process with execution details and how to deploy it to the iProcess™ Engine.
- *TIBCO Business Studio Customization* Read this manual to learn about the areas of TIBCO Business Studio that can be customized or extended.

The following documentation is also provided:

- *TIBCO Business Studio Installation Guide* Read this manual for instructions on site preparation and installation.
- *TIBCO Business Studio Release Notes* Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

Other TIBCO Product Documentation

You may find it useful to read the documentation for the following products:

- TIBCO iProcess™ Suite, a comprehensive collection of process management software. Processes developed in TIBCO Business Studio can be directly deployed to the TIBCO iProcess Engine, or imported to the TIBCO iProcess™ Modeler.
- TIBCO General Interface™ Builder, a development environment for building rich internet applications. TIBCO General Interface is used for creating Forms in TIBCO Business Studio.
- TIBCO BusinessWorks™, a scalable, extensible, and easy to use integration platform that allows you to develop integration projects. BusinessWorks™ Processes that are exposed as services can be called by a TIBCO Business Studio service task.

Third Party Documentation

The Eclipse help also contains useful information on the Workbench and the Eclipse UI.

Typographical Conventions

The following typographical conventions are used in this manual.

Table 1 General Typographical Conventions

Convention	Use
<i>TIBCO_HOME</i> <i>STUDIO_HOME</i>	Many TIBCO products must be installed within the same home directory. This directory is referenced in documentation as <i>TIBCO_HOME</i> . The default value of <i>TIBCO_HOME</i> depends on the operating system. For example, on Windows 7 systems, the default value is C:\Program Files (x86)\tibco TIBCO Business Studio installs into a directory within < <i>TIBCO_HOME</i> >. This directory is referenced in documentation as <i>STUDIO_HOME</i> . The default value of <i>STUDIO_HOME</i> depends on the operating system. For example on Windows 7 systems, the default value is C:\Program Files (x86)\TIBCO\studio-bpm-35.
code font	Code font identifies commands, code examples, filenames, pathnames, and output displayed in a command window. For example: Use MyCommand to start the foo process.
bold code font	Bold code font is used in the following ways: <ul style="list-style-type: none"> • In procedures, to indicate what a user types. For example: Type admin. • In large code samples, to indicate the parts of the sample that are of particular interest. • In command syntax, to indicate the default parameter for a command. For example, if no parameter is specified, MyCommand is enabled: MyCommand [enable disable]
<i>italic font</i>	Italic font is used in the following ways: <ul style="list-style-type: none"> • To indicate a document title. For example: See <i>TIBCO ActiveMatrix BusinessWorks Concepts</i>. • To introduce new terms For example: A portal page may contain several portlets. <i>Portlets</i> are mini-applications that run in a portal. • To indicate a variable in a command or code syntax that you must replace. For example: MyCommand <i>PathName</i>

Table 1 General Typographical Conventions (Cont'd)




Convention	Use
Key combinations	Key name separated by a plus sign indicate keys pressed simultaneously. For example: Ctrl+C. Key names separated by a comma and space indicate keys pressed one after the other. For example: Esc, Ctrl+Q.
	The note icon indicates information that is of special interest or importance, for example, an additional action required only in certain circumstances.
	The tip icon indicates an idea that could be useful, for example, a way to apply the information provided in the current section to achieve a specific result.
	The warning icon indicates the potential for a damaging situation, for example, data loss or corruption if certain steps are taken or not taken.

Table 2 Syntax Typographical Conventions

Convention	Use
[]	An optional item in a command or code syntax. For example: MyCommand [optional_parameter] required_parameter
	A logical OR that separates multiple items of which only one may be chosen. For example, you can select only one of the following parameters: MyCommand para1 param2 param3

Table 2 Syntax Typographical Conventions

Convention	Use
{ }	<p>A logical group of items in a command. Other syntax notations may appear within each logical group.</p> <p>For example, the following command requires two parameters, which can be either the pair param1 and param2, or the pair param3 and param4.</p> <pre>MyCommand {param1 param2} {param3 param4}</pre> <p>In the next example, the command requires two parameters. The first parameter can be either param1 or param2 and the second can be either param3 or param4:</p> <pre>MyCommand {param1 param2} {param3 param4}</pre> <p>In the next example, the command can accept either two or three parameters. The first parameter must be param1. You can optionally include param2 as the second parameter. And the last parameter is either param3 or param4.</p> <pre>MyCommand param1 [param2] {param3 param4}</pre>

Connecting with TIBCO Resources

How to Join TIBCOCommunity

TIBCOCommunity is an online destination for TIBCO customers, partners, and resident experts, a place to share and access the collective experience of the TIBCO community. TIBCOCommunity offers forums, blogs, and access to a variety of resources. To register, go to <http://www.tibcommunity.com>.

How to Access All TIBCO Documentation

After you join TIBCOCommunity, you can access the documentation for all supported product versions here:

<http://docs.tibco.com/TibcoDoc>

How to Contact TIBCO Support

For comments or problems with this manual or the software it addresses, please contact TIBCO Support as follows.

- For an overview of TIBCO Support, and information about getting started with TIBCO Support, visit this site:

<http://www.tibco.com/services/support>

- If you already have a valid maintenance or support contract, visit this site:

<https://support.tibco.com>

Entry to this site requires a user name and password. If you do not have a user name, you can request one.

Chapter 1 **Getting Started**

This chapter provides an overview of the BPM/SOA implementation approach espoused by TIBCO Business Studio and related products.

Topics

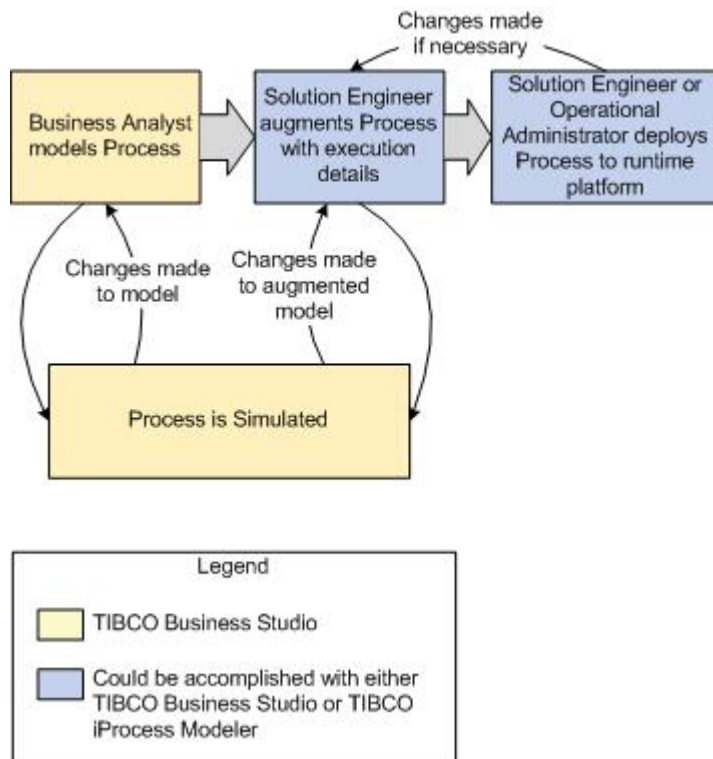
- [Who Should Use TIBCO Business Studio?, page 2](#)
- [The Solution Design Capability, page 4](#)
- [Implementation Approach, page 7](#)
- [BPM/SOA Implementation Overview, page 9](#)
- [Deploying a Process, page 16](#)

Who Should Use TIBCO Business Studio?

Typically a business analyst defines a new business process and the solution engineer must implement it. For example:

- An insurance company wants to enable their financial advisors to sell pension plans door-to-door using portable tablets. The solution engineer must implement a business process for doing this.
- A company has acquired a competitor. The business analyst has rationalized both companies' working procedures and created a process for a standard way of working. The solution engineer must implement this across the organization.

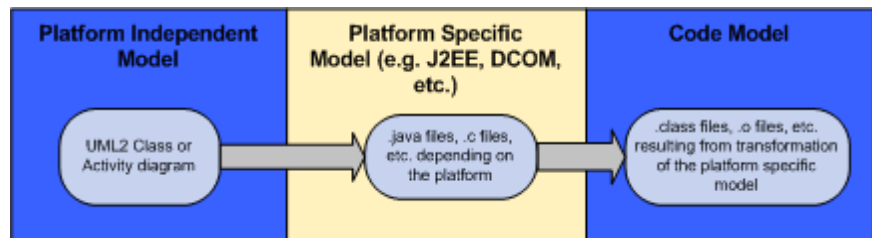
"Implementing a process" refers to taking the process that the business analyst has generated, refining it, adding detail to it, and taking it through the packaging and deployment phases. The following diagram shows how TIBCO Business Studio is intended to be used:



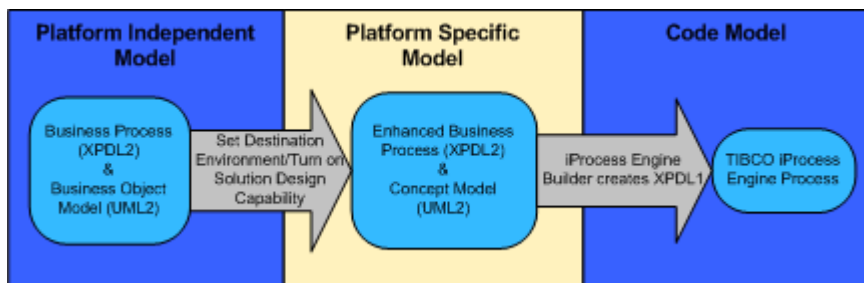
The following section describes how the abstract implementation of a process can be done first, and then the process can be transformed to create a code model that can be used with a specific execution engine.

How TIBCO Business Studio Supports MDA


Model Driven Architecture (MDA) is typically used to design applications and write specifications. It consists of a Platform Independent Model, usually written in Universal Modeling Language (UML), as well as a Platform Specific Model and Code Model, both of which are applied at the programming language level. For example:



TIBCO Business Studio supports an MDA approach to Business Process Management (BPM) and Service Oriented Architecture (SOA) at a higher level of abstraction than the programming language level by allowing you to develop Platform Independent Models of business processes and then transform them into Platform Specific Models, depending on the destination environment that you select. For example:



The Solution Design Capability

TIBCO Business Studio allows the solution engineer to show implementation details of a process that were previously hidden from the business analyst. To see the parts of the UI relevant to process implementation, either switch to the Solution Design capability from the toolbar () or click the provided links in the Properties view. For example:

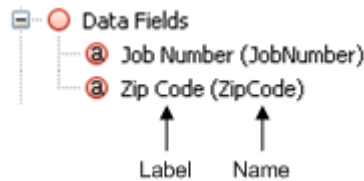
Activity Type:
[Provide Implementation Details](#)

Labels and Names

When business analysts (using the Business Analysis capability) create process objects such as data fields, task names, and so on, they assign the objects labels that may contain spaces or non-alphanumeric characters.



With the Solution Design capability selected, the Label as well as the Name is displayed. For example:



In scripts and sub-processes, you must use the Name to reference process data. If you migrate a process from an earlier version, the migration XSLT creates labels from the name.

Displaying Implementation Details

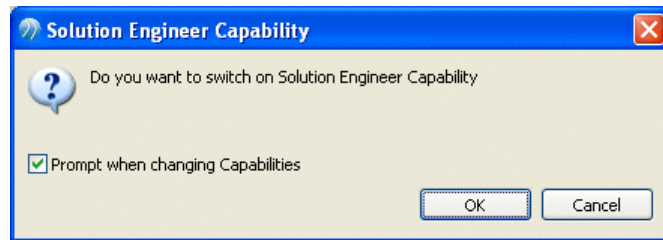
As mentioned previously, you can show and hide the implementation details of a process. For example, the business analyst view of a service task looks like this:

The screenshot shows the 'Task' properties window with the following details:

- Properties** | **Problems** | **Fragments**
- Task**
- General**
 - Label: Service Task
 - Description
 - Interface
 - Scripts
 - Appearance
 - Extended
 - Advanced
- Activity Markers: ☐ Standard Loop ☐ Multiple Instance Loop ☐ Ad-Hoc
- Participants: - not set -
- Activity Type: Service Task
- [Provide Implementation Details](#)

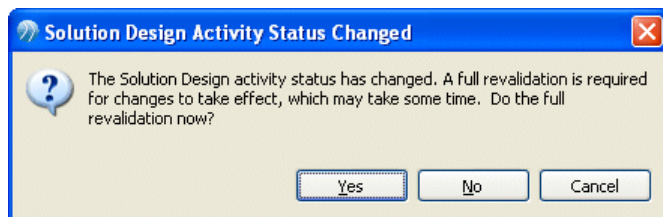
To show the implementation details do the following:

1. Click the Provide Implementation Details link. The following dialog is displayed:



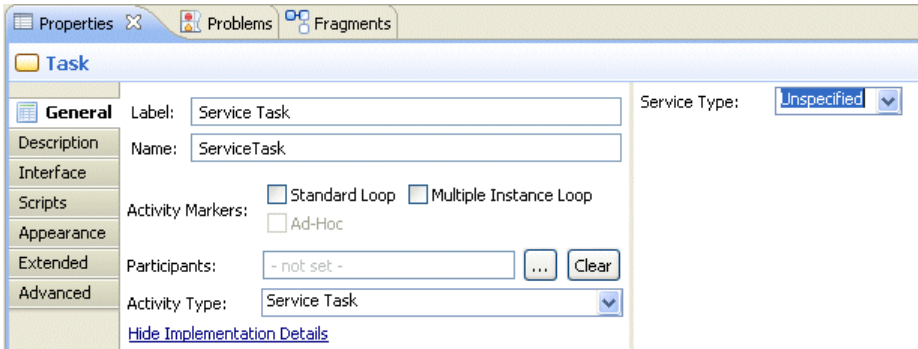
Click **OK**.

2. The following dialog is displayed:



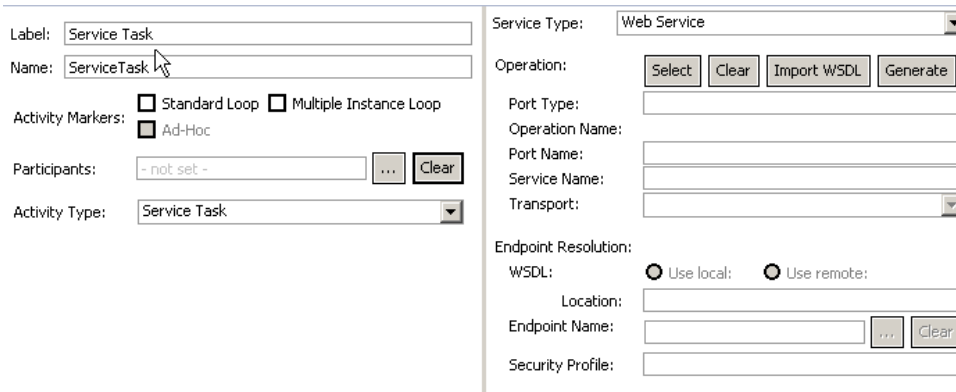
Click **Yes** to revalidate the workspace.

3. After revalidation, the solution design view shows the Service Type on the right of the Properties view:



Selecting a Service Type (such as Web Service) displays further implementation details, and additional tabs that would be hidden in the Business Analyst capability:

The Generate button gives the facility to generate a new WSDL file from a Service Task.



Implementation Approach

Implementing a process can be achieved in several different ways, however the following general approach reflects best practices.

Hand Over from Business Analyst

1. Receive process from business analyst.
2. Review process with business analyst.

BPM/SOA Implementation

For Service Tasks

There are two parts to implementing services:

- Selecting the **service contract**, which implies an abstract WSDL file. This is a mandatory part of implementing a service.
- Selecting a **service implementation**, which implies a concrete WSDL file. This is optional for technologies such as Web Services that support late binding.

To implement a service call, you must have a service that can come from one of the following sources:

- Select an existing service interface and optionally an implementation (WSDL file) from the Service Registry, file, URL, or BusinessWorks live link, then bind it to the process.
- Employ integration technology (such as BusinessWorks processes perhaps employing adapters) to create a new service and bind its service interface to the process.
- Write new custom service implementation using .NET, Enterprise Java Beans (EJBs), or other technologies, and bind its service interface to the process.

For User Tasks

- Specify the input and output parameters to the task.
- Use TIBCO Business Studio Forms to generate a form (see *TIBCO Business Studio Forms User's Guide*). If you create a form outside of TIBCO Business Studio (for example, a JSP page authored with the JSP editor provided by WTP inside Eclipse), specify a URL that refers to the location of the form.

Service Creation/Testing for Sub-Processes

Optionally, create any custom Java services that are required, for example using the Eclipse Integrated Development Environment (IDE).



In some organizations, service implementations are created by different people or groups from those who consume the services provided by the process implementation. This is especially likely when the service implementation is done in a programming language such as Java rather than a declarative technology such as BusinessWorks. This guide focusses on the work of the person consuming services; for more information on creating services, see the documentation for TIBCO BusinessWorks and TIBCO ActiveMatrix.

Test and debug the services created in Java using tools external to TIBCO Business Studio, either those embedded in the Eclipse Platform Development Environment (PDE) or any other tools that are available.

Hand Off/Deployment

There are several options depending on the procedures in your organization:

- Publish process documentation to web site and return the enhanced process model to Business Analyst and Business Owner for approval. See the *TIBCO Business Studio Process Modeling User's Guide* for more information about generating process documentation.
- Package process for deployment to User Acceptance Testing/Production by the Operations staff.
- Deploy the package/process to the repository and initialize it without handoff to Operations staff.

Process Testing

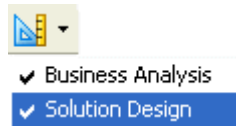
Ensure that the process functions as intended including starting cases, test queues, undelivered work items, and so on. For more information, see the TIBCO iProcess Modeler set of guides.

BPM/SOA Implementation Overview

This section provides an overview of how to augment the process with execution details. For example, you may want to call a web service from a service task. For more information about specific tasks, see subsequent chapters.



The features described in this guide are available in the Solution Design capability. To change between this capability and the Business Analysis capability, choose from the following menu on the toolbar:

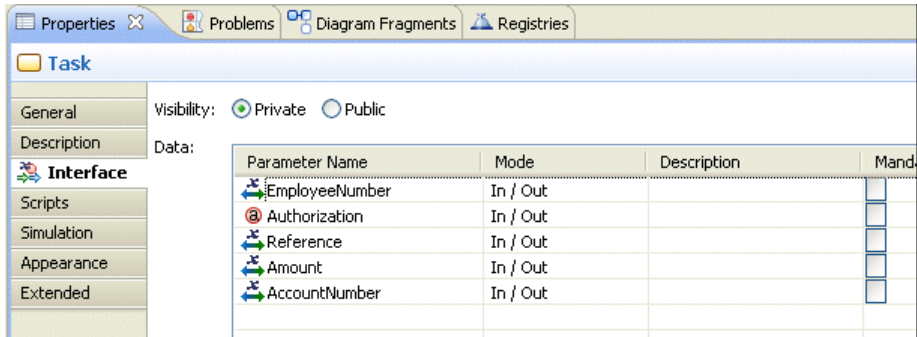


User Tasks

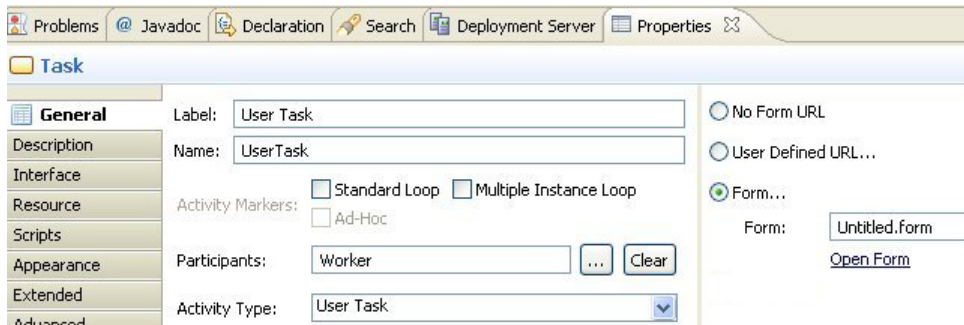
There are several aspects of a user task that you can specify in TIBCO Business Studio:

- Generate a form for the user task, using TIBCO Business Studio Forms. For more information see *TIBCO Business Studio Forms User's Guide*. Forms generated in other ways can be displayed by specifying either a relative or absolute URL (see [Configuring User Tasks on page 119](#)).

- You can specify the input and output parameters of a user task on the **Interface** tab of the Properties view for the user task:



On the **Interface** tab, you can also specify visibility of an event or task (whether it is private or public). This controls whether the event or task publishes its information (such as required parameters) to an external process or application.



Service Tasks

Service tasks can be implemented in the following ways:

- **Web Service** - Calls to a web service can be implemented as well as the mapping of input/output parameters.
- **Java** - Allows you to call Java code that you want to execute at runtime, including functions that emulate standard iProcess Script commands.
- **Database** - Native or Java Database Connectivity (JDBC) calls to Oracle, SQL and DB2 databases.



The iProcess database step only supports stored procedures.

- **E-mail** - You can either configure an email message that is sent when the process is executed, or provide a service descriptor (WSDL) for the behavior.
- **BW Service** - Calls to a BusinessWorks process exposed as a service can be implemented as well as the mapping of input/output parameters.

This is done in the **General** tab of the Properties view for the service task. For example:

The screenshot shows the 'General' tab of the Properties view for a service task. It contains the following fields and controls:

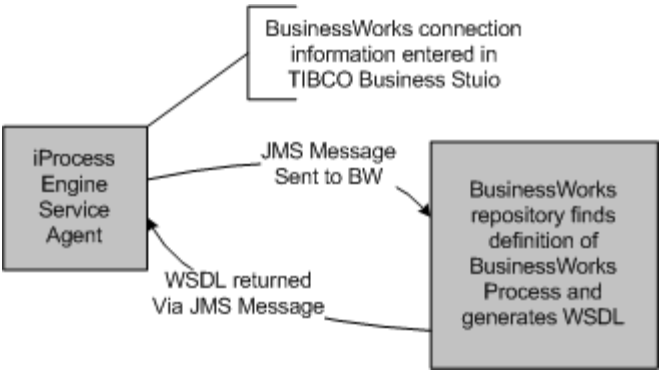
- Service Type:** A dropdown menu with 'Web Service' selected.
- Service Name:** A dropdown menu with 'Web Service' selected. To its right is a text input field.
- Port Name:** A dropdown menu with 'Java' selected. To its right is a text input field.
- Operation Name:** A dropdown menu with 'BW Service' selected. To its right is a text input field.
- Buttons:** 'Select' and 'Clear' buttons are located to the right of the Service Name and Port Name dropdowns.
- WSDL Options:** Two radio buttons are at the bottom: 'Use local WSDL' and 'Use remote WSDL'. Each has a corresponding text input field to its right.

TIBCO BusinessWorks/Web Service

A service task can be configured to call a service that has a WSDL file or XML schema (in the case of XML/JMS). Before you can configure a service task, you must import the WSDL file or schema description to the Eclipse workspace, from one of the following sources:

- **Descriptor for XML over JMS** - use this option if you are importing an XML or XSD document.
- **File** - browse to select a local WSDL file.
- **URL** - specify a URL from which the WSDL file is retrieved.
- **Uniform Description, Discovery, and Integration (UDDI) Registry** - obtain the WSDL file from a UDDI registry.

- BusinessWorks live link - use the TIBCO BusinessWorks Connector live link feature to dynamically create and import a WSDL file from a BusinessWorks process exposed as a service:



To use the TIBCO BusinessWorks live link invocation method, the iProcess Service Agent must be running and the JMS provider transport information must be configured. For more information, see the *TIBCO iProcess BusinessWorks Connector User's Guide*.

Except for a WSDL file obtained from a TIBCO BusinessWorks live link, the WSDL file that you use must conform to the following requirements for deployment to iProcess:

Process to Service Binding

In the Properties view for a service task, you can select either **Use local** or **Use remote**:

Endpoint Resolution:

WSDL: ☐ Use local: ☒ Use remote:

Location:

This is taken from the Alias URL at Run-time

Endpoint Name:

Bank

...

Clear

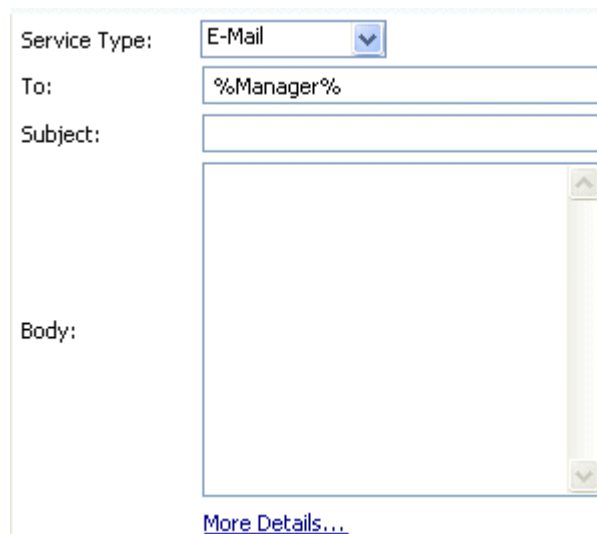
For a remote WSDL file, a logical name (defined as a system participant) is specified. At deployment, this logical name must be mapped to an actual endpoint name (retrieved from a JMX server). Note that although an abstract WSDL file can be associated with a service task, abstract WSDL files cannot be deployed to iProcess.

Database

A service task can be configured to perform an action on a database using stored procedure calls. When the process is deployed to iProcess, the appropriate DB2, Oracle or SQL database calls are made in the iProcess database. Although you do not need to be concerned about which database is used in conjunction with iProcess, the stored procedure writer will need to be familiar with the particular database.

E-mail

A service task can be configured to send an E-mail message on the **General** tab of the Properties view for the service task as follows:



The screenshot shows a configuration window for an E-Mail service task. It contains the following fields:

- Service Type:** A dropdown menu with "E-Mail" selected.
- To:** A text box containing the parameter "%Manager%".
- Subject:** An empty text box.
- Body:** A large, empty text area with a vertical scrollbar on the right.
- More Details...** A blue hyperlink located below the Body text area.

Clicking **More Details** or clicking the **E-Mail** tab allows you to specify further parameters. Note that instead of explicitly entering email addresses, subject lines and so on, you can define data fields or parameters for this purpose. For example, the previous dialog shows **%Manager%** rather than an explicit email address.

For more information, see [Sending an Email on page 156](#).

Java

A service task can be configured to call Java code on the **General** tab of the Properties view for the service task as follows:

Service Type:

Project:

Class:

Method:

Select a factory to create the class:

Class:

Method:



The Java service class and any parameters that are not primitives (for example, **int**, **float**, and so on) must adhere to JavaBeans semantics. Specifically they must have a default constructor (one without parameters) and the properties need to have standard **get** and **set** operations. This means that a String property 'name' must have an accessor named **getName():String** and a mutator named **setName(:String)**.

You can browse to select the Java class and select a method from the drop-down list. Alternatively you can specify a Factory to be used for the creation of the class. The factory class must either offer a static factory method or must have a default constructor as the service class does.

Script Tasks

You can enter scripts in script tasks, user tasks, on conditional sequence flows, on the **Audit** tab of service tasks, and to timer start or intermediate events.



For information about creating scripts, see [Working with Scripts on page 185](#).

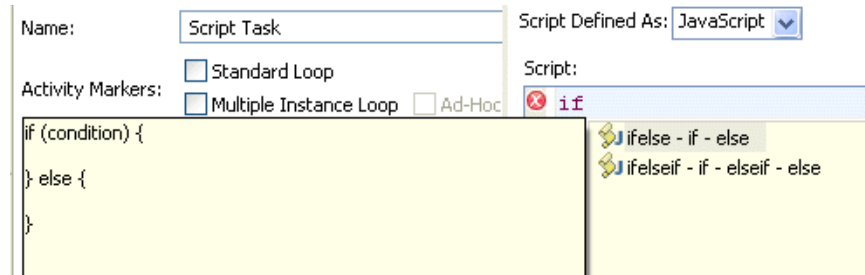
The following constructs are supported in TIBCO Business Studio for processes that you want to export/deploy to iProcess Modeler/iProcess Engine:

- If, elseif, and else statements
- Do while loops and while loops
- For loops

- Assignment operators
- Conditional operators

The result of an expression should correspond to one of the known data types listed in the *TIBCO iProcess Expressions and Functions Reference Guide*.

Content assist is available for process data that you define and also provides templates for common JavaScript constructs. For example, if you enter **if**, then press Ctrl+Space, you can use the following template to construct an if else:



iProcess Script is supported for backward compatibility via the TIBCO iProcess JavaScript class library. Within the packaging/export phase this JavaScript is converted to iProcess Script for execution by iProcess. If you have selected the **iProcess Engine** or **iProcess Modeler** destination environment, you can view the available list of iProcess Script functions by typing **IPE** and pressing Ctrl+Space.

Complete the Process Details

Besides the service tasks and user tasks, add any sub-process calls, mappings to sub-process parameters, events, and so on that your process requires. For more information about iProcess-specific constructs such as delayed release, deadline expressions, withdraw links, and so on, see [Chapter 3 on page 73](#).

Set the Destination Environment

To deploy directly to iProcess, select **iProcess** as the destination environment. The process is validated for deployment to the iProcess Engine.

Before you can deploy or export the process, you must correct any problems reported in the Problems view. For more information, see [Chapter 7 on page 213](#).

Deploying a Process

Deployment is part of the software development cycle (design, deploy, execute). After preparing the software, some transformation, packaging, physical delivery, configuration and initialization takes place. All of these, some of which may be optional, are aspects of deployment.

To look at a concrete example of deployment, after a solution engineer has elaborated a process, the next step could be to deploy it to a server (for example, a server in the test environment).

The primary way to deploy a process once you have finished elaborating it with the necessary execution details is to create a deployment server within TIBCO Business Studio and deploy the process directly to the iProcess Engine (see [Deploying to the iProcess Engine on page 219](#)).

You can also export the process to iProcess Modeler XPDL and then import it into iProcess Modeler (see the *TIBCO iProcess Modeler Integration Techniques*).



Direct deployment and export to the TIBCO iProcess Engine are one-way operations. Information can be lost if you deploy to the iProcess Engine, modify the process, re-import to TIBCO Business Studio, deploy again, and so on. This is because the TIBCO Business Studio process description contains more information than merely the execution information required for iProcess Modeler/iProcess Engine, and this extra information can be lost upon re-importing. For this reason, you should regard the import of an iProcess Modeler XPDL into TIBCO Business Studio as a one-off activity to evaluate or migrate to TIBCO Business Studio.

Chapter 2 **Tutorials**

This chapter contains tutorials that demonstrate aspects of elaborating and deploying a process to the iProcess destination environment.

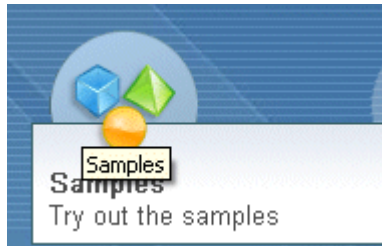
Topics

- [iProcess Developer Samples Project, page 18](#)
- [Tutorial 1: Working with Destination Environments, page 19](#)
- [Tutorial 2: Elaborating a Process: User Tasks, page 24](#)
- [Tutorial 3: Automatically Creating Forms, page 30](#)
- [Tutorial 4: Elaborating a Process: Service Tasks, page 31](#)
- [Tutorial 5: Adding a Script Task, page 39](#)
- [Tutorial 6: Creating Dynamic Sub-Processes, page 45](#)
- [Tutorial 7: Deploying a Process, page 54](#)
- [Tutorial 8: Advanced Deployment, page 59](#)
- [Tutorial 9: Exporting to the TIBCO iProcess Modeler, page 68](#)

iProcess Developer Samples Project

The sample processes on the Welcome page include a project (**iProcess Developer Samples**) that has the completed solutions for the tutorials in the following sections. You can either install the sample processes and explore the project, or continue with this tutorial to implement the example yourself. To install the TIBCO Business Studio sample processes, do the following:

1. Click **Help > Welcome**.
2. Click the **Samples** button.



This displays a page that lists samples that are available.

3. The TIBCO samples are grouped together under sections for the Business Analyst and Solution Designer.
4. To install the iProcess Developer Samples, click the link under **Solution Designer** and in the resulting dialog, click **Finish**.
5. Expand the **iProcess Developer Samples** project and expand the **Process Packages** folder, which contains the tutorial solutions.



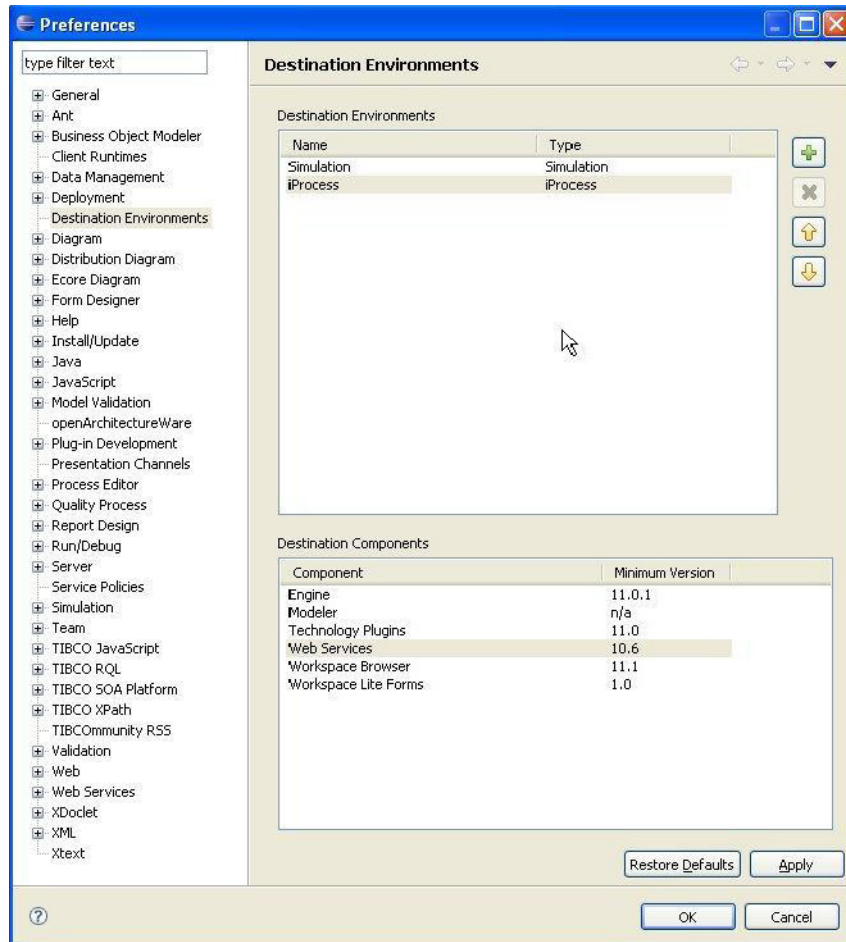
The samples also include processes and TIBCO BusinessWorks projects for the examples found in *TIBCO iProcess™ Web Services Plug-in User's Guide*. For more information about these examples, refer to that guide.

Tutorial 1: Working with Destination Environments


Destination environments are used to specify the runtime environment in which a process is expected to run. TIBCO Business Studio performs validation depending on the destination environment that you select. By default, TIBCO Business Studio provides two destination environments, iProcess and Simulation. A destination environment can also have destination components that specify the component software that makes up a destination environment. You can also create your own custom destination environments and share them with others as shown in this tutorial.

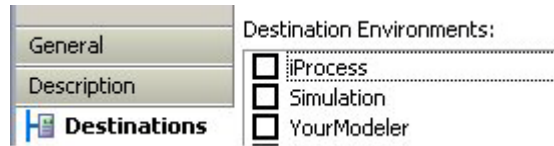
Task A Create a Destination Environment

1. In the Project Explorer, select a process.
2. In the Properties view, click the **Destinations** tab.
3. Select **Window > Preferences**, and highlight **Destination Environments**:
4. Selecting the iProcess destination environment and iProcess from the dropdown menu for Type, shows the destination components and minimum versions of the engine against which the process is supported/validated.



To change the minimum version of a component to use, click on the dropdown from a version number, and select from those available.

5. Click  to add a new destination environment.
6. Click the newly added destination environment and **YourModeler** as the name for it.
7. In the **Type** column, select iProcess. The iProcess destination components are displayed.
8. Select **YourModeler** and click **OK**.
9. Return to the Properties view for a process, and note that the newly created **YourModeler** destination environment now appears on the **Destinations** tab:

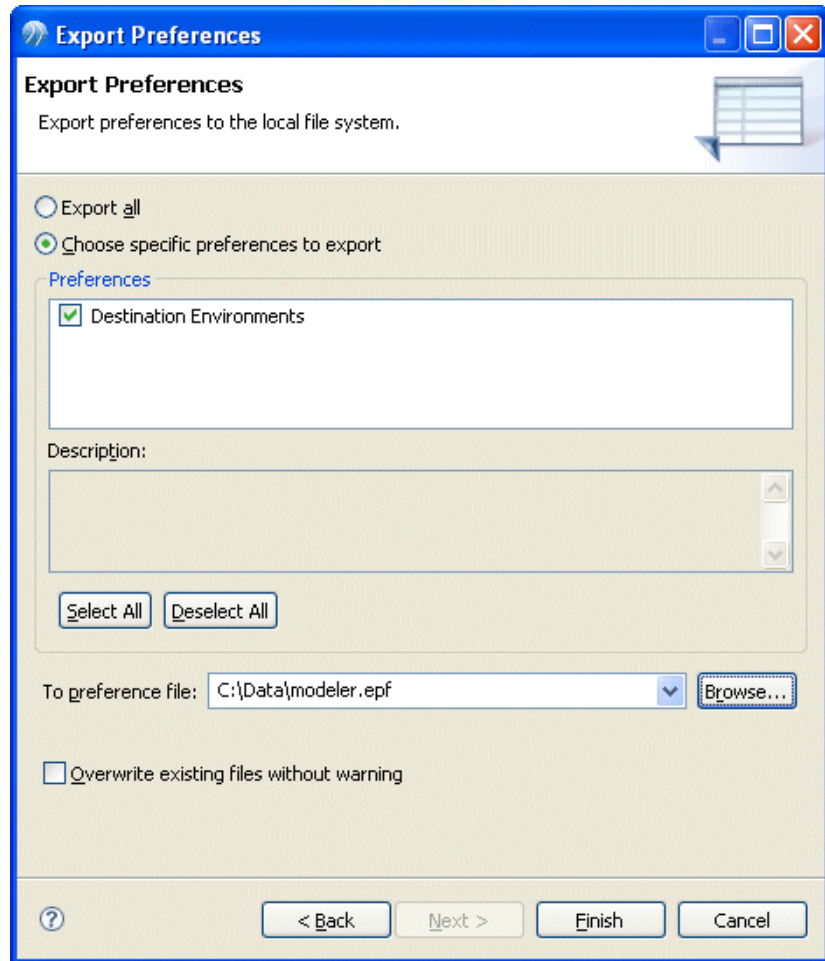


Task B Export Destination Preferences

This section describes how to export your destination preferences to a file so they can be shared by others.

1. Select **File > Export > General**, select **Preferences**, and click **Next**.
2. In the **Export Preferences** dialog, select **Choose specific preferences to export** and **Destination Environments**.

3. Click **Browse** and enter a filename for the export file:



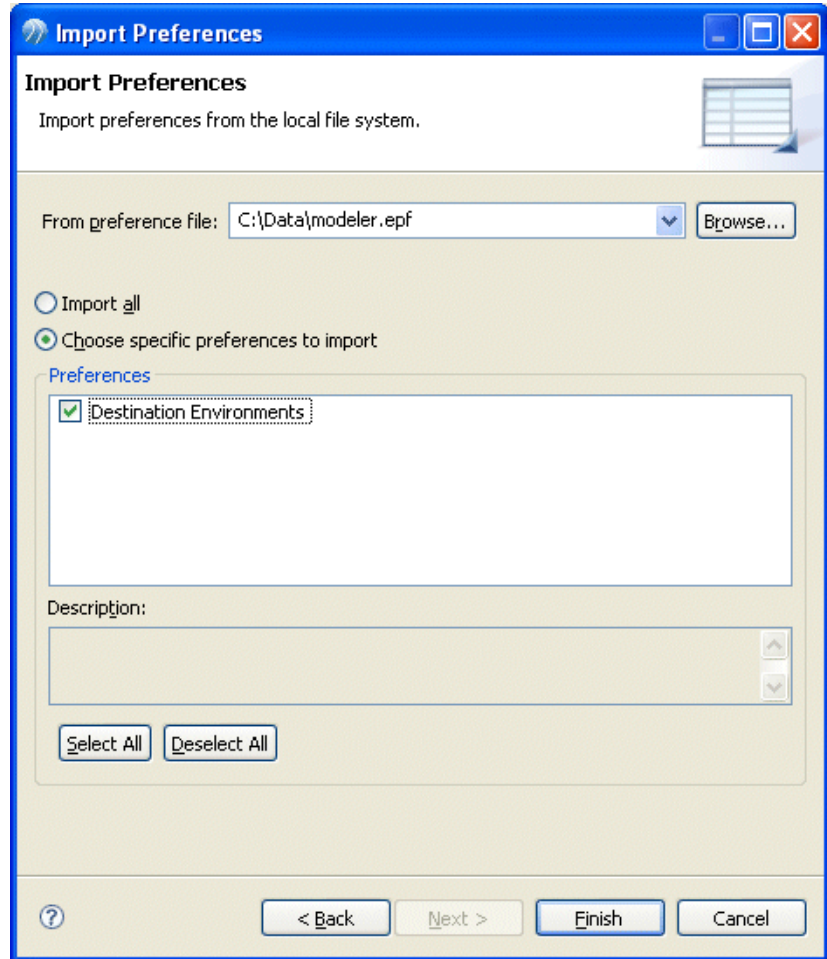
4. Click **Finish** to complete the export.

Task C Import Destination Preferences

This section describes how to import destination preferences from a file.

1. Switch to the workspace into which you want to import the preferences by selecting **File > Switch Workspace** selecting a workspace, and restarting TIBCO Business Studio.
2. Select **File > Import > General**, select **Preferences**, and click **Next**.

3. In the **Import Preferences** dialog, click **Browse** to locate the saved preferences file:



4. Select **Choose specific preferences to import** and **Destination Environments**.
5. Click **Finish** to complete the import.
6. Return to the Properties view for a process, and note that the newly imported Modeler destination environment now appears on the **Destinations** tab.

Tutorial 2: Elaborating a Process: User Tasks

To learn how to implement user tasks in a process, follow this tutorial.

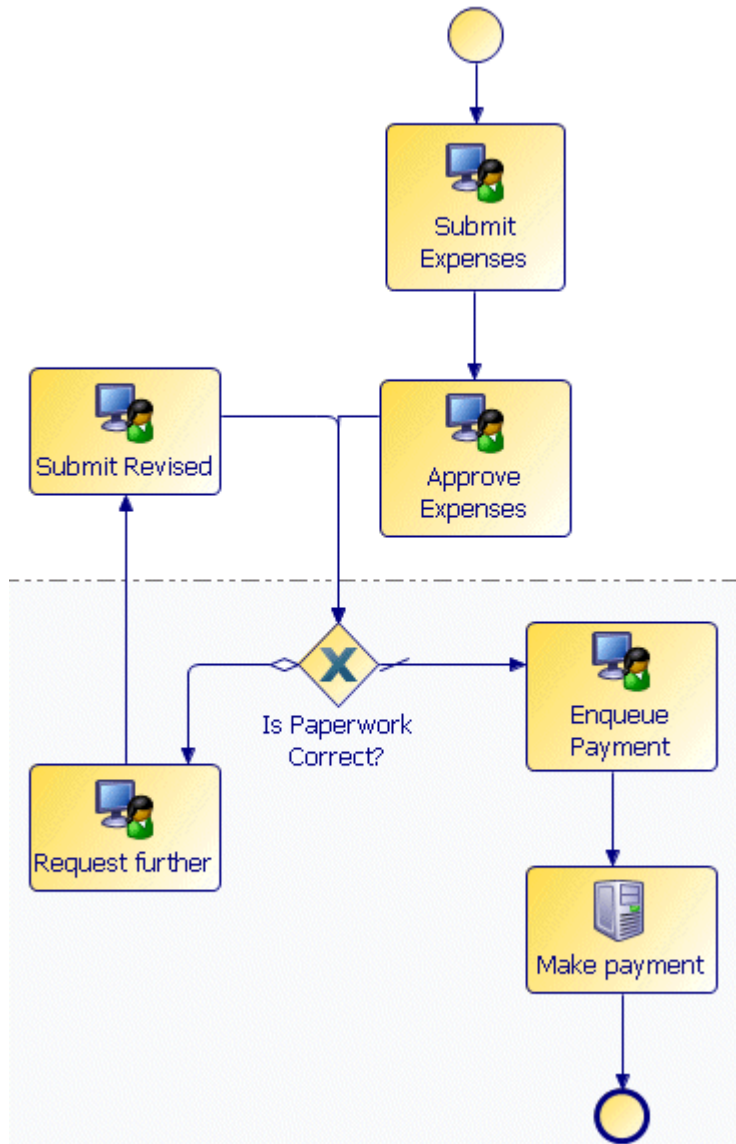
Task A Open the Sample Process

1. If you have not already done so, switch to the Solution Design capability using the following menu.



2. Install the Basic Samples from the Welcome page (using the procedure described in the previous section).

3. Open the Submit Expenses process:



Many of the tasks in this process require input and output. For example, before the expenses can be approved, the performer of this action needs basic information such as the employee name, cost center, expense amount, and so on. The information that is internal to the process will be represented as data fields and information required outside the process will be represented by parameters. In practice, these data fields and parameters may have already been added by the Business Analyst.



Adding parameters to the process means that when it is exported to the iProcess Modeler it will be a sub-procedure.

The data fields and parameters that are created in this section will be exported to fields and I/O parameters in iProcess.

Task B Create Parameters/Data Fields

Create the data fields and parameters that will be used by the process as follows:

1. In the Project Explorer, expand the Submit Expenses process, right-click **Data Fields** and select **New > Data Field**.




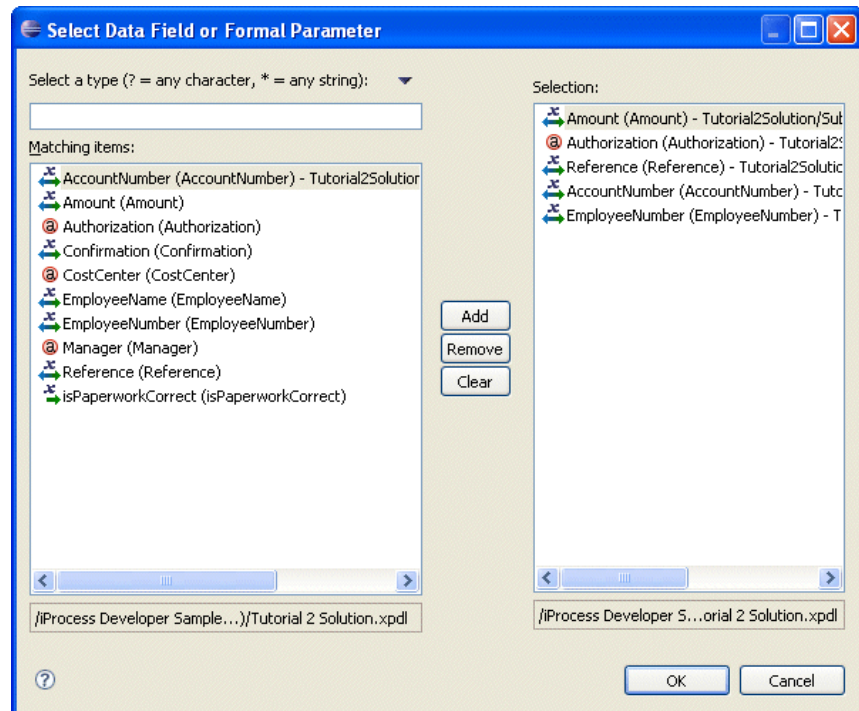
When creating data fields, events, parameters, tasks, and other objects with the Solution Design capability enabled, you can enter both a **Label** and a **Name**. The **Label** is displayed on the process diagram and may have been created by the business analyst. The **Name** is used by the solution engineer (for example, in scripts and when referring to process data in a sub-process). For more information, see [Labels and Names on page 4](#).

2. Create the following data fields:
 - Manager (Text)
 - CostCenter (Text)
 - Authorization (Text)
3. Right-click **Parameters** and select **New > Parameter**.

4. Create the following parameters:
 - EmployeeName (Text)
 - EmployeeNumber (Integer Number)
 - Reference (Text)
 - Amount (Decimal Number)
 - AccountNumber (Integer Number)
 - Confirmation (Text)

Task C Specify Input and Output Parameters to the Tasks

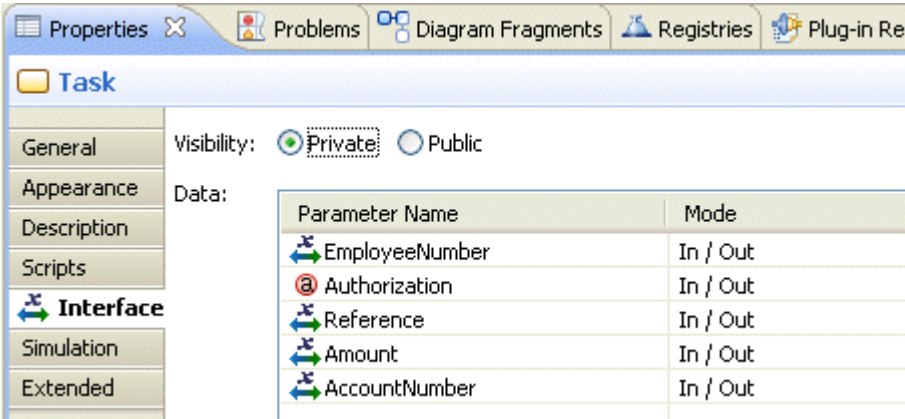
1. Click the **Enqueue Payment** event. In the Properties view, click the **Interface** tab.
2. Click the  button to display the following dialog:



While pressing the Ctrl key, highlight **AccountNumber**, **Amount**, **Authorization**, **EmployeeNumber**, and **Reference**, then click OK



- The wildcard ? returns all matching data fields or formal parameters. Use the * wildcard to restrict the results (for example, *2 to return all data fields or parameters ending in 2). Note that the wildcard * by itself does not return any results; it only works in conjunction with a string.
 - To select several parameters in the **Select Data Field or Formal Parameter** dialog, press the **Ctrl** key and click the desired data field or parameter.
 - Data fields or parameters with spaces in their names cannot be used for mapping (for example, mapping to parameters in a web service).
3. The parameters you selected are displayed as In/Out parameters:



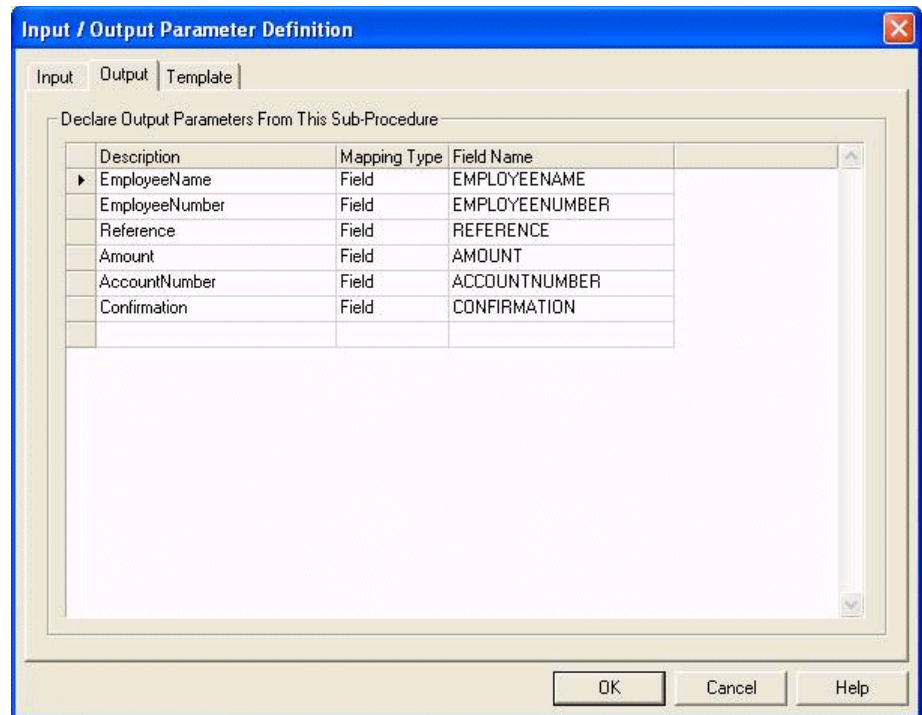
Inbound and outbound parameters are from the perspective of the *form*, not the *user*. This means that inbound parameters are sent to the form by the process, not the user. Outbound parameters are sent to the form by the user.

Upon deployment to the iProcess Engine, inbound parameters (parameters with a mode of **In**) become display fields (the mandatory flag is ignored). Outbound parameters (mode **Out**) or **In/Out** parameters become either optional or mandatory fields, depending on the setting of the Mandatory flag. For more information about using fields in forms, see *TIBCO iProcess Modeler - Basic Design*.

4. Install the iProcess Developer Samples from the Welcome page to check your process against the solutions.

Summary

You can create data fields and parameters and specify their input and output mappings on user tasks. Upon import or deployment to the iProcess Modeler, the data fields become fields and the parameters become I/O parameters and fields. For example:



For more information about deployment, see [Tutorial 7: Deploying a Process on page 54](#) and [Deploying to the iProcess Engine on page 219](#).

Tutorial 3: Automatically Creating Forms

This tutorial shows how to automatically create forms from a user task.

Task A Create a Forms Special Folder



A Forms special folder is created automatically if you create a TIBCO Business Studio BPM Developer Project.

1. Select the **iProcess Developer Solutions** project folder, right-click and select **New > Folder**.
2. Name the folder **Forms**, and click **Finish**.
3. Right-click the newly created **Forms** folder and select **Special Folders > Other > Use as Forms Folder**.
4. In the **Project Asset Configuration** dialog, click **Finish**.
5. The icon of the **Forms** folder changes to indicate that the folder has been designated as a special folder for forms.

Task B Generate the Forms

1. Right-click the **Enqueue Payment** user task and select **Form > Generate**.
2. In the **New User Activity Form** dialog, review the parameters associated with the form and click **Next**.
3. Review the path and name of the form, and click **Finish**.
4. The newly created form opens in the Form Designer, and the **Form** field in the Properties view of the user task is completed automatically.

Tutorial 4: Elaborating a Process: Service Tasks

This tutorial describes how to work with WSDL files in TIBCO Business Studio, use a WSDL file in a services task, and how to map the input and output parameters.

Calling Web Services

For the purposes of this tutorial, continue use the Submit Expenses process from the Basic Samples project. Install the iProcess Developer Samples project if you have not already done so. This project contains the WSDL file used in this tutorial.

Task A Create the Services Descriptors Special Folder

1. Right-click the Basic Samples project and select **New > Folder**.
2. In the **New Folder** dialog, enter **Service Descriptors** as the folder name and click **Finish**.
3. In the Project Explorer, right-click the newly-created **Service Descriptors** folder, and select **Special Folders > Use as Service Descriptors Folder**.

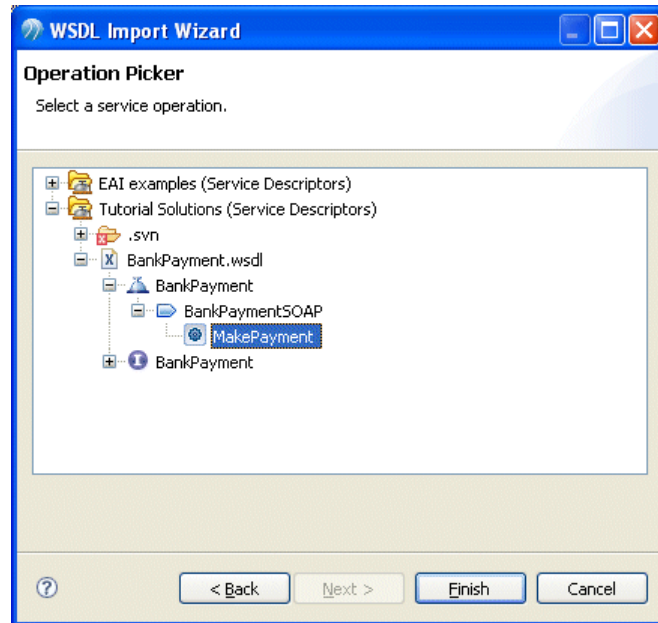
Task B Import the WSDL File and Select the Operation

1. If you have not already done so, switch to the Solution Design capability using the following menu.



2. In the Project Explorer, open the project that contains the Submit Expenses process.
3. Click the **Make Payment** task.
4. In the Properties view, select **Web Service** from the **Service Type** drop-down list.
5. Click the **Import WSDL** button.
6. Select **Import from a File** and click **Next**.
7. Browse to find **BankPayment.wsdl** in the workspace and click **Next**. This WSDL file is located in the iProcess Developer Samples project.

8. Browse to specify the **Service Descriptors** folder where you want to store the WSDL file and click **Next**.
9. Expand the **BankPayment** service and select the **MakePayment** operation, and click **Finish**:



- This populates the rest of the web services fields such as **Port Name**, **Operation**, and **Transport**.

Service Type: Web Service

Operation: Select Clear Import WSDL Generate

Port Type: BankPayment

Operation Name: MakePayment

Port Name: BankPaymentSOAP

Service Name: BankPayment

Transport: SOAP over HTTP

Endpoint Resolution:

WSDL: ☐ Use local: ☒ Use remote:

Location: This is taken from the Alias URL at Run-time

Endpoint Name: Payroll System ... Clear

Security Profile:

- Save the project.

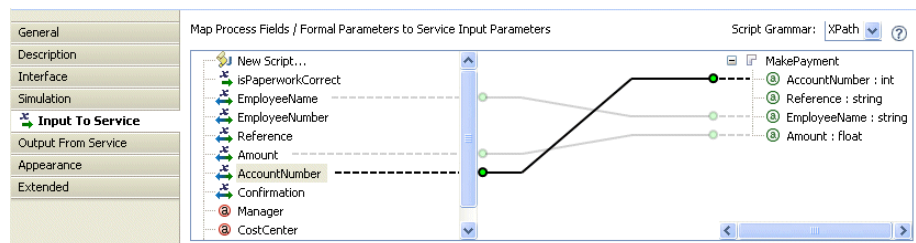
Task C Map the Input/Output Parameters

- Click the **Input To Service** tab. On the left of the tab are the parameters and data fields and on the right are the formal parameters that the **MakePayment** service expects.

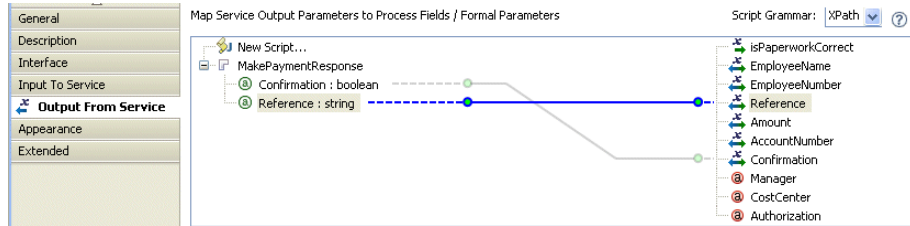


For more information about the mapping parameters and applying scripts to mappings, see [Using the Mapper on page 149](#).

- Expand the **MakePayment** service. Drag the pointer from the **EmployeeName** formal parameter to the **EmployeeName** actual parameter in the web service. Complete the input parameter mapping as follows:



- Click the **Output From Service** tab. Using the same method as on the **Input To Service** tab, complete the output parameter mapping as follows.

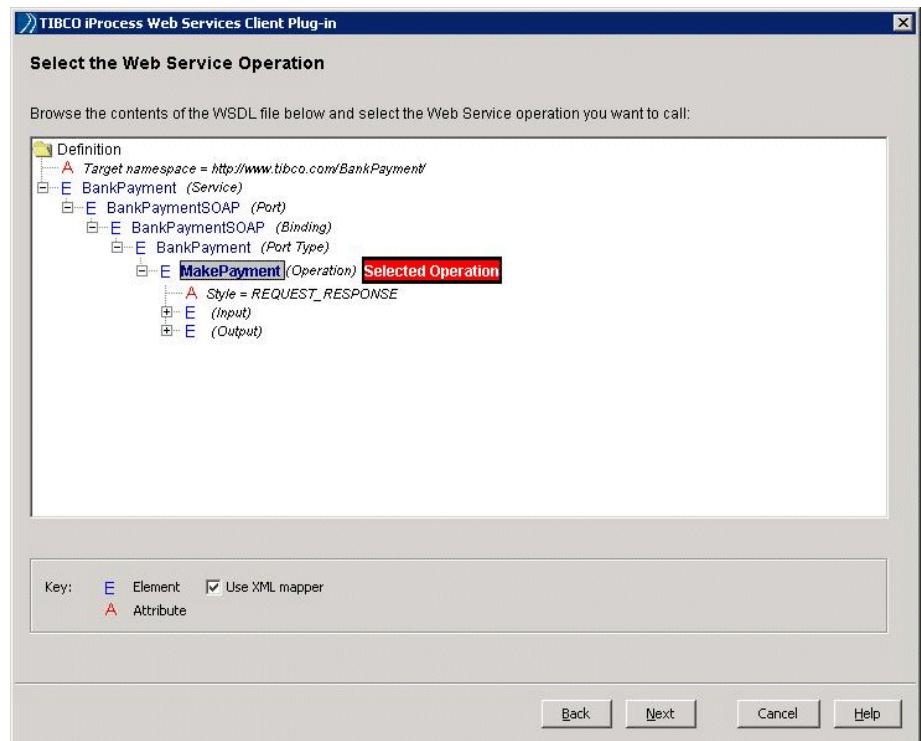


You can see that the web service returns **Confirmation** and **Reference** parameters. In this case the **Confirmation** parameter is Boolean and indicates whether the transaction completed.

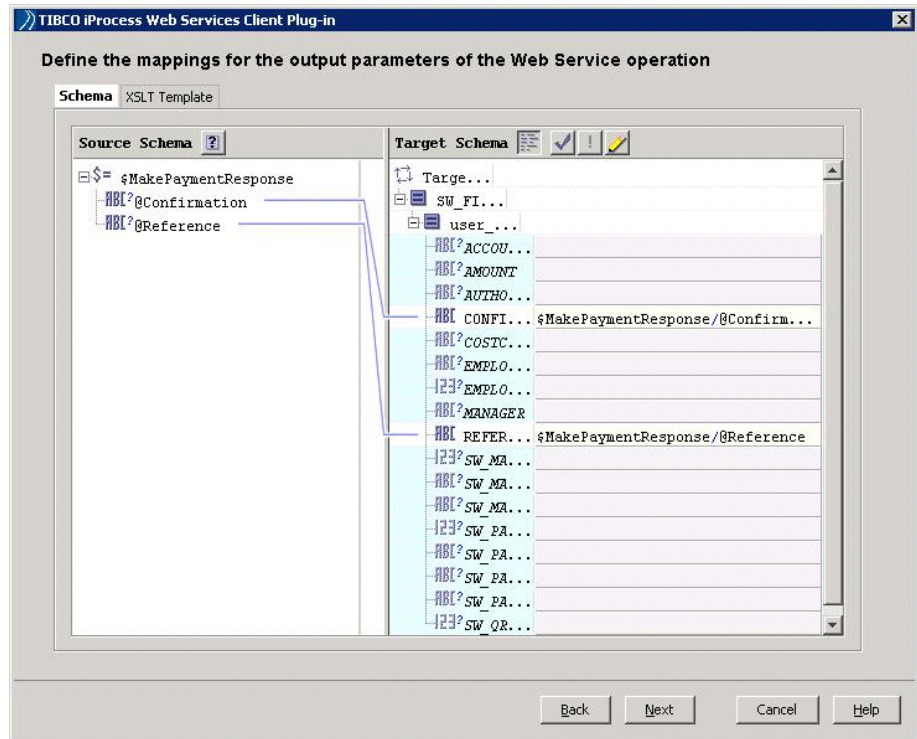
- Save the package.

Summary

This tutorial showed how to add a WSDL file to a project and then associate that WSDL file with a service task, mapping the input and output parameters. Upon import or deployment to the iProcess Modeler, the service task becomes a Web Services EAI step:



You can see that the parameter mappings are configured in the EAI step as well:



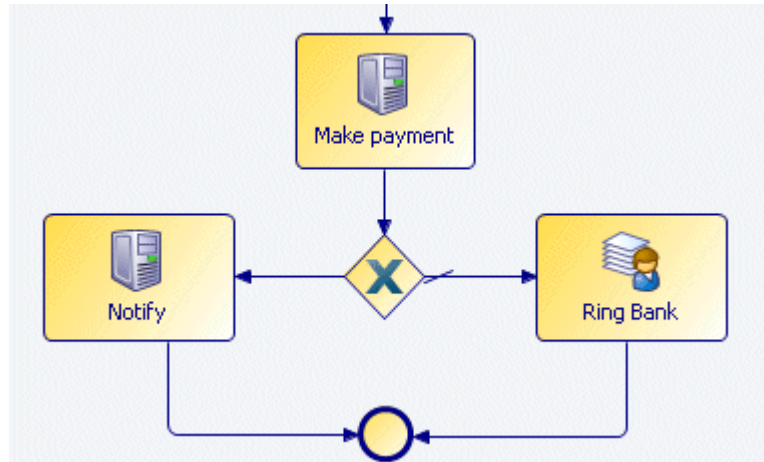
For more information about deployment, see [Tutorial 7: Deploying a Process on page 54](#) and [Deploying to the iProcess Engine on page 219](#).

Creating an E-Mail Task

In the previous section, we added a call to a web service from the **Make a Payment** activity. This included a Confirmation parameter which indicated whether the payment was successful. In this section, we will add an email task to notify the person who submitted the claim when the payment is made. We will also add a manual task in the event that the payment fails.

1. Insert a gateway after the Make a Payment activity.

- After the gateway, add a service task called **Notify** and a manual task called **Ring Bank**. It should look like this:



- Select the **Notify** service task, then on the **General** tab of the Properties view for the service task, select the **E-Mail** option from the **Service Type** drop-down list:

Service Type: E-Mail ▼

To: %Manager%

Subject:

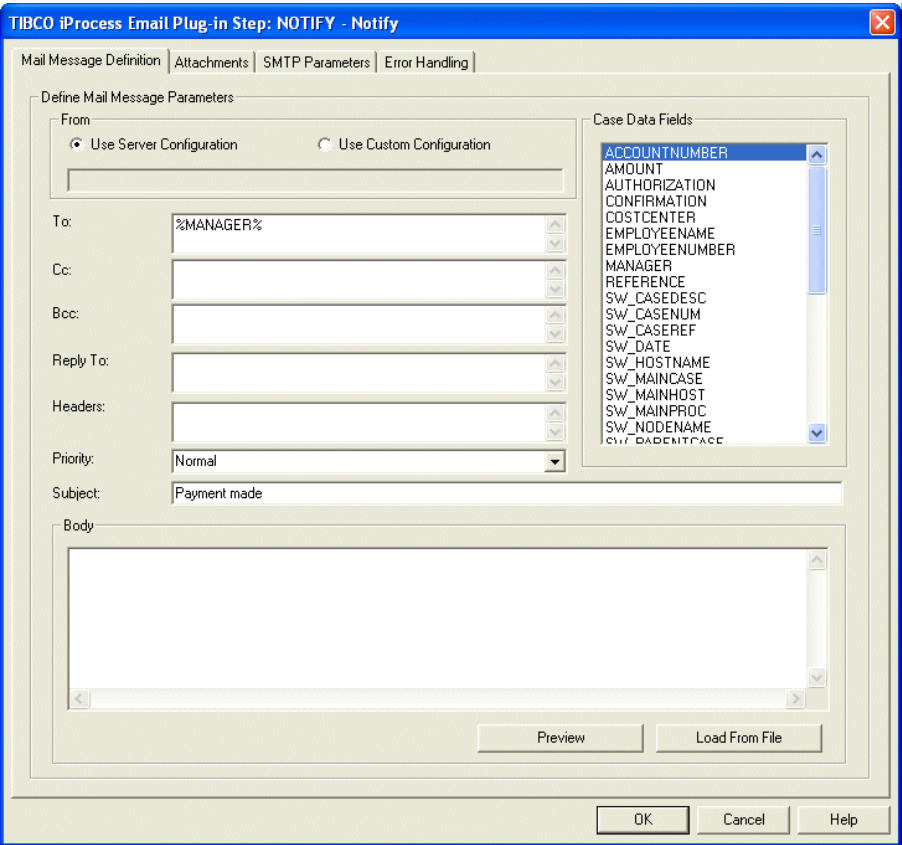
Body:

[More Details...](#)

- Enter an email address for the recipient in the **To:** field, a subject and the body for the message. This is the minimum configuration necessary to send an email message. For further options, click **More Details** or the **E-Mail** tab and continue to specify further parameters.

Summary

You can configure a service task so that it sends an e-mail at runtime. Upon import or deployment to the iProcess Modeler, the service task becomes an E-mail EAI step. For example:



For more information about deployment, see [Tutorial 7: Deploying a Process on page 54](#) and [Deploying to the iProcess Engine on page 219](#).

Tutorial 5: Adding a Script Task

The process in this tutorial represents the approval process for a loan. First the applicant's details are taken (a user task). Then a script task calculates the allowed amount of the loan, whether manager approval is needed, and so on.

Prerequisites

1. Create a process called loanApproval.
2. On the **Destinations** tab, select **iProcess**.
3. Define the following data fields at the process level.

Data Field	Details
allowedLoan	Decimal Number, Length 10, Decimal Places 2
custName	Text, Length 50
dayOfMonth	Integer, Length 10
decisionDt	Date Time
lapseTime	Date Time
loanAmt	Decimal Number, Length 10, Decimal Places 2
loanApproved	Boolean
loanDifference	Decimal Number, Length 10, Decimal Places 2
mgrApprovalNeeded	Boolean
monthNum	Integer, Length 10
percentageDiff	Decimal Number, Length 10, Decimal Places 2
personalLoan	Decimal Number, Length 10, Decimal Places 2
propertyValue	Decimal Number, Length 10, Decimal Places 2
rmks	Text, Length 50
salary	Decimal Number, Length 10, Decimal Places 2
salaryMultiple	Integer, Length 10

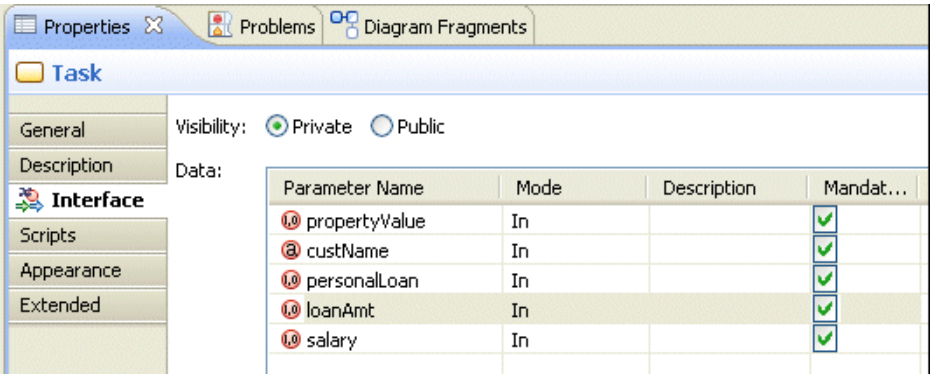
Data Field	Details
salesAdvisor	Text, Length 50

- 4. Define the **ipadmin** participant that is of the Participant Type **ROLE**.

Gather User Details and Apply Core Logic

Define the first two steps in the process as follows:

- 1. Define a user task called **Enter User Details** with **ipadmin** as the participant. At runtime, this step will display a form that prompts the user to enter the applicant's details.
- 2. Assign the following data fields as inbound parameters:



- 3. Add a script task called **coreLogic**. The calculations in the script determine how the subsequent flow proceeds.
- 4. On the **General** tab, select **JavaScript** from the **ScriptDefinedAs** drop-down list.
- 5. Maximize the Properties view (either double-click its tab or choose **Maximize** from the tab's popup menu).
- 6. Next, enter the following lines to determine the allowed loan and loan difference. These calculations use some of the parameters that will be entered on the form at runtime (**loanAmt**, **personalLoan**, and **salary**).

```
allowedLoan = salary * 5;
allowedLoan = allowedLoan - personalLoan;
loanDifference = loanAmt - allowedLoan;
```

- 7. Next add the logic that determines whether a manager needs to approve the loan and whether the loan itself is approved:

```
mgrApprovalNeeded = false;
```



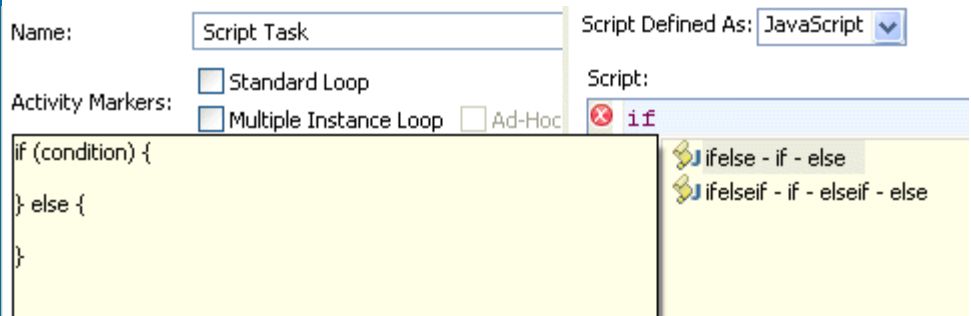
```

if(loanDifference > 0){
    percentageDiff = loanDifference / allowedLoan * 100;
    if(percentageDiff < 10.00){
        mgrApprovalNeeded = true;
        loanApproved = false;
    }else{
        // loan rejected
        loanApproved = false;
    }
}
}else{
    // loan approved
    loanApproved = true;
}
decisionDt.Date = IPESystemValues.SW_DATE;
decisionDt.Time = IPESystemValues.SW_TIME;

```

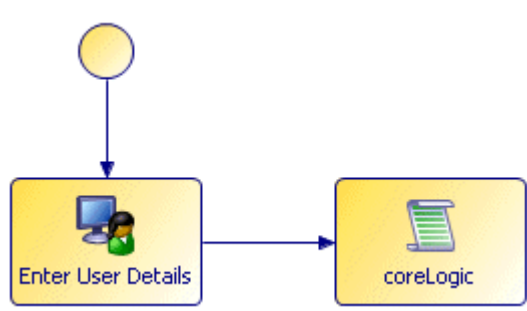


Content assist is available for process data that you define (for example, data fields), and there are also templates for common JavaScript constructs. For example, if you enter `if`, then press Ctrl+Space, you can use the following template to construct an if else:



Similarly, if entering the data field **allowedLoan**, simply enter **a**, press Ctrl+Space and in the content assist window, double-click **allowedLoan** to insert it in the script.

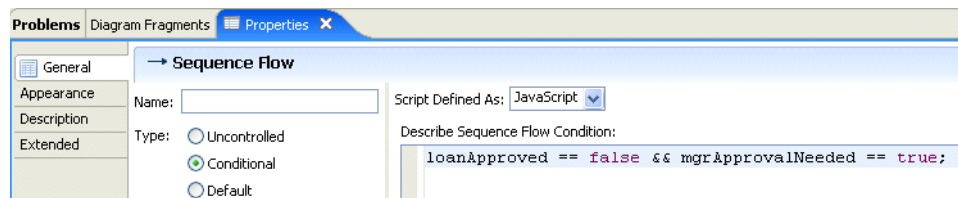
8. Add sequence to connect the start event and the two tasks. The process should look similar to this:



Add Branching

In this section we will add gateways to control the branching of the process.

1. Add an XOR data-based gateway to the process to the right of the **coreLogic** task.
2. Above the gateway, add a script task called **dateCheck**.
3. Connect the gateway to the new script task and to the **coreLogic** step with sequence flow.
4. In the Properties view for the sequence flow to the new script task, change the Sequence Flow type to **Conditional**, and select **JavaScript** from the **ScriptDefinedAs** drop-down list.
5. Describe the Sequence Flow Condition as follows:
`loanApproved == false && mgrApprovalNeeded == true;`
6. The **General** tab of the Properties view should look like this:



7. From the **dateCheck** task, add a user task called **MgrApproval** with **ipeadmin** as the participant, and after that an end event:



Call iProcess Functions

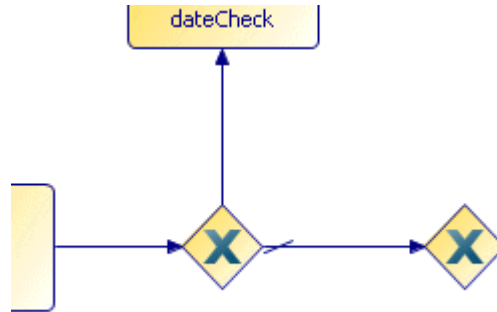
This section describes how to add iProcess functions to a script.

1. Select the **dateCheck** task, and on the **General** tab, select **JavaScript** from the **ScriptDefinedAs** drop-down list.
2. Add the following script which uses iProcess Engine Date and Time functions:

```
dayOfMonth = IPEDateTimeUtil.DAYNUM(IPESystemValues.SW_DATE);
monthNum = IPEDateTimeUtil.MONTHNUM(IPESystemValues.SW_DATE);
if(dayOfMonth == 30 && monthNum == 12){
    // do not approve the loan on 30th of December
    loanApproved = false;
```

```
rmks = "30th of December, no loan Approval";
}
```

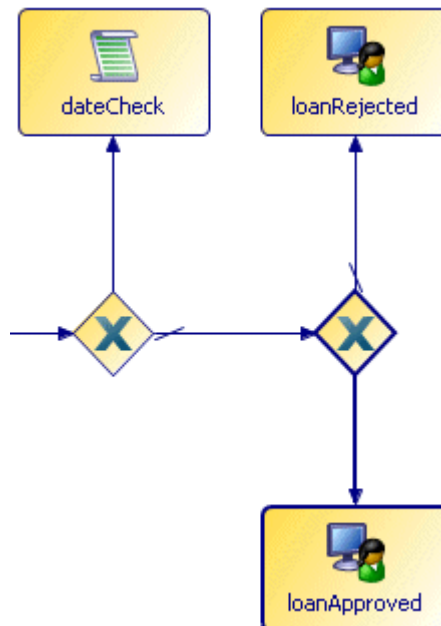
3. Add another XOR data-based gateway to the process as shown and connect the two gateways with a default sequence flow:



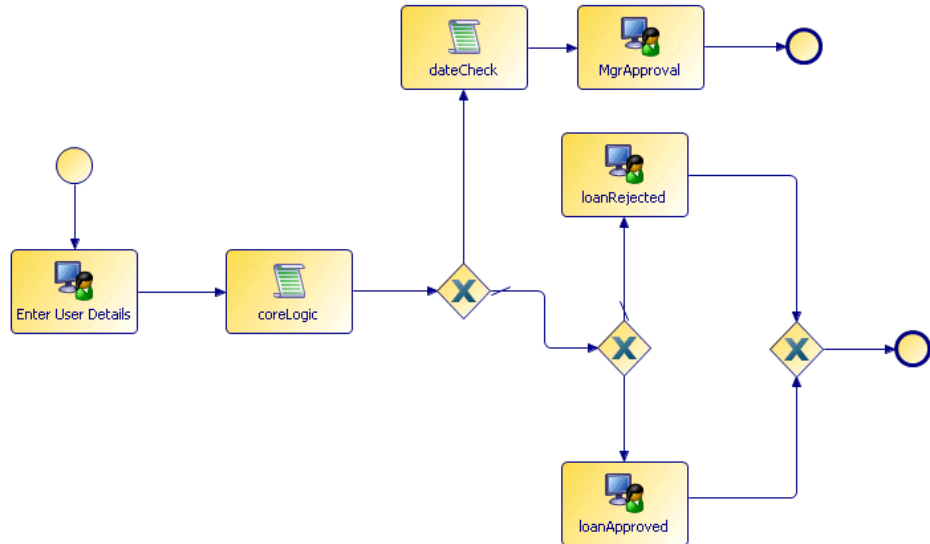
4. Add two new user tasks, **loanRejected** and **loanApproved** each with **ipeadmin** as the participant.
5. Connect the user tasks with the new gateway, by adding a default sequence flow to the **loanRejected** task, and a conditional sequence flow to the **loanApproved** task. Define the condition on the sequence flow as follows:

```
loanApproved == true;
```

This part of the process should look like this:



6. Add another XOR data-based gateway, an end event, and sequence flow as follows:



Tutorial 6: Creating Dynamic Sub-Processes

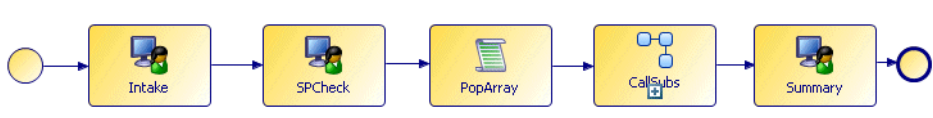
Dynamic sub-processes are used when a number of sub-processes could be called upon, but it is not known at design time which sub-process will be called. The exact sub-process to be run is chosen at runtime, depending on the process data. In TIBCO Business Studio, this is accomplished by using a process interface.

This tutorial shows how to use process interfaces. In this process:

1. A nurse enters some basic information about the patient such as name, address and the medical conditions they have.
2. A supervisor checks the details and makes a choice as to what else needs to be done as part of the patient's intake. They may require an X-ray or the booking of a bed.
3. As a result of the supervisor's choice, a work item is sent to the required departments (for example, the X-ray department) and someone in that department books an appointment. An email is sent with the appointment details.
4. Finally, a form displays a summary of the patient's admission information along with any new booking information that has been entered.

Looking at the Example Process

1. Create a process called ConsultDoctor that looks like this:



2. Create the following data fields for the ConsultDoctor process:

Data Field	Details
Address1stline	Text, Length 50
ApptTimeArr	Array of Date Times
BloodTestAppointment	Date Time
BloodTestComments	Text, Length 50
Complaint	Text, Length 50

Data Field	Details
DeptArr	Array of Text, Length 10
FName	Text, Length 50
Gender	Text, Length 6
LName	Text, Length 50
loopIndex	Integer, Length 10
NeedBloodTest	Boolean
NeedXRay	Boolean
PostCode	Text, Length 10
ProcessIdentifierAr	Array of Text, Length 50
testCommentsArr	Array of Text, Length 50
XRayAppointment	Date Time
XRayComments	Text, Length 50

3. Create a process interface called **BookAppointment** that specifies the start event and its input/output parameters as follows:

The screenshot shows the 'Process Interface Editor' window with the 'BookAppointment' interface selected. The interface is configured with the following details:

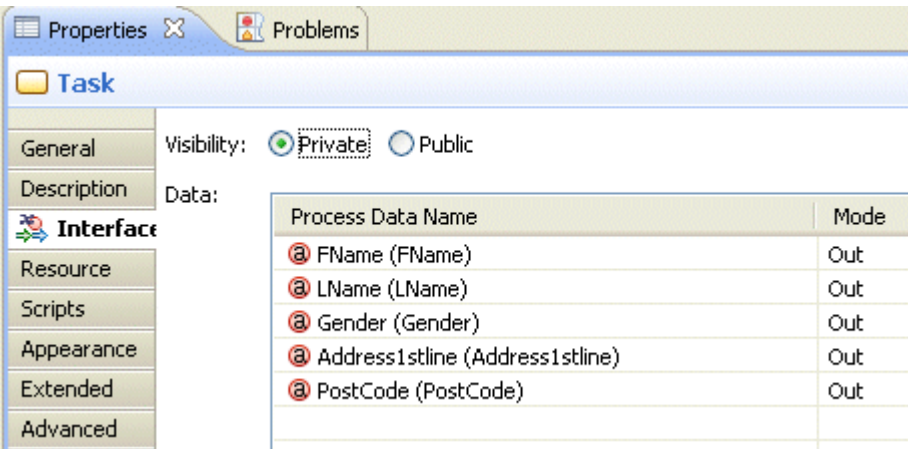
- General Information:**
 - Name: BookAppointment
 - Description: (Empty text area)
- Formal Parameters:**
 - Specify the formal parameters used by events in this interface.
 - Parameters listed: FName (FName), LName (LName), Adress1stLine (Adress1stLine), Gender (Gender), PostCode (PostCode), ApptTime (ApptTime), testComments (testComments), and Dept (Dept).
 - Buttons: New... and Remove.
- Start Events:**
 - Specify events that can be used to start processes that implement this interface.
 - Event listed: Start Event (Interface Parameters).
 - Buttons: New... and Remove.
- Intermediate Events:**
 - Specify in-flow events that can be triggered in processes that implement this interface.
 - Buttons: New... and Remove.

The status bar at the bottom indicates 'Process Interface Editor'.

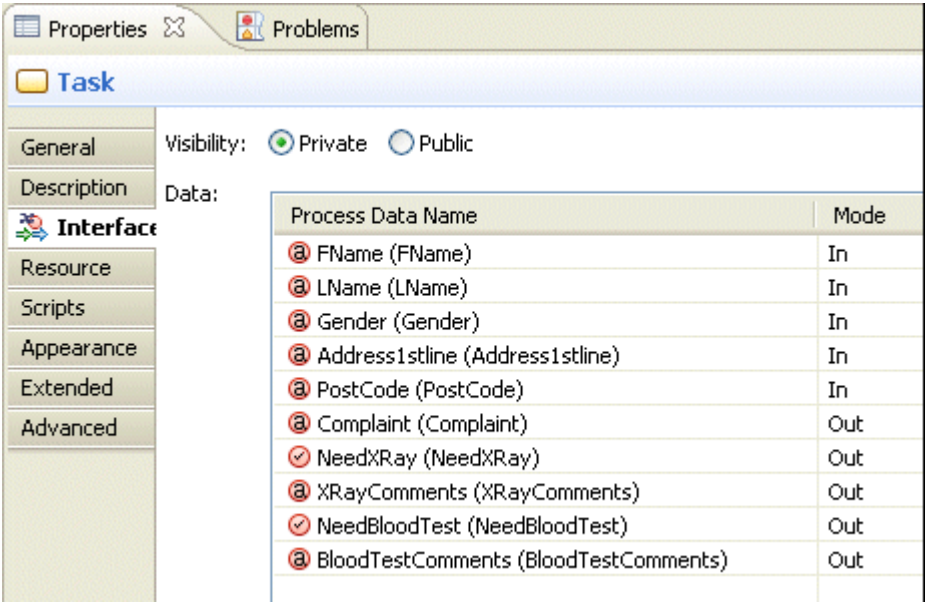
All of the parameters should be mandatory, and all specified with a mode of **In**, except **ApptTime** which should be **Out**, and **Dept** which should be **In/Out**.

Right-click the **BookAppointment** process interface and **New > Process**, and create two processes based on the **BookAppointment** process interface: **BloodTestAppointment** and **XRayAppointment**. Note that because these processes were based on the process interface, they are created with exactly the same events and parameters as the interface.

4. Add the parameters to the **Intake** user tasks as follows:



5. Add the parameters to the **SPCheck** user task as follows:



6. Add the parameters to the **Summary** user task as follows:

Properties Problems

Task

General

Description

Interface

Resource

Scripts

Appearance

Extended

Advanced

Visibility: ☒ Private ☐ Public

Data:

Process Data Name	Mode
@ FName (FName)	In
@ LName (LName)	In
@ Gender (Gender)	In
@ Address1stline (Address1stline)	In
@ PostCode (PostCode)	In
@ Complaint (Complaint)	In
<input checked="" type="checkbox"/> NeedXRay (NeedXRay)	In
<input type="checkbox"/> XRayAppointment (XRayAppointment)	In
<input checked="" type="checkbox"/> NeedBloodTest (NeedBloodTest)	In
<input type="checkbox"/> BloodTestAppointment (BloodTestAppointment)	In

7. Create the participant **sw_starter** and assign it to the user tasks.
8. The script task **PopArray** is where the logic occurs to set the exact sub-process that will be selected at runtime.



When you use process names in TIBCO Business Studio scripts, use the method **iPEProcessNameUtil.GETPROCESSNAME** to convert valid TIBCO Business Studio process names into equivalent iProcess procedure names. For example, long process names are truncated as they would be upon deployment to iProcess. This ensures that the sub-process call task works as expected upon deployment. The method takes a string and returns a string. Similarly, use the **iPETaskNameUtil** method when referring to task names.

Add the following JavaScript to the script task:

```
if(NeedXRay){
    testCommentsArr[0] = XRayComments;
    DeptArr[0] = "XRAY";
    ProcessIdentifierArr[0] =
    IPEProcessNameUtil.GETPROCESSNAME("XRayAppointment");
    if(NeedBloodTest){
        testCommentsArr[1] = BloodTestComments;
        DeptArr[1] = "BLOODTEST";
        ProcessIdentifierArr[1] =
        IPEProcessNameUtil.GETPROCESSNAME("BloodTestAppointment");
    }
    return;
}else {
```



```

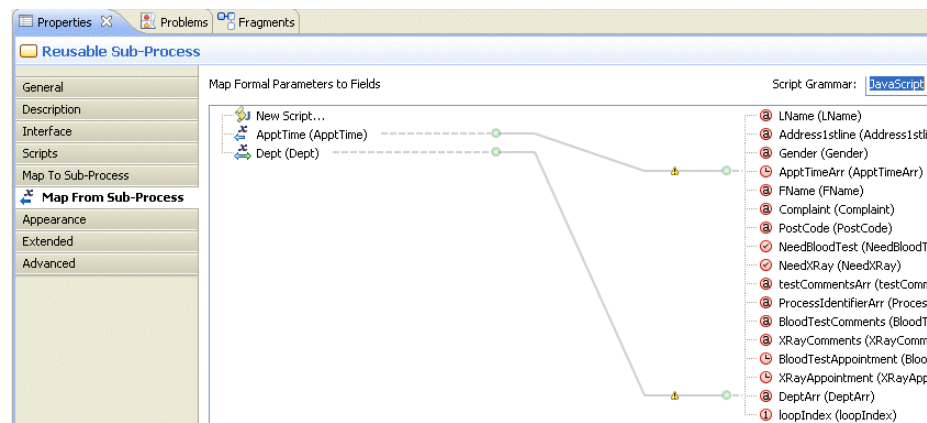
testCommentsArr[0] = BloodTestComments;
DeptArr[0] = "BLOODTEST";
ProcessIdentifierArr[0] =
IPEProcessNameUtil.GETPROCESSNAME("BloodTestAppointment");
}

```

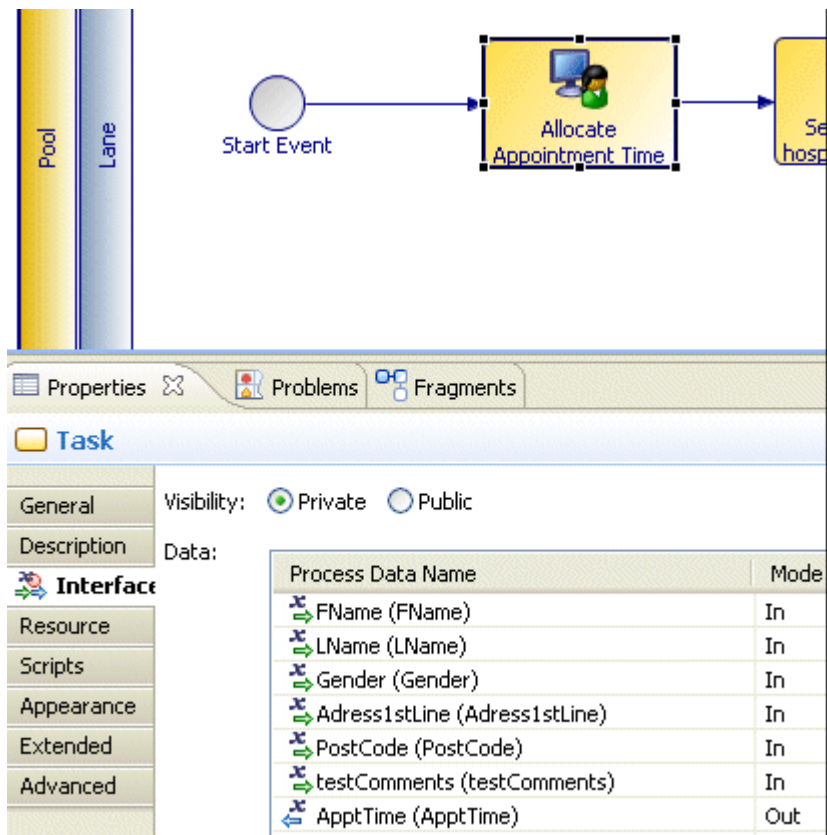
- In the reusable sub-process task, select the **BookAppointment** process interface, and set the runtime identifier to the **ProcessIdentifierArr** array data field (that holds the list of potential sub-processes to call).

- Map the input to the interface as follows:

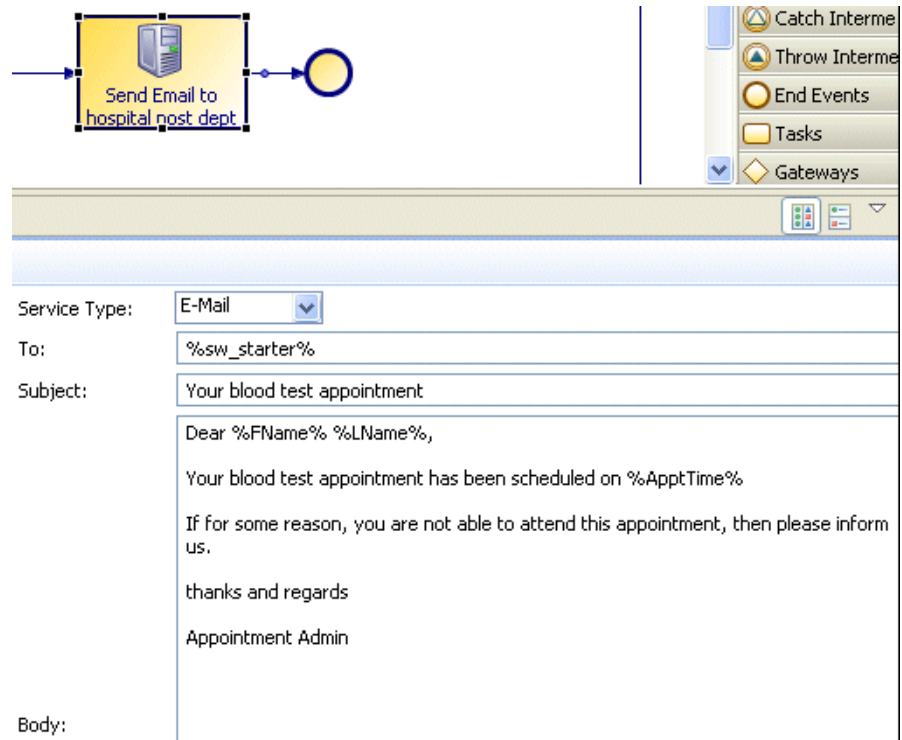
11. Map the output from the sub-process as follows:



12. Configure the **BloodTestAppointment** and **XRayAppointment** subprocesses. The user task takes the patient details and outputs an appointment time:



13. Configure the email task as follows:



The screenshot shows the iProcess configuration interface. At the top, a process diagram shows a yellow task box labeled "Send Email to hospital post dept" connected to a blue circle. To the right is a palette with icons for "Catch Intermediate", "Throw Intermediate", "End Events", "Tasks", and "Gateways". Below the diagram is the configuration form for the "Send Email" task.

Service Type:

To:

Subject:

Body:

```

Dear %FName% %LName%,

Your blood test appointment has been scheduled on %ApptTime%

If for some reason, you are not able to attend this appointment, then please inform us.

thanks and regards

Appointment Admin
  
```

The recipient of the email task is set to **sw_starter** for testing purposes, but this could be set to a different participant, or the email address of the patient by modifying the process.

14. Returning to the main process, the script task preceding the final user task sets up the final user task which displays a summary of the patient and appointment information. Add the following Java script:

```

loopIndex = IPEArrayUtil.FINDARRELEMENT("DeptArr",-1,"XRAY");
if(loopIndex!=-1){
    XRayAppointment.Date = ApptTimeArr[loopIndex].Date;
    XRayAppointment.Time = ApptTimeArr[loopIndex].Time;
}
loopIndex = IPEArrayUtil.FINDARRELEMENT("DeptArr",-1,"BLOODTEST");
if(loopIndex!=-1){
    BloodTestAppointment.Date = ApptTimeArr[loopIndex].Date;
    BloodTestAppointment.Time = ApptTimeArr[loopIndex].Time;
}
  
```

Example Case

Running this example in iProcess proceeds as follows:

- 1. Patients details are entered in the initial form.

The screenshot shows a web browser window with the address bar displaying 'http://grp-ron.emea.tibco.com:8080 - Start Case: UAT test - CONSULT...'. The main content area is a form titled 'Intake:' with a separator line. The form contains the following fields: 'FName:' with the value 'Joe', 'LName:' with the value 'Murphy', 'Gender:' with a dropdown menu showing 'Male', 'Address1stline:' with the value '112 Abbey Lane', and 'PostCode:' with the value 'SN2 8BL'. Each field has a small question mark icon to its right. At the bottom of the form are three buttons: 'Undo', 'Keep', and 'Release'.

- 2. A diagnosis is made and further appointments are recommended.

The screenshot shows a web browser window titled 'Web Browser'. The main content area is a form titled 'SPCheck:' with a separator line. The form contains the following fields: 'FName:' with the value 'Joe', 'LName:' with the value 'Murphy', 'Gender:' with a dropdown menu showing 'Male', 'Address1stline:' with the value '112 Abbey Lane', 'PostCode:' with the value 'SN2 8BL', 'Complaint:' with the value 'headache', 'NeedXRay:' with a dropdown menu showing '1', 'XRayComments:' with the value 'asap', 'NeedBloodTest:' with a dropdown menu showing '1', and 'BloodTestComments:' with the value 'headache'. Each field has a small question mark icon to its right. At the bottom of the form are three buttons: 'Undo', 'Keep', and 'Release'.

- 3. The appointments are scheduled in forms displayed by the sub-processes (in this case for an X-ray and blood test since it was determined that the patient needed both).

4. Email is sent, and a final form displays a summary.

Web Browser

Summary:

=====

FName:

Joe

LName:

Murphy

Gender:

Male

Address1stline:

112 Abbey Lane

PostCode:

SN2 8BL

Complaint:

headache

NeedXRay:

1

XRayAppointment:

10/28/2008

10:00

NeedBloodTest:

1

BloodTestAppointment:

10/28/2008

09:00

Undo

Keep

Release

Tutorial 7: Deploying a Process

This tutorial describes deploying one of the sample processes to the iProcess Engine. For more information about destination environments and validation, see [Validating Processes on page 213](#).

Prerequisites

- Network access to a running iProcess Engine system where you can deploy the process
- Username and password of an iProcess Engine user with either the PRODEF or ADMIN permission that you can use to connect to the iProcess Engine
- Host name (machine name or IP address)
- The sample process contains a service task that calls a web service. If you do not have the TIBCO iProcess™ Web Services Plug-in installed, you will receive a warning during deployment. However, this warning does not prevent the process from deploying.
- Port number that the iProcess Engine uses for the Java Management Extension (JMX) engine. By default the Port is **10025**. This was configured during installation of the iProcess Engine and stored in the **SWJMXConfig.port** entry in the **SWDIR\etc\swjmx.properties** file. If you cannot determine the port number, contact your iProcess administrator.



You now can run TIBCO Business Studio through a firewall, as long as you configure and open the necessary ports.

By default, TIBCO Business Studio uses port **10025** for the JMX engine and also uses port **10026** as an RMI port. The user will not be able to deploy if either of these ports are closed (which one could be for example if a customer was using a VPN).

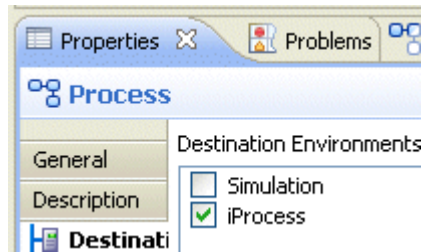
So while **swjmx.properties** can define the JMX port, be aware you need to open that port as well as that port +1 as an RMI port.

If you are unsure of any of these prerequisites, see your iProcess administrator for details.

Task A Install the Sample Process

1. Install the iProcess Developer Samples as described in [iProcess Developer Samples Project on page 18](#).

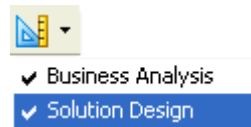
- Expand the **EAI examples (Process Packages)** project and select the **BANKSOAP** process in the Project Explorer. In the Properties view, on the **Destinations** tab, confirm that **iProcess** is selected:



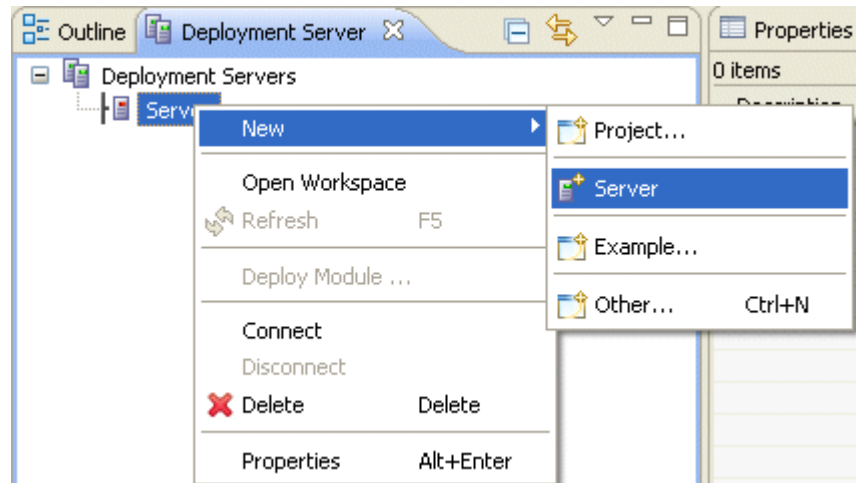
- Confirm that there are no errors or warnings in the Problems view.

Task B Create the Deployment Server

- If you have not already done so, switch to the Solution Design capability using the following menu.



- In the **Deployment** view, right-click **Deployment Servers**, and select **New > Server**.



- Name the server **TestDeploy**.
- Accept **iProcess Engine Server** as the **Runtime** and click **Next**.

5. Enter the runtime server parameters.



The following list does not contain all parameters displayed in the dialog, however you should be able to accept the default values for most parameters. For more information see [Synchronizing your Step Index on page 223](#).

- **Host** Enter **localhost** if the server is on your local machine; otherwise enter the machine name or IP address of the computer where the iProcess Engine is installed.
- **MBean Port** The port number that the iProcess Engine uses for JMX. By default the Port is **10025**. This was configured during installation of the iProcess Engine and stored in the **SWJMXConfig.port** entry in the **SWDIR\etc\swjmx.properties** file. If you cannot determine the port number, contact your iProcess administrator.



You now can run TIBCO Business Studio through a firewall, as long as you configure and open the necessary ports.

By default, TIBCO Business Studio uses port **10025** for the JMX engine and also uses port **10026** as an RMI port. The user will not be able to deploy if either of these ports are closed (which one could be for example if a customer was using a VPN).

So while **swjmx.properties** can define the JMX port, be aware you need to open that port as well as that port +1 as an RMI port.

- **MBean Name** The MBean Name is set in the configuration file **SWDIR\etc\swjmx_config.xml**. TIBCO Business Studio uses the default name **TIBCO:iProcessDeployment=default**. If you have changed the default MBean Name setting for the iProcess Engine by editing the configuration file, you must ensure that you also change **MBean Name** in TIBCO Business Studio so that both settings match.
 - **Username** Valid iProcess Engine user with either the PRODEF or ADMIN permission that can connect to the iProcess Engine (for example, **IPEADMIN**).
 - **Password** Password for the user connecting to the iProcess Engine.
 - **Repository Type** Select **Workspace**.
6. Click **Finish**. The new server is created and displayed in the Deployment view.

Task C Connect to the Server

You can connect to a server you have created as follows:

1. In the Deployment view, expand **Deployment Servers**.
2. Right-click the server name **TestDeploy** and select **Connect**.



The username and password you entered when you created the server is authenticated on the deployment server to prevent you from deploying a process to a server which you do not have authorization to use.

3. When you have connected, the Properties view for the server displays Connected as the Server State:

TestDeploy (Connected)	
Property	Value
Base	
Name	TestDeploy
Misc	
Workspace Modules	
Server	
Client Runtime	
Runtime	iProcess Engine Server
Server State	Connected

In addition, the icon in the Project Explorer and the status bar text in the lower left of the workspace changes to indicate that you are connected.

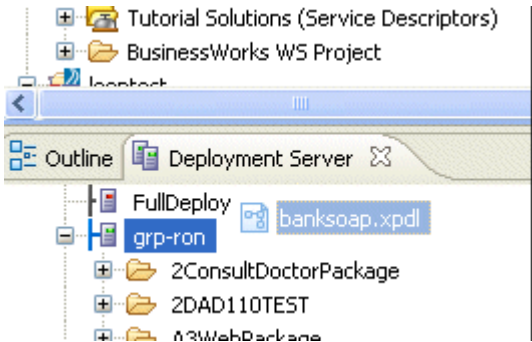


If you encounter any problems connecting to a Server, check the error log by selecting **Help > About TIBCO Business Studio**. From the resulting dialog, click **Configuration Details** then click **View Error Log**.

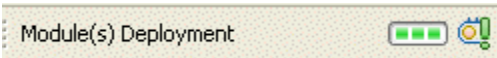
Task D Deploy the Process

Deploy the process to the iProcess Engine as follows:

- 1. Drag the XPDL package file onto the **TestDeploy** server in the Deployment view as follows:



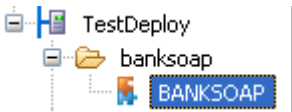
- 2. The deployment progress is shown in the lower right of the TIBCO Business Studio window. When deployment finishes successfully, a message and symbol are displayed:



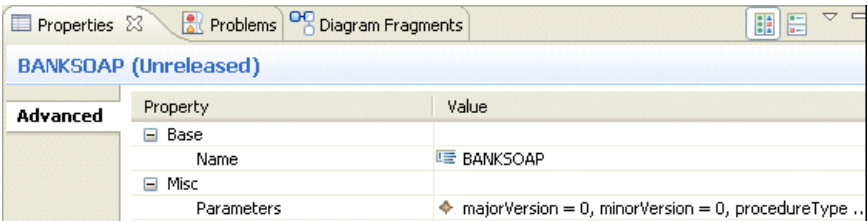
If errors are reported, the following symbol is displayed:



- 3. You can view the newly-deployed process in the Project Explorer:



The state of the procedure (Unreleased) and other details such as the version information is displayed in the Properties view:



For more information about managing deployed modules, see [Managing Deployed Modules on page 244](#).

Tutorial 8: Advanced Deployment

The previous tutorial showed the deployment of a simple process to iProcess. This tutorial shows how to deploy a process that uses more advanced features such as URL and security aliases, and forms deployment.

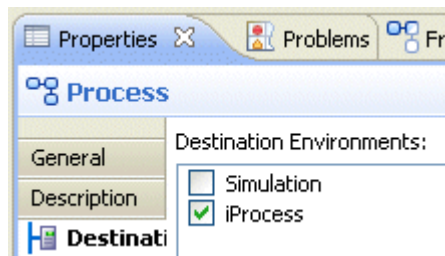
Prerequisites

In addition to the prerequisites from the previous tutorial, ensure that you have done the following:

- Installed the TIBCO iProcess™ Technology plug-ins (this will provide the JMS destination name defined on the server).
- Installed the TIBCO iProcess™ Workspace Plug-ins and created a security alias (see *TIBCO iProcess Web Services Plug-in User's Guide*).
- Defined the JMS queue name on using JMS admin tool (see "Managing JMS Providers" in *TIBCO BusinessWorks Connector User's Guide*).

Task A Install the Sample Process

1. Install the iProcess Developer Samples as described in [iProcess Developer Samples Project on page 18](#).
2. Expand the **Tutorial 8 Solution** package and select the **SettleVendorAcct-Process** process in the Project Explorer. In the Properties view, on the **Destinations** tab, ensure that the **iProcess** destination has been selected:



3. Confirm that there are no errors or warnings in the Problems view.

Task B Explore the Process

The first two tasks in the **SettleVendorAcct-Process** process are a user task, in which the vendor name and ID are entered, and a service task that calls a web service to pass the name and ID and return more vendor details. The service task uses the security alias **mySecurityProfile**:

Service Type:

Web Service

Operation:

Select

Clear

Import WSDL

Port Type:

portType

Operation Name:

GetAccountDetailsOp

Port Name:

intfwsGetAccountDetailsEndpoint1

Service Name:

intfGetAccountDetails-service

Transport:

SOAP over HTTP

Endpoint Resolution:
WSDL:

☒ Use local: ☐ Use remote:

Location:

VendorDetails.wsdl

Endpoint Name:

...

Clear

Security Profile:

mySecurityProfile

The vendor details are examined, and if there is a discrepancy, the Request for Appointment service task calls a web service that returns an appointment. The WSDL file for this web service will be obtained via a BusinessWorks live link at runtime. It uses a system participant **AppointmentRequestAlias** as a logical name for the endpoint name:

Service Type:	BW Service ▼
Operation:	Select Clear Import WSDL
Port Type:	__sol_VendorAccountSettlement_sol_iProcess_sp_Service_sp_Agent
Operation Name:	__sol_VendorAccountSettlement_sol_AppointmentDateTime
Port Name:	__sol_VendorAccountSettlement_sol_iProcess_sp_Service_sp_AgentServiceJMSPort
Service Name:	__sol_VendorAccountSettlement_sol_iProcess_sp_Service_sp_AgentService
Transport:	XML over JMS
Endpoint Resolution:	
WSDL:	<input type="radio"/> Use local: <input checked="" type="radio"/> Use LiveLink:
Location:	BookVendorAppointment.wsdl
Endpoint Name:	AppointmentRequestAlias

Task C Create the Deployment Server

1. If you have not already done so, switch to the Solution Design capability using the following menu.



2. In the Deployment view, right-click **Deployment Servers** and select **New > Server**.
3. Name the server **FullDeploy**.
4. Accept **iProcess Engine Server** as the **Runtime** and click **Next**.
5. Enter the runtime server parameters as described in [Enter the Runtime Server Parameters on page 225](#).

Use the **Test Connection** button to ensure that the details are correct, and click **Next**.

6. The following dialog has tabs that allow you to enter configuration information for MBean servers. This is required because the process we are deploying uses logical names for BusinessWorks endpoint names and security

profile information. The connection information you provide will be used to retrieve the required information from the runtime environment. Click the **BW Config** tab and select the **Connect to Server** checkbox:

New Server

Other Server Configurations
Set parameters of the other servers hosting the WS and BW endpoint MBeans

WS Config | **BW Config** | Security Profile Config

BW MBean Server Parameters:

Connect to Server: ☒

Host: localhost

MBean Port: 10021

Path: /server

MBean Name: TIBCO:SWAliasControlDAO=default

MBean Connector: rmi

Username:

Password:

☐ Save password

Saved passwords are stored on your computer in a file that is difficult, but not impossible, for an intruder to read.

Test Connection

< Back | Next > | Finish | Cancel

7. Enter the runtime server parameters for BusinessWorks configuration as follows.
 - **Host** Enter the machine name or IP address where BusinessWorks is located.
 - **MBean Port** The port number that the iProcess Engine uses for the JMX engine. The default port is **10021**. This was configured during installation of the iProcess Engine. If you cannot determine the port number, contact your iProcess Administrator.
 - **Path** Specifies the path on the server to the JMX service, where the JMX objects are stored. The default for **Path** is **/server** (this is the default iProcess Engine setting). If you have changed the default location of the JMX service,

you must ensure that you also change **Path** in TIBCO Business Studio so that both settings match.

- **MBean Name** Do not change this setting.
- **MBean Connector** Do not change this setting.
- **Username** Enter the username that you want to use to connect to the server.
- **Password** Enter the password that corresponds to the username you entered.

Use the **Test Connection** button to ensure that the details are correct, and click the **Security Profile Config** tab.

8. Select the **Connect to Server** checkbox and enter the parameters for the server where the security profile information will be retrieved.
 - **Host** Enter the machine name or IP address where Jetty is located (by default this is the same location as the iProcess Engine server).



The hostname that you enter must match the server name in the **TibcoiProcessWorkspace\ei_websvcs\urlaliasmanager.properties** file, in the following property:

```
URLAliasManager.aliasHosts=servername:port
```

If you do not enter the exact string listed in this property, you cannot view the security profile information in the EAI callout definition using iProcess Modeler.

- **MBean Port** The port number that the iProcess Engine uses for the JMX engine. For the **Security Profile Config** tabs, the default Port is **10010**. This was configured during installation of the iProcess Engine. If you cannot determine the port number, contact your iProcess Administrator.
- **Path** Specifies the path on the server to the JMX service, where the JMX objects are stored. The default for **Path** is **/server** (this is the default iProcess Engine setting). If you have changed the default location of the JMX service, you must ensure that you also change **Path** in TIBCO Business Studio so that both settings match.
- **MBean Name** Do not change this setting.
- **MBean Connector** Do not change this setting.
- **Username** Enter the username that you want to use to connect to the server.
- **Password** Enter the password that corresponds to the username you entered.

Use the **Test Connection** button to ensure that the details are correct, and click **Next**.

9. In the following dialog, select the **Connect to Server** checkbox and enter the parameters for connecting the TIBCO iProcess Workspace (Browser) and connection to the TIBCO Forms WebDAV server. These parameters are described in [Enter the Workspace \(Browser\) Server Configuration Details on page 229](#).

Test the connection by clicking the **Test Connection** button. After you have verified the connection settings, click **Finish**. The new server is created and displayed in the Deployment view.

Task D Connect to the Server

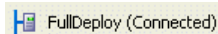
You can connect to a server you have created as follows:

1. In the Deployment view, expand **Deployment Servers**.
2. Right-click the server name **FullDeploy** and select **Connect**.



The username and password you entered when you created the server is authenticated on the deployment server to prevent you from deploying a process to a server which you do not have authorization to use.

3. When you have connected, the server state is displayed at the bottom of the TIBCO Business Studio window:

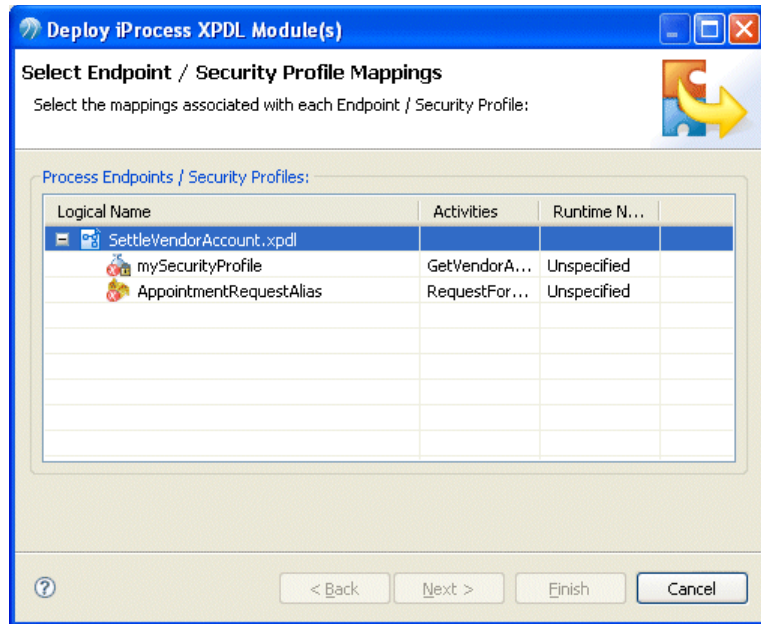


Task E Deploy the Process

Deploy the process to the iProcess Engine as follows:

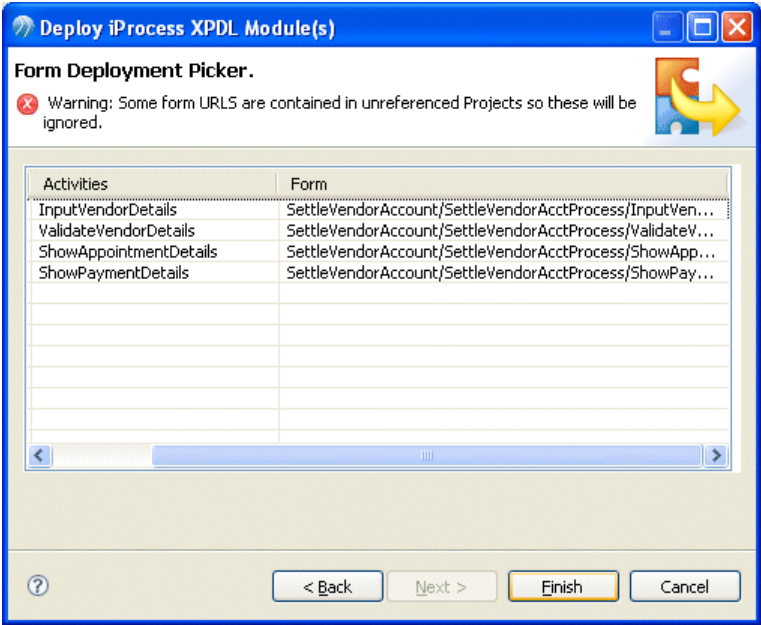
1. Drag the XPDL package file onto the **FullDeploy** server in the Deployment view as follows.

The following dialog is displayed:

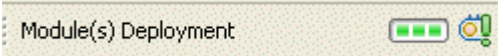


This dialog lists the logical names used in the process and the activities that use them. The **Runtime Name** column lists the actual runtime name that it has retrieved from the MBean servers. You must select a runtime name for each logical name before you can proceed. Note that the problem marker associated with each logical name remains until you select a runtime name and then click on a different part of the dialog. When you have finished the mapping, click **Next**.

2. The following dialog is displayed:



- Select the forms for deployment and click **Finish**.
3. The deployment progress is shown in the lower right of the TIBCO Business Studio window. When deployment finishes successfully, a message and symbol are displayed:



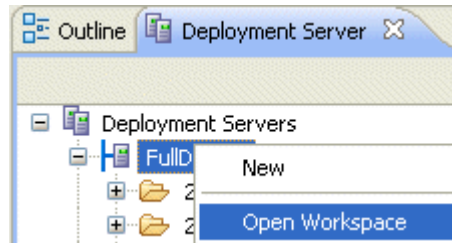
If errors are reported, the following symbol is displayed:



4. You can view the newly-deployed process in the Project Explorer. The state of the procedure (Unreleased) and other details such as the version information is displayed in the Properties view. For more information about managing deployed modules, see [Managing Deployed Modules on page 244](#).
5. To start a case of the process, do the following:
- Install the sample BusinessWorks project from the Welcome page. It is in the Solution Designer Samples section and is labelled **Sample TIBCO BusinessWorks Project for use with TIBCO iProcess Technology Plug-ins**. You can either open this project in TIBCO Business Studio if you have the TIBCO Designer™ Add-in for TIBCO Business Studio™, or you

can copy the project and open it with the standalone TIBCO Designer application.

- Start the BusinessWorks project in tester mode, select **Open Workspace**.



To open the iProcess Workspace (Browser) from within TIBCO Business Studio, you must have direct login enabled (see *TIBCO iProcess Workspace (Browser) Configuration and Customization*).

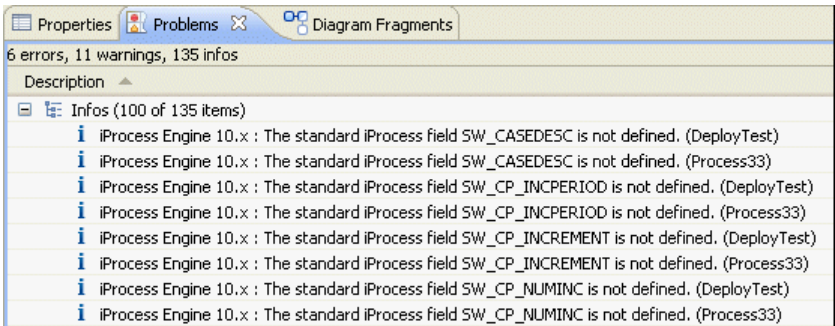
Tutorial 9: Exporting to the TIBCO iProcess Modeler

This tutorial shows how to take the process defined in the scripting tutorial (Tutorial 5: Adding a Script Task on page 39) and export it to the iProcess Modeler.

- Set the Destination Environment and Correct Validation Errors
- Export the Process from TIBCO Business Studio
- Import the Process into the TIBCO iProcess Modeler

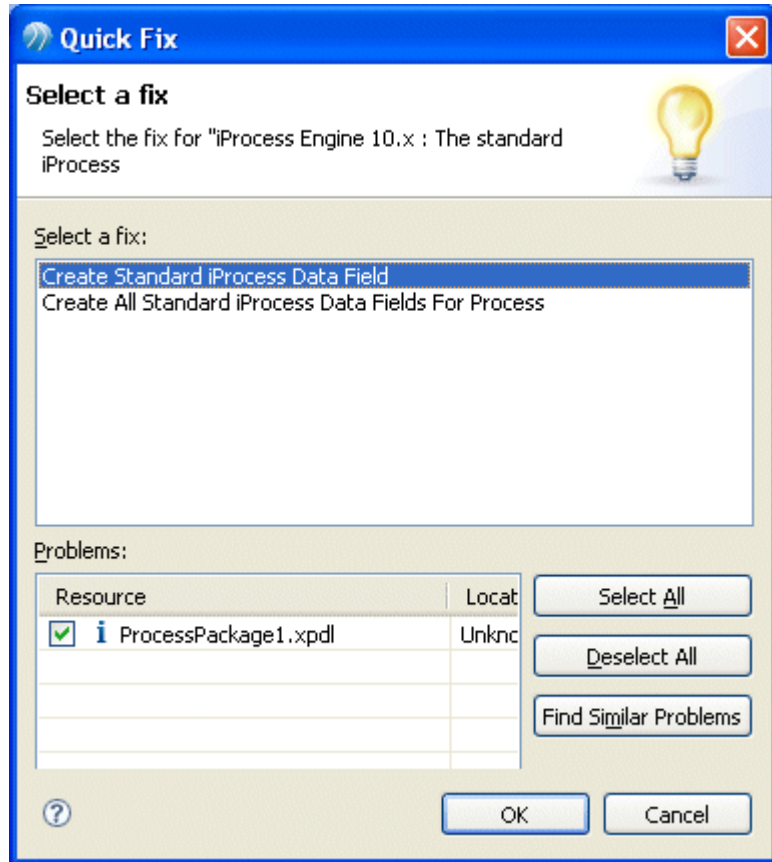
Task A Set the Destination Environment and Correct Validation Errors

1. Go to the Properties view for the loanApproval process and click the **Destinations** tab.
2. Confirm that the selected destination environment includes **Modeler** as a destination component (for more information, see Tutorial 1: Working with Destination Environments on page 19).
3. In the Problems view, check for errors and warnings. There should not be any errors or warnings if you completed the scripting tutorial correctly. However, there are some informational messages about standard iProcess fields that are not defined. For example:



The process can be exported without creating these fields, however there is a Quick Fix that allows you to easily create them as follows:

- a. Right-click one of the informational messages about standard iProcess fields and select **Quick Fix**. The following dialog is displayed:



- b. Select **Create All Standard iProcess Data Fields For Process** and click **OK**.



You can create system iProcess data fields (such as SW_DATE), in a similar manner, using the supplied Quick Fix. These fields are also created automatically when you deploy a process to the iProcess Engine.

4. Save the package and check the Problems view to ensure that any validation errors have been corrected.

Task B Export the Process from TIBCO Business Studio

Export the process and its containing package as follows:

1. Select the package that contains the **loanApproval** process.
2. Select **File > Export**. The **Export** dialog is displayed.
3. Select **Business Process Management > TIBCO iProcess Modeler XPDL** and click **Next**.
4. Select the package that you want to export.
5. Select the destination for the export.



Although you can select **Project** as an export destination, TIBCO recommends that you select the **Path** radio button and enter the path to the directory that contains the iProcess Workspace software.

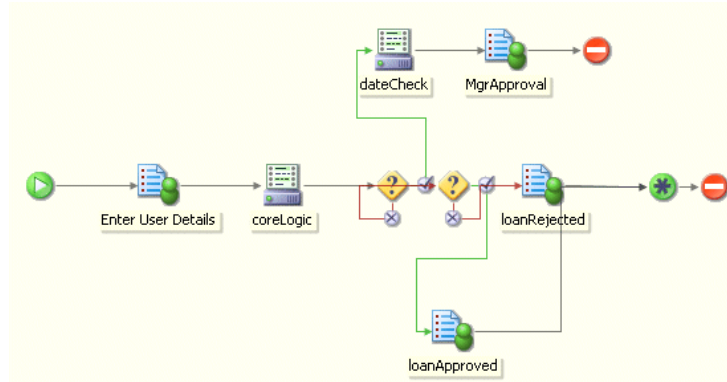
6. Click **Finish**.
7. The process is exported to the directory you specified, or if you exported to the project, the exported process appears in the Project Explorer under **Exports > iProcess Modeler XPDL** and in the file system in your workspace in the `\projectname\Exports\iProcess Modeler XPDL` directory.

Task C Import the Process into the TIBCO iProcess Modeler

Import your TIBCO Business Studio process as described in the *TIBCO iProcess Modeler Procedure Management* guide. See [Object Mapping on page 112](#) for information about how TIBCO Business Studio objects are mapped into TIBCO iProcess objects.

1. Start the TIBCO iProcess Client.
2. In the Procedure Manager, select the library into which you want to import the process.
3. Select **Procedure Management > Load From** and browse for the XPDL that you exported from TIBCO Business Studio. Click **OK**.
4. Information about the Package and Procedures is displayed. Click **Load**.

5. The **loanApproval** procedure when opened in iProcess Modeler looks like this:



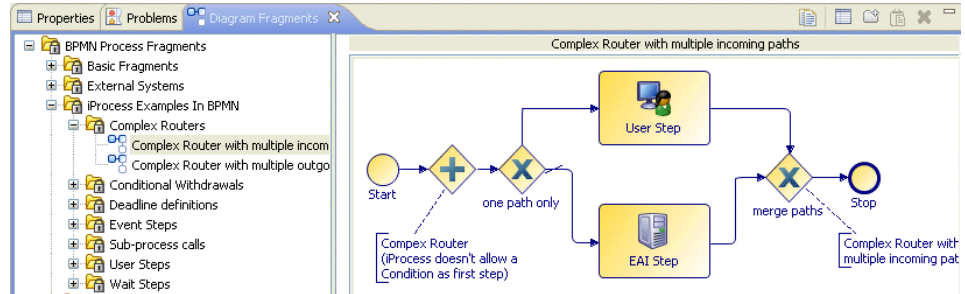
This chapter is intended for users who are familiar with the TIBCO iProcess Modeler, but are new to using TIBCO Business Studio. It describes how to implement some common iProcess constructs in TIBCO Business Studio.

Topics

- [iProcess Examples in BPMN, page 74](#)
- [Using the Advanced Tab, page 75](#)
- [Configuring Web Services, page 78](#)
- [Configuring BusinessWorks Service Tasks, page 82](#)
- [Implementing TIBCO iProcess Conductor Processes, page 84](#)
- [Customizing the Audit Trail, page 87](#)
- [Transaction Control Steps, page 89](#)
- [User Tasks, page 93](#)
- [Deadline Expressions, page 97](#)
- [Withdraw Links, page 98](#)
- [Dynamic Sub-Procedures, page 103](#)
- [Graft Steps, page 106](#)
- [Reusable Sub-Process Call Task Validation, page 109](#)
- [Groups and Roles, page 110](#)
- [Public Steps and Events, page 111](#)
- [Object Mapping, page 112](#)

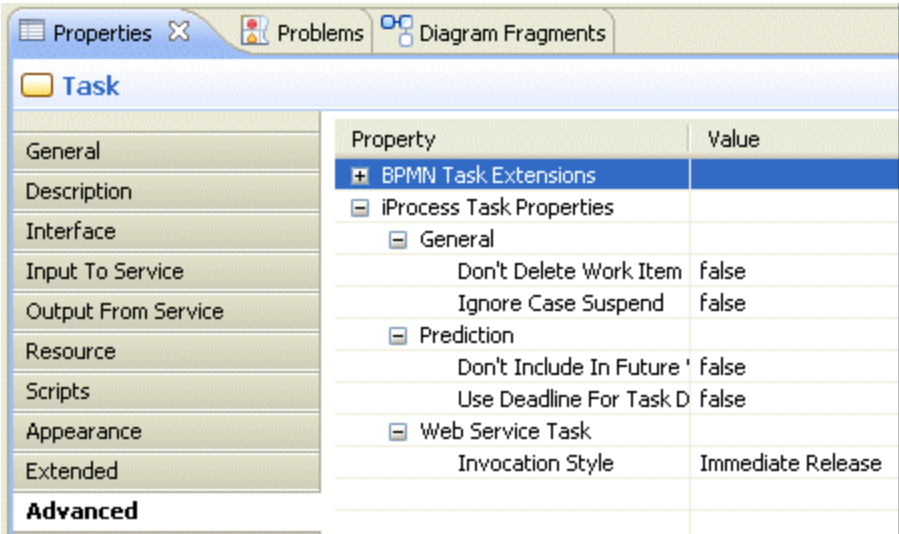
iProcess Examples in BPMN

In the Diagram Fragments view, there is a category of diagram fragments called **iProcess Examples in BPMN**. You can use these as templates when creating a new process, or browse the fragments in the Fragments view to see how common iProcess constructs can be created in BPMN.



Using the Advanced Tab

Many of the iProcess-specific settings for events, tasks, and processes are found on the **Advanced** tab of the Properties view when you select the iProcess destination environments. For example, a service task of the type Web Service has the following options on the **Advanced** tab:



Common
Properties

The following properties are displayed on the **Advanced** tab for user tasks, service tasks, script tasks, and reusable sub-process tasks. They also apply to send tasks and manual tasks, although these are not exportable to iProcess.

iProcess Property	Description
Don't Delete Work Item on Withdraw	<p>If set to true, when the process is executed in iProcess, and the deadline on an outstanding step expires or it is withdrawn as an action (release or deadline expiry) of another step:</p> <ul style="list-style-type: none">the deadline actions are processed.the step remains outstanding (the step remains in the work queue or the sub-procedure case is not purged).

iProcess Property	Description
Ignore Case Suspend	<p>If set to true, when the process is executed in iProcess, the step will still be processed as normal while a case is suspended.</p> <p>If set to false, (the default setting), the step is not processed while the case is suspended.</p>
Don't Include In Future Work Items List	<p>If set to false, the step is included in prediction.</p> <p>If set to true, the step is excluded from prediction. For example, you may want to exclude EAI steps because these steps are processed automatically. This means that although these steps are taken into account as part of the case prediction, they do not appear in the outstanding step list.</p>
Use Deadline for Task Duration	<p>If set to true, if a deadline is set then the task duration will be the same as the deadline. This means you do not have to manually set the duration to be the same as the deadline.</p>

Depending on the type of event or task you have selected, other properties are displayed. For more information, see [Reference on page 253](#).

Invocation Styles

The invocation style of service task is set on the **Advanced** tab. In iProcess, different types of EAI plug-ins use different terminology to refer to the same invocation style. For consistency, the following terminology has been adopted in TIBCO Business Studio:

TIBCO Business Studio Term	iProcess Database Plug-in Term	iProcess Web Services Plug-in Term	iProcess BusinessWorks Plug-in Term	iProcess Conductor Term
Immediate Release	Delayed Release - Never	Automatic Delayed Release	Immediate Release	Synchronous/ Immediate
Delayed Release	Delayed Release - Always	Manual Delayed Release	Delayed Release	Delayed Release
Conditional Delayed Release	Delayed Release - Conditional	n/a	n/a	

TIBCO Business Studio Term	iProcess Database Plug-in Term	iProcess Web Services Plug-in Term	iProcess BusinessWorks Plug-in Term	iProcess Conductor Term
Immediate Release (Asynch with Reply)	n/a	Asynch With Reply	n/a	

Configuring Web Services

In TIBCO Business Studio, web service calls are implemented using service tasks as follows:

1. Import a WSDL file into the project (either from the **File > Import** menu or directly from the service task). TIBCO Business Studio supports design time import of a WSDL file (this can be from a file, URL, or UDDI registry). You can also use a system participant as a logical name for a WSDL endpoint.
2. Place the service tasks in the process.
3. Drag the web service operation from the Project Explorer and drop it onto the service task. This populates the relevant fields in the Properties view of the service task. For example:

Service Type:

Operation:

Port Type:

Operation Name:

Port Name:

Service Name:

Transport:

Endpoint Resolution:

WSDL: ☐ Use local: ☒ Use remote:

Location:

Endpoint Name:

Security Profile:

4. Complete the service input and output mappings using the **Input to Service** and **Output From Service** tabs. See [Using the Mapper on page 149](#).

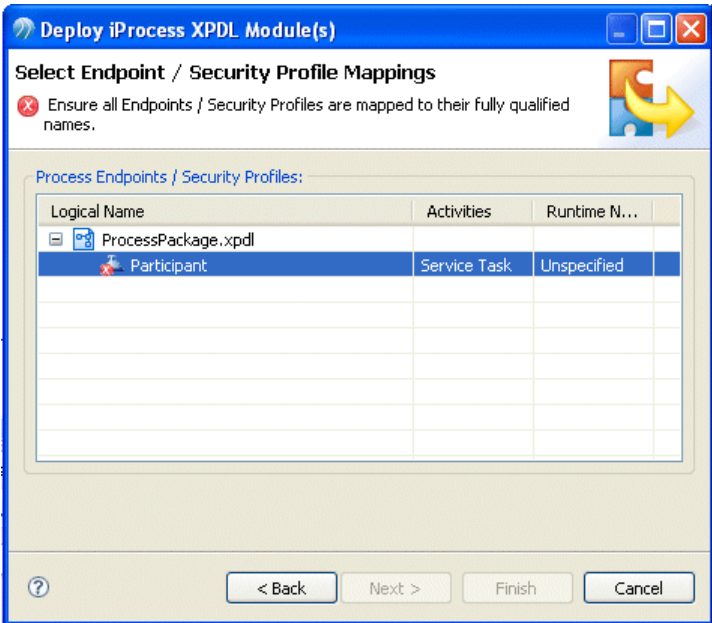


The Delayed Release ID field is displayed in the mapper if you select **Web Service** as the **Service Task Type**.

5. Click the **Advanced** tab. In addition to the properties described in [Using the Advanced Tab on page 75](#), configure the **Advanced** tab parameters that are

specific to service tasks as described in [Service Task Properties \(Web Service/BusinessWorks Service\)](#) on page 261.

- 6. When you deploy the process, any logical names (system participants) that you used for endpoints or security profiles must be mapped to URL and security profiles in the runtime environment in the following dialog:



For more information, see [Deploying a Module](#) on page 235.

Abstract and Concrete WSDL Files

TIBCO Business Studio allows you to specify a web service operation from either an abstract or concrete WSDL file with a service task.

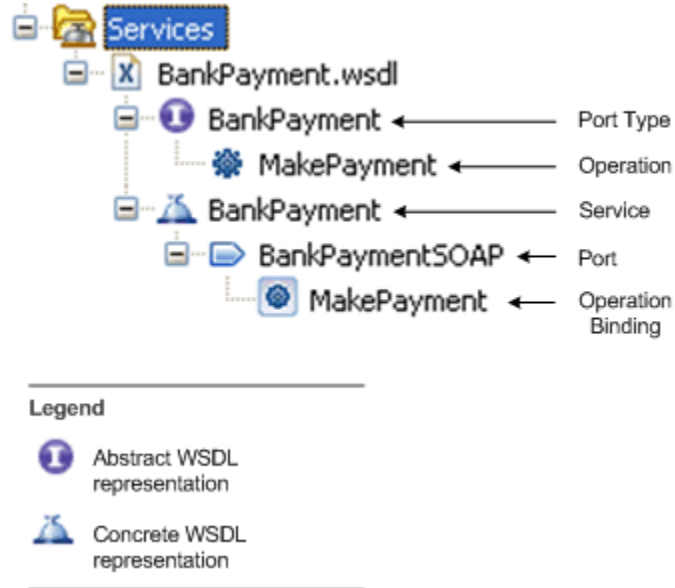


Only concrete WSDL file invocations can be deployed to iProcess.

An *abstract WSDL* document defines an abstract messaging model without reference to protocols or encodings, and consists of port types and operations.

A *concrete WSDL* document contains the abstract definitions and the communication protocols and data formats by which the operations defined in the abstract WSDL document can be invoked. A concrete WSDL file consists of the elements from the abstract WSDL file as well as an operation binding.

In the Project Explorer, both the abstract and concrete parts of the WSDL file are represented when you expand the WSDL file:



Both an operation (from an abstract WSDL file) and an operation binding (from a concrete WSDL file) can be dragged onto a web services or BusinessWorks service task.

Using Aliases

TIBCO Business Studio supports URL and security profile aliases that are equivalent to those found in the TIBCO iProcess Web Services Plug-in. The use of aliases allows you to change the location of the WSDL file, the WSDL endpoint, or security information without having to modify the Web Services step. For example, when a procedure is migrated from a test environment to a live environment, the URL alias can be updated to point to the new location of the WSDL file. For more information about aliases, see *TIBCO iProcess Web Services Plug-in User's Guide*.

The logical names you use in TIBCO Business Studio are defined as system participants, and do not need to match the alias names in the iProcess Web Services Plug-in, because during deployment you must map the TIBCO Business Studio URL and security profile logical names to specific aliases in the iProcess Web Services Plug-in. This alias information is retrieved from an MBean server, that you configure in TIBCO Business Studio - see [Enter the MBean Server Configuration Parameters \(Optional\)](#) on page 227.



Aliases can be mapped only during deployment; they are ignored when exporting to iProcess Modeler XPDL.

Using Scripts for Mapping

TIBCO Business Studio supports both XML Path Language (XPath) expressions and Extensible Stylesheet Language Transformations (XSLT) to create mappings between process fields/formal parameters and service input/output parameters. For more information, see [Applying Scripts to a Mapping](#) on page 150.

Data Transport Mechanisms

There are two data transport mechanisms for web services that you can specify in TIBCO Business Studio to send data between the iProcess Engine and an external application.

- Simple Object Access Protocol (SOAP) requests over the Hypertext Transfer Protocol (HTTP) - (SOAP/HTTP)
- Extensible Markup Language (XML) text using a Java Message Server (JMS) - (XML/JMS).

The main difference between using SOAP/HTTP and XML/JMS is that SOAP/HTTP uses Web Services Description Language (WSDL) source to determine how the text is sent. However, when using XML/JMS you must define your own XML schema for sending data between an iProcess Engine and an external application.

Select the desired transport in the Properties view:

Transport:	HTTP
Endpoint Resolution:	XML Over JMS

For more information, see *TIBCO iProcess Web Services Plug-in User's Guide*.

Configuring BusinessWorks Service Tasks

TIBCO Business Studio allows you to create a service task where the WSDL file is obtained from TIBCO BusinessWorks using a live link.

1. Import a WSDL file into the project using a BusinessWorks live link. To do this, you must create and connect to a JMS server (see [Creating a JMS Server \(TIBCO BusinessWorks Only\) on page 136](#) and [Connecting to a JMS Server \(TIBCO BusinessWorks Only\) on page 138](#)).



If after importing a WSDL file using a BusinessWorks live link, the WSDL file is changed, the changes are not automatically reflected in TIBCO Business Studio. You must use the Service Import Wizard to reimport the changed WSDL file.

2. Place the service task in the process, and select **BW Service** for the **Service Type**.
3. Select a system participant to use as an alias for the BusinessWorks **Endpoint name**.



Only system participants can be used as aliases for BusinessWorks endpoint names.

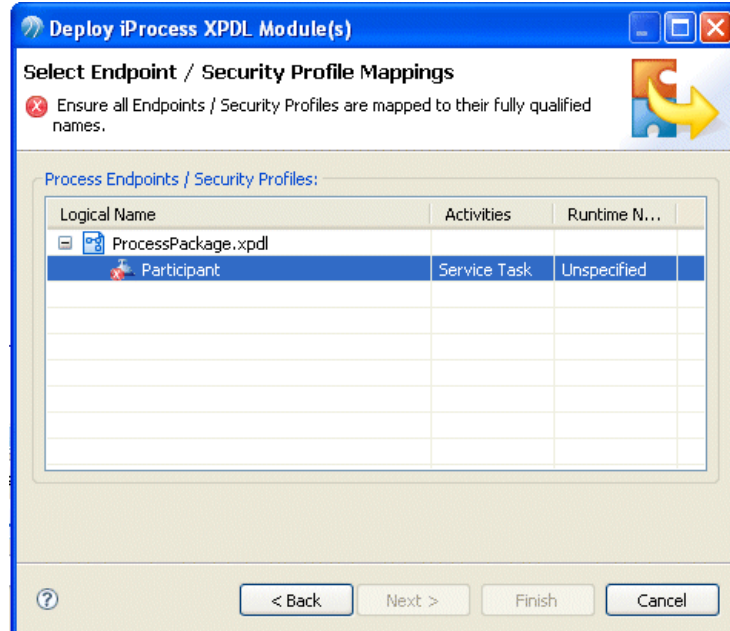
4. Complete the mapping of the process fields/formal parameters and service input/output parameters using the **Input to Service** and **Output From Service** tabs. You can use either the mapper or enter scripts in the area provided (see [Using Scripts for Mapping on page 81](#)).



The Delayed Release ID fields are displayed in the mapper if you select **BW Service** as the **Service Task Type**.

5. Click the **Advanced** tab. In addition to the properties described in [Using the Advanced Tab on page 75](#), configure the **Advanced** tab parameters as described in [Service Task Properties \(Web Service/BusinessWorks Service\) on page 261](#).

6. When you deploy the process, map the endpoint aliases in the following dialog:

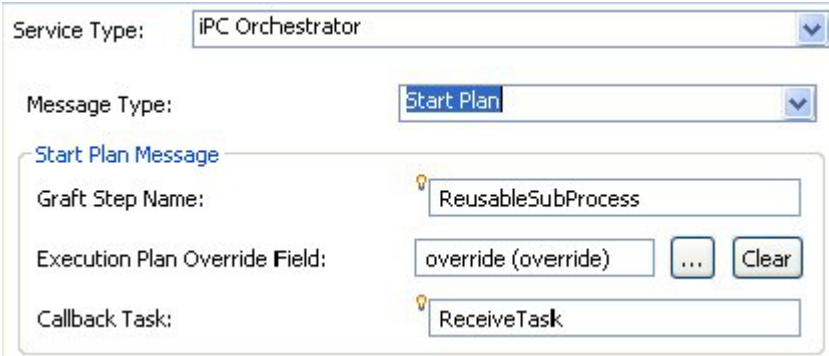


Implementing TIBCO iProcess Conductor Processes

You can design and implement processes for the fulfillment of orders in TIBCO Business Studio. This includes such tasks as defining processes, process components, and execution plans. TIBCO Business Studio provides service task types that provide the functionality of EAI Orchestration and EAI Order steps. The equivalent functionality of an EAI Transform step is provided by the transform script step in TIBCO Business Studio. Processes are also validated to ensure that they adhere to iProcess Conductor-specific practices.

Orchestrator Service Tasks

Orchestrator service tasks can be used to send an Activate Sink, Start Plan, or Task Complete message. For example, a service task can be configured as follows to start an execution plan in the parent fulfillment process:



Service Type: iPC Orchestrator

Message Type: Start Plan

Start Plan Message

Graft Step Name: ReusableSubProcess

Execution Plan Override Field: override (override) ... Clear

Callback Task: ReceiveTask



- In this example, the **Graft Step Name** specifies a reusable sub-process that is set up to be a graft step (see [Graft Steps on page 106](#)). This is used to orchestrate the required execution plan.
- The **Execution Plan Override Field** is a data field or parameter that contains the unique identifier for the execution plan or execution plan template.
- The **Callback Task** specifies a receive task that is used to pass the execution plan ID back to the fulfillment process (this was achieved in iProcess Modeler using an ad-hoc orchestration event step).

For more information, see [Configuring TIBCO iProcess Conductor Service Tasks on page 180](#).

Order Service Tasks

Order service tasks can be used to change the state of an order, or to change the state of an order line. For example, you can configure an order service task to change the state of an order from **Feasibility** to **Execution** as follows:

Service Type: IPC Order

Order Identification

Order Primary Key Field: SWO_ORDERNUMBER (SWO_ORDERNUMBER) ... Clear

Order Amendment Index Field: SWO_AMENDINDEX (SWO_AMENDINDEX) ... Clear

Order State Change

Transition: Execution

Ad-Hoc Event Step Name: ReceiveTask

Event Case: This Procedure

Event Case Number: ... Clear

Event Case Procedure Name: ... Clear

Order Line State Change

Line Number Selection: ... Clear

Order Line Status: Unspecified

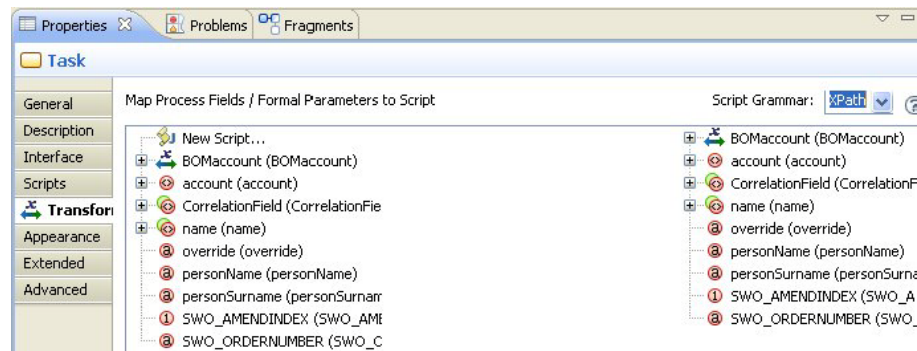
For more information, see [Configuring TIBCO iProcess Conductor Service Tasks on page 180](#).

Transformation Steps

TIBCO iProcess Conductor Transformation steps can be implemented in TIBCO Business Studio using a script task of the type Transform:

Script Defined As: Transform

Selecting this option displays the **Transform** tab, on which you can create XPath or XSLT mappings:



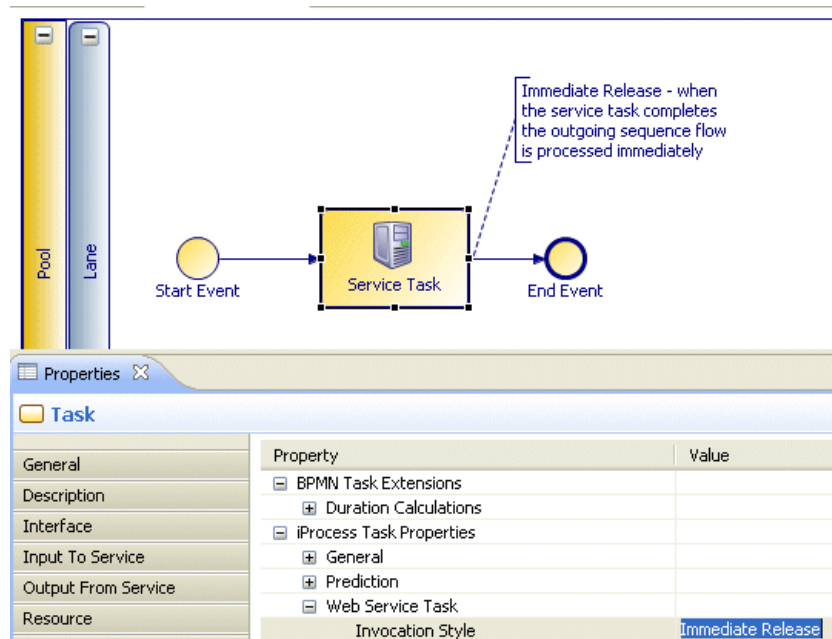
For more information, see [Transforming Data \(XPath and XSLT\)](#) on page 188.

Customizing the Audit Trail

You can add your own custom label to the audit trail for script tasks and service tasks by entering text or a script on the **Scripts** tab in the Properties view. For more information, see [Creating Audit Scripts on page 202](#).

Immediate Release

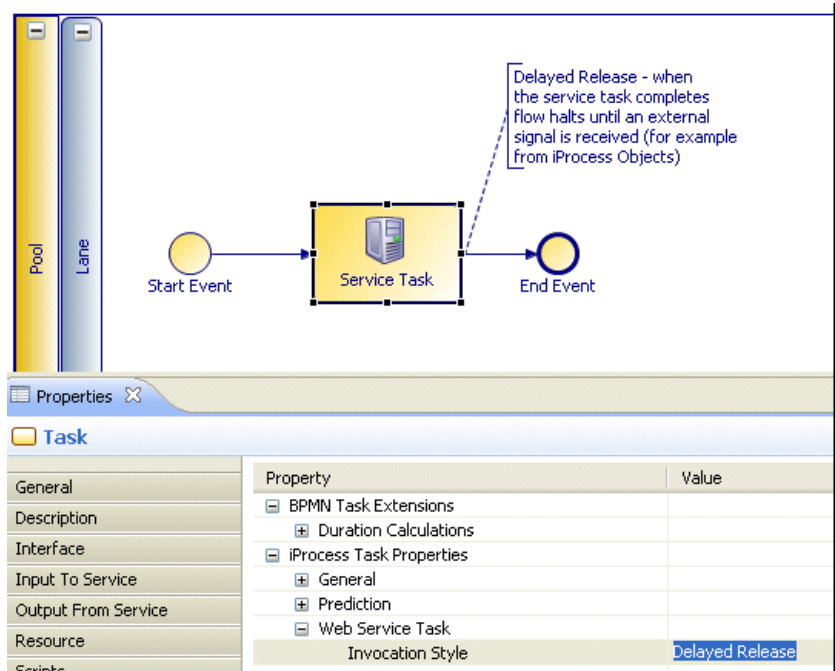
An immediate release step is equivalent to the following in TIBCO Business Studio:



The service task in this example is processed immediately in a single transaction.

Delayed Release

A delayed release step can be specified as follows in TIBCO Business Studio:



In this example, the outgoing sequence flow from the service task does not continue until an external signal is received.

Transaction Control Steps

The iProcess Background process bundles all consecutive/concurrent EAI steps into the same transaction. If they are successful, the resulting data is committed to the database. If there is a failure in one step then all steps are rolled back.

Transaction Control (TC) steps are used primarily to break up sequences of EAI steps into groups that should be committed or aborted together.

In TIBCO Business Studio, transactions are created by specifying an embedded sub-process, and selecting the **Is a transaction** check box.



You must include an end event in a embedded sub-process that is marked as a transaction. When deployed, the end event becomes a commit step.

If the **Start a new transaction** checkbox is selected, upon deployment a commit step is inserted at the beginning of the transaction. If this checkbox is not selected, upon deployment a complex router is inserted.

Deferring New Transactions

On the **Advanced** tab for the Properties view of an embedded sub-process, there is a setting that allows you to control whether the iProcess background immediately begins processing the new transaction that follows the transaction created by the embedded sub-process:

Embedded Sub-Process		
	Property	Value
General	iProcess Task Properties	
Description	Transaction Control	
Scripts	Defer Start Of New Transaction For Post-Embedded Sub-Process Tasks	false
Appearance	Non-Deferred Transaction Failure Retry Delay (Minutes)	5
Extended		
Advanced		

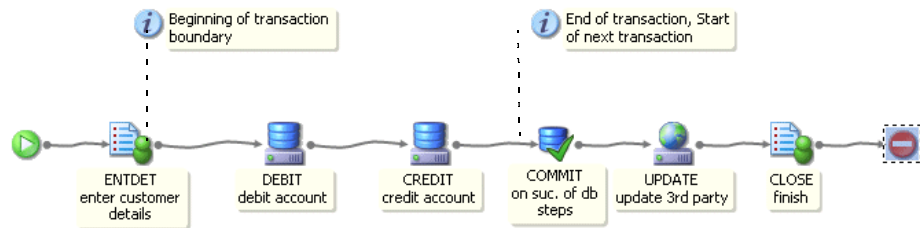
Processing occurs as follows:

- If the start of the new transaction is deferred (the setting is changed to **true**), the transaction created by the embedded sub-process is committed. Then, a message is placed in the Mbox queue instructing the background to continue processing from the first post-embedded sub-process task. However, processing of this task *deferred* because other messages in the Mbox queue are processed before returning to the post-embedded sub-process task.
- If the start of the new transaction is not deferred (the setting is **false**), the transaction created by the embedded sub-process is committed, and

processing of the first post-embedded sub-process task continues immediately. In this case you can also specify a time (in minutes) for which the background should wait before attempting to retry the transaction created by the embedded sub-process. The transaction control step has a configurable deadline, and if the deadline period expires, then the deadline manager posts an instruction to one of the background processes to retry this case. This retry instruction will be retried with the standard 5minute x12 times mbox retry mechanism (or any other that is configured) but Business Studio does not configure the Mbox retry settings.

Example 1

Consider the following iProcess procedure that transfers money from one bank account to another:



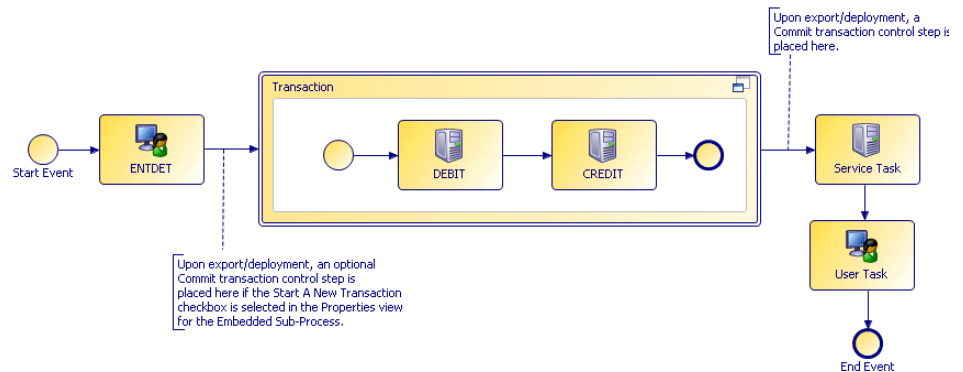
1. The transaction details are entered in the form displayed by the first step.
2. When the form is released, a new transaction begins in which the first EAI database step debits the source account.
3. The second EAI database step deposits the funds in the target account.
4. The COMMIT TC step immediately following the EAI database steps ensures that if both database steps are successful, the transaction is committed.
5. The UPDATE EAI Web Services step uses the data committed to the database to update a 3rd party system.



If the COMMIT TC step were not present, the third-party system would not have access to the latest credit/debit updates because they would not yet have been written to the database (since the EAI Web Services step and EAI database steps would be part of the same transaction).

6. The procedure concludes with a form.

This can be implemented in TIBCO Business Studio as follows:

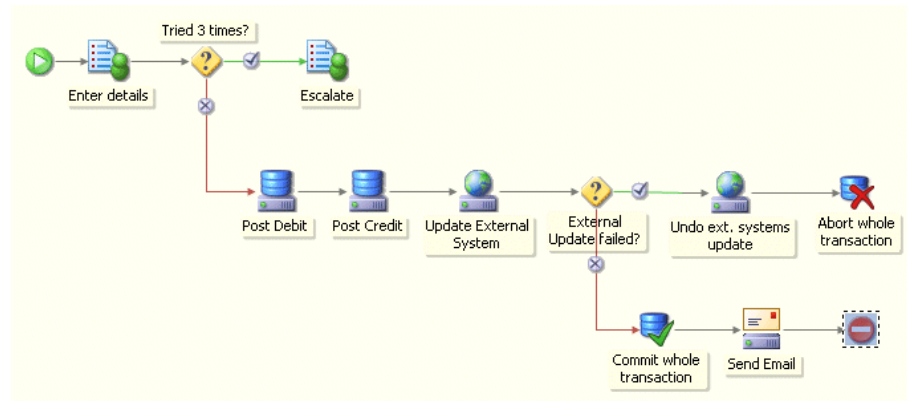


The embedded sub-process with the database service tasks is transactional as indicated by the double line boundary. Flow does not continue until both of these tasks succeed.

Example 2

This procedure is a variation of Example 1, however it has the following enhancements:

- A condition ensures that if the transaction fails more than three times, a form is displayed to an Administrator escalating the case for investigation.
- The non-transactional step that performs an external system update is checked with a condition. If the update was unsuccessful, its effects are undone by another EAI step.

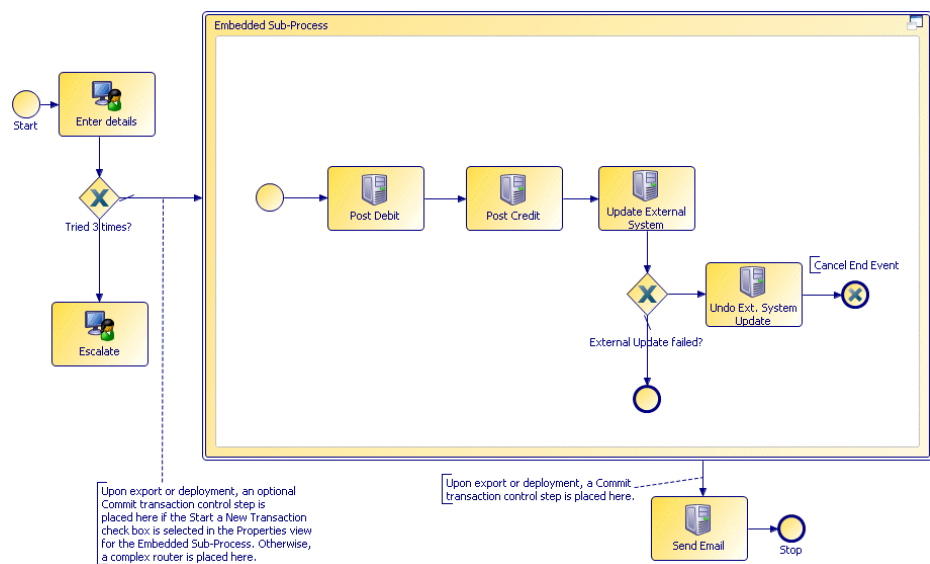


The steps in the process do the following:

1. The transaction details are entered in the form displayed by the first step.
2. The condition checks **sw_qretrycount** is greater than 3. If it is, the case is escalated for investigation.
3. A new transaction begins in which the first EAI database step debits the source account.
4. The second EAI database step deposits the funds in the target account.
5. The Update External System step is a web services step.
6. The condition evaluates the results of the Update External System step. In the case of an error with the update, another EAI web services step undoes the effect of the update, and the entire transaction is aborted. In the case of a correct result, the entire transaction is committed.
7. The procedure concludes with an Email step in another transaction.

The key points in this procedure are that the failure of the transaction is catered for, and the effects of the non-transactional update step is undone if the transaction fails.

This can be implemented in TIBCO Business Studio as follows:



User Tasks

This section describes some of the features available for user tasks.

Forms

You can use TIBCO Business Studio Forms to create rich, fully-featured forms that can be displayed by your user tasks. Selecting a user task and using the right-click menu, you can automatically generate forms. For more information, see *TIBCO Business Studio Forms User's Guide*.

Scripts

You can specify scripts for a user task on the **Scripts** tab. The following table shows the types of user task scripts that you can create.

TIBCO Business Studio User Task	iProcess Procedure Form Command	Result in iProcess Procedure
Open Script	Initial	This command is run when the work item form is opened from the user's Work Queue.
Close Script	Keep	This command is run when the form is returned to the user's Work Queue.
Submit Script	Release	This command is run when the form is released.

You can use iProcess Engine functions in your scripts. To see a list of those available functions, process data, and JavaScript templates, press Ctrl+Space from the Script area to display content assist.



The iProcess functions available for a user task are a subset of those available for script tasks.

Priority and Step Permissions

In iProcess Modeler, when defining your procedure you can set the base priority level and escalation criteria by using a series of special system fields. These system fields take the form of SW_CP_xxx (CP meaning Case Priority), and are set using the following dialog:

The **Step Definition** dialog box is shown with the following sections and fields:

- Definition** | Addressees | Deadlines | Duration | Status
- Form Commands**
 - Initial:
 - Keep:
 - Release:
- Step Priority**
 - Base Priority Value (1-999):
 - Automatic Priority Escalation**
 - Increment (negative to raise):
 - Number of increments (-1=unlimited):
 - Increment Period:
 - Period Type ("m"ins, "h"ours, "d"ays):
- Permissions**
 - ☐ Forward ☐ Edit
 - ☐ Don't delete work items on withdraw
 - ☐ Ignore Case Suspend
- Buttons: OK, Cancel, Help

Also in the previous dialog are two checkboxes that control whether other users can forward the work item (**Forward**), or copy the contents of the work item (**Edit**).

To specify step priority and escalation values in TIBCO Business Studio, click the **Advanced** tab in the Properties view for the user task:



The properties for Priority are set to data fields that correspond to the standard iProcess fields used for priority (for example, SW_CP_INCREMENT). If these data fields do not exist, you can create them using a quick fix as described in [Set the Destination Environment and Correct Validation Errors on page 68](#). For more information about iProcess system fields, see [TIBCO iProcess System Fields on page 283](#).

Task		
General	Property	Value
Description	[-] BPMN Task Extensions	
Interface	[+] Duration Expressions	
Resource	[-] iProcess Task Properties	
Scripts	[+] General	
Appearance	[+] Prediction	
Extended	[-] User Task	
Advanced	Work Item Copy Enabled	false
	Work Item Forwardable	false
	[-] Work Item Priority	
	Priority Increment Expression	SW_CP_INCREMENT
	Priority Increment Period Expression	SW_CP_INCPERIOD
	Priority Increment Period Type Expression	SW_CP_PERIODTYP
	Priority Number Of Increments Expression	SW_CP_NUMINC
	Priority Value Expression	SW_CP_VALUE

Set the values as follows:

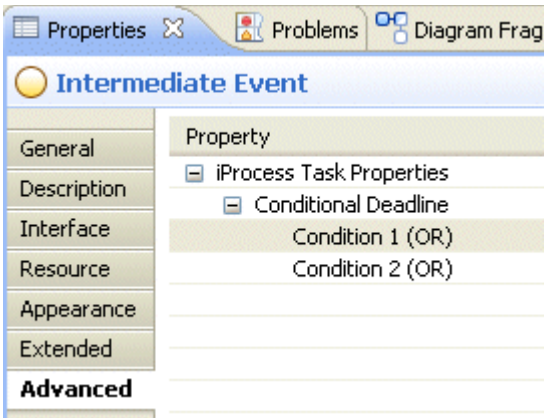
iProcess Property	Description
Work Item Copy Enabled	If set to true , allows the user to copy the contents of the form when it is displayed to them in iProcess.
Work Item Forwardable	If set to true , allows the user to forward a step to the work queue of another user (subject to the user attribute USERFLAGS in iProcess).

iProcess Property	Description
Priority Increment Expression	This integer expression defines the amount that will be added to the item's Priority Value whenever the Increment Period expires. By default this is set to the value of the data field SW_CP_INCREMENT (integer number, length 4). Assigning the data field a negative number causes the priority to increase (move closer toward 1, the highest priority).
Priority Increment Period Expression	This is the time period, in the units specified in Period Type, which must expire before the item's Priority Value is incremented. By default this is set to the data field SW_CP_INCPERIOD (integer number, length 4).
Priority Increment Period Type Expression	<p>This is the unit of measure of the Increment Period. By default this is set to the data field SW_CP_PERIODTYP (a string). Assign this data field one of the following values:</p> <ul style="list-style-type: none"> • "M" or "m" for minutes • "H" or "h" for hours • "D" or "d" for days
Priority Number of Increments Expression	This is the number of increments that will be added to the item's Priority Value. By default this is set to the data field SW_CP_NUMINC (integer number, length 4).
Priority Value Expression	This is the Base Priority Value that each step will be given. By default this is set to the data field SW_CP_VALUE. This can be set to between 1 and 999, where 1 is the highest priority.

Deadline Expressions

Deadlines in TIBCO Business Studio are set by timer events. You can either use JavaScript or specify a constant period (see [Timer Event Scripts on page 205](#)).

You can also specify a conditional deadline on the **Advanced** tab:



Specify the conditional deadline as follows. If either condition is true, the deadline is set. Also, if neither conditions are present the deadline is set.

iProcess Property	Description
Condition 1 (OR)	Enter any deadline conditions required in text format. You can set a deadline to only take effect if a certain condition is true. For example, a deadline could be set if the anticipated completion date for the property purchase is less than 4 weeks after the application date. This condition is evaluated when the step is sent out.
Condition 2 (OR)	Second condition (see previous).

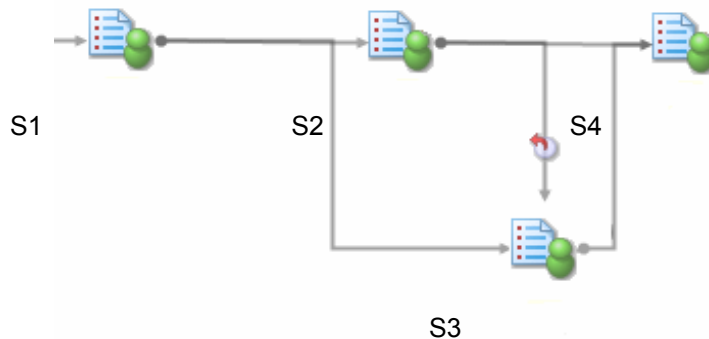
Withdraw Links

A withdraw link in iProcess means that the step being connected to will be withdrawn from the work queue. For example, you might have two steps that are sent out in parallel but only one needs to be actioned and released.

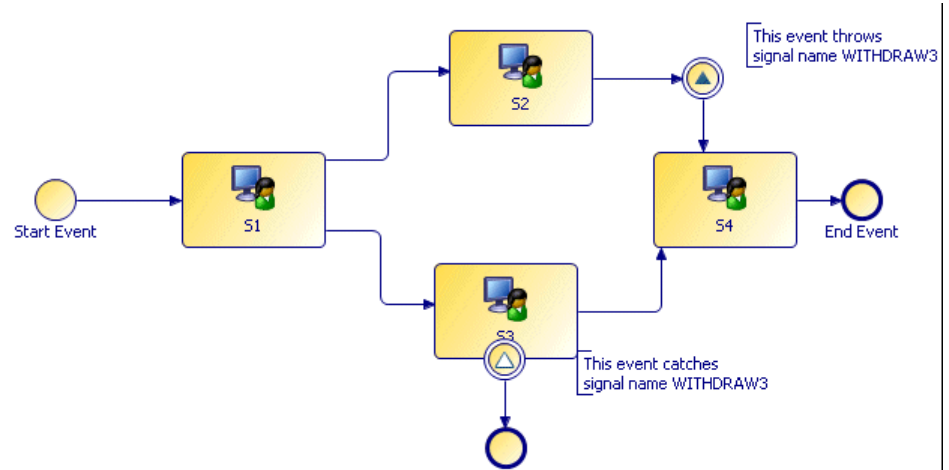
Withdraw links can be implemented in TIBCO Business Studio using throw and catch signal event pairs. With a throw/catch signal event pair, an *inflow* signal event "throws" the signal, and if there is an active task with a signal event on the task boundary, that event "catches" the signal, cancelling the task and following the exception flow.

Example 1

In the following iProcess example, if step **S2** is released before **S3**, then **S3** will automatically be withdrawn from the work queue.



In TIBCO Business Studio, this is implemented using throw/catch signal events as follows:



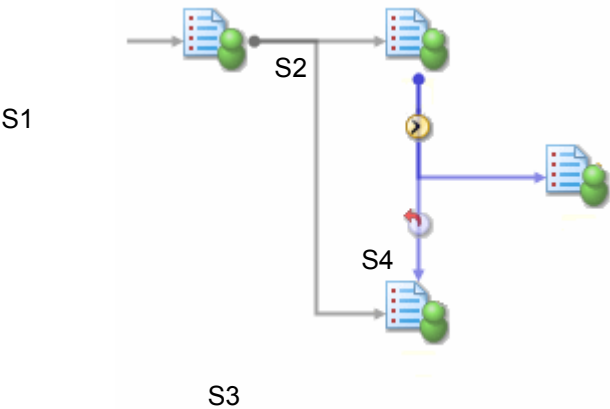
In this example, if **S2** completes while **S3** is in progress, the inflow signal event throws a signal, which is caught by the signal event on the boundary of **S3**. This withdraws **S3**.



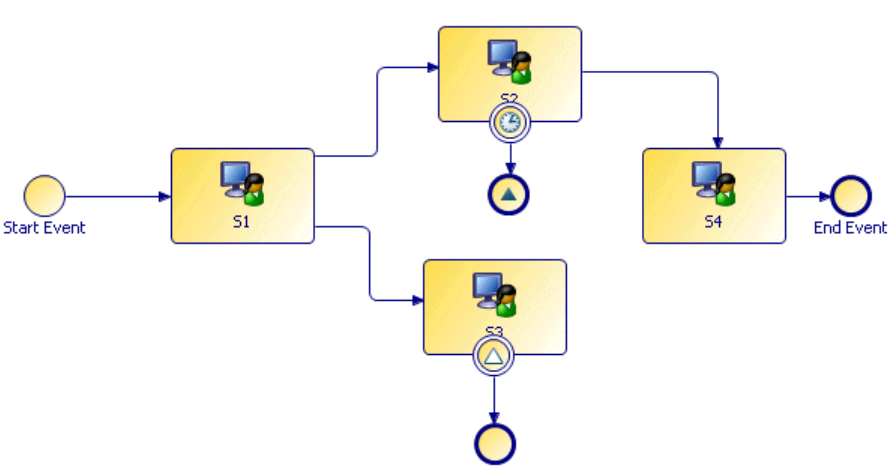
- If the previous TIBCO Business Studio process is imported into iProcess Modeler, the throw signal event will become a complex router. Although this differs from the earlier iProcess example, the two business processes are functionally identical.
- A throw signal event in a process that is being exported to the iProcess Modeler or deployed to the iProcess Engine must specify a signal name.

Example 2

In the following example, **S1** actions **S2** and **S3**. **S2** has a deadline on it, which if it expires, will withdraw **S3** and action **S4**.



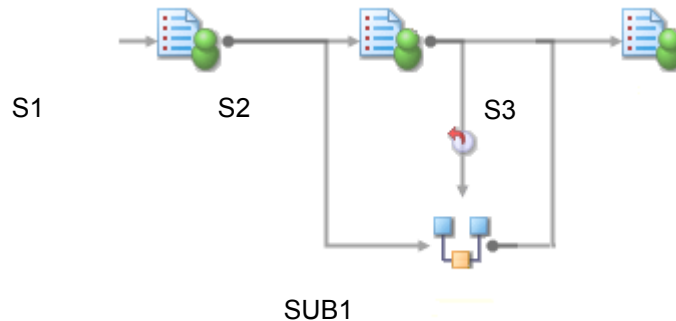
In TIBCO Business Studio, this can be implemented as follows:



If the timer event on the boundary of **S2** expires, the throw signal end event throws a signal name which is caught by the signal event on the boundary of **S3**. This withdraws **S3**.

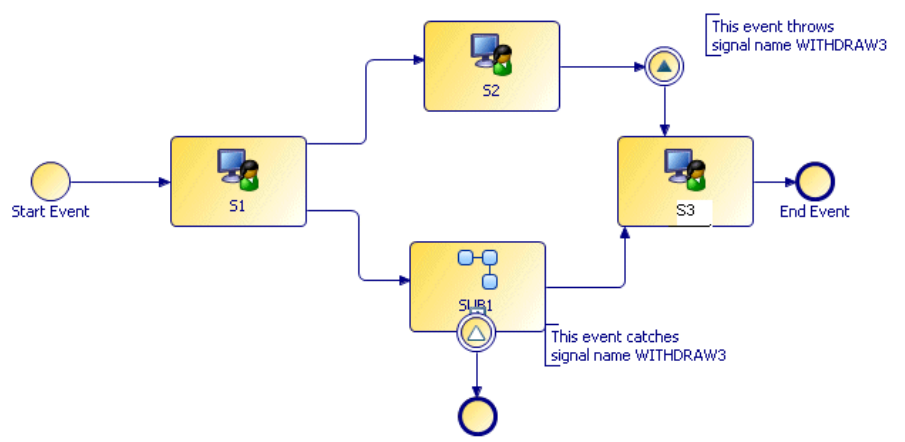
Example 3

If a sub-procedure case is withdrawn by the parent procedure case it will be closed immediately. In the diagram below, if step **S2** is released before the sub-procedure called by **SUB1** is completed, then **S2** will withdraw **SUB1** causing the sub-procedure case to be terminated prematurely.



When a sub-procedure case is terminated prematurely, no data is transferred back to the parent procedure and if the sub-procedure calls any further sub-procedures, these will also be closed.

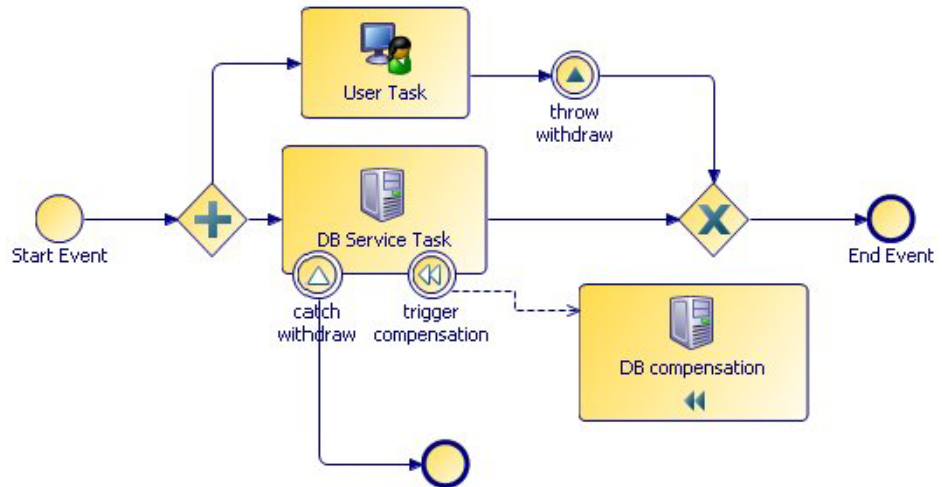
In TIBCO Business Studio, this is implemented as follows:



For more information about throw / catch signal events, see *TIBCO Business Studio Process Modeler User's Guide*.

Compensation Events

BPMN Compensation is implemented in iProcess in conjunction with a withdraw link (see [Withdraw Links on page 98](#)). For example:



In this example, if the user task executes before the database service task, the throw signal event withdraws the database service task. This triggers the compensation event. The database compensation task compensates or reverses the effect of the database service task.

Dynamic Sub-Procedures

Dynamic sub-procedures are used when a procedure that calls one of several sub-procedures at runtime, but it is not known at design time which sub-procedure will be called. In iProcess, a sub-procedure parameter template is used to provide the input and output to dynamic sub-procedures. In TIBCO Business Studio, this is accomplished by using a process interface (see *TIBCO Business Studio Modeling User's Guide* for general information about process interfaces). For more information about dynamic sub-procedures, see *TIBCO iProcess Modeler - Advanced Design*.

The following example shows how calls to dynamic sub-procedures are configured in iProcess:

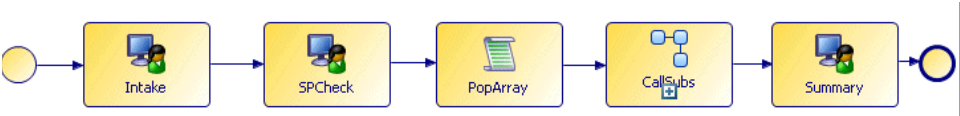


When a case of this procedure is run, the steps do the following:

1. The first step (**intake**) requires the nurse to enter some basic information about the new patient such as name, address and the medical condition(s) they have.
2. The next step (**spcheck**) relies on a supervisor to check the details and make a choice as to what else needs to be done as part of the patient's intake. They may require an X-ray or the booking of a bed.
3. The EAI Script step populates an array field called SPROCS with sub-procedure names e.g. if an X-ray is required, the sub-procedure name for the X-ray process is added.
4. The next step is the dynamic call to the sub-procedures. The call looks at the array field to find out what sub-procedures to start. In this case, just an X-ray is required so there is only one data element in the array (**xray**). The **xray** sub-procedure is started which delivers work items to the X-ray department and results in a booking for the patient. The booking date and time is returned back to the main procedure. The last work item displays a summary of the patient's admission information along with any new booking information that has been entered.

Implementing this Example in TIBCO Business Studio

Dynamic sub-procedures are implemented in TIBCO Business Studio using process interfaces. For a detailed tutorial on how to create this process, see [Tutorial 6: Creating Dynamic Sub-Processes on page 45](#). This section contains a summary of how to implement the example. The top-level process looks like this:



1. Create a process interface that specifies the start event and its input/output parameters. Each process that is to be invoked from the dynamic sub-process task must implement this interface.



Process interfaces that are used with iProcess only support basic data types.

2. Create the reusable sub-process task, and instead of a specific sub-process, select the process interface that you created.
3. Create an array data field to hold the names of the sub-processes that can be invoked.



When you have an array in a step in a procedure, you need to initialize the array before you can deploy the procedure.

4. In the sub-process call task, set the runtime identifier to the array data field that holds the list of potential sub-processes to call.
5. If potential sub-processes need to be called at different starting points, create an array data field for the task names where each sub-process should be started. This array data field can then be specified on the **Advanced** tab of the sub-process call to the process interface:

iProcess Task Properties	
Dynamic Sub-Process Task	
Is Graft Step	false
Return Status Array Field	
Runtime Sub-Process Validation	
Start Step Array Field	Field2

For more information, see [Reusable Sub-Process Call Task Validation on page 109](#)

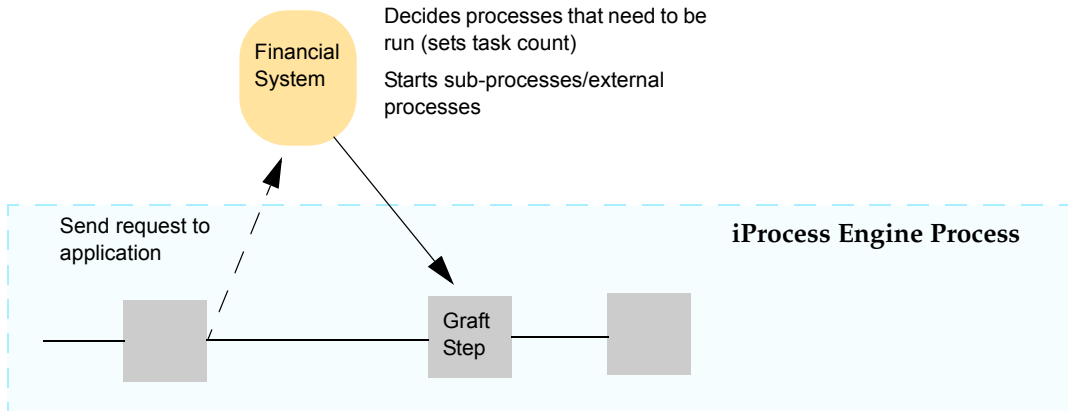
6. The script task (PopArray) is where the logic occurs to set the exact sub-process that will be selected at runtime.



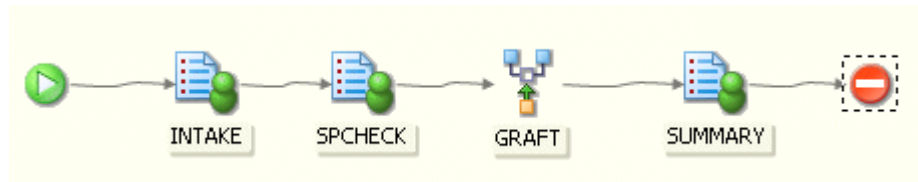
When you use process names in TIBCO Business Studio scripts, use the method **iPEProcessNameUtil.GETPROCESSNAME** to convert valid TIBCO Business Studio process names into equivalent iProcess procedure names. For example, long process names are truncated as they would be upon deployment to iProcess. This ensures that the sub-process call task works as expected upon deployment. The method takes a string and returns a string. Similarly, use the **iPETaskNameUtil** method when referring to task names.

Graft Steps

In *TIBCO iProcess Modeler - Integration Techniques*, the following procedure is used to illustrate the use of a graft step to attach several processes to an iProcess procedure when a case is run. This example is based upon using a financial application as the external application and this triggers the graft step using a TIBCO iProcess Objects call.



The example used in the previous section ([Dynamic Sub-Procedures on page 103](#)), could be implemented using a graft step as follows:



When a case of this procedure is run, the steps do the following:

1. The first step (**intake**) requires the nurse to enter some basic information about the new patient such as name, address and the medical condition(s) they have.
2. The next step (**spcheck**) relies on a supervisor to check the details and make a choice as to what else needs to be done as part of the patient's intake. They may require an X-ray or the booking of a bed.
3. The graft step signals an external application that it is waiting for the names of the procedures that need to be started. The external application populates an array field with the names of the procedure that need to be run. This is in

contrast to a dynamic sub-procedure call, where the names of the procedures that need to be run are supplied to the array field by a script task.

Implementing this Example in TIBCO Business Studio

A graft step is implemented in a similar manner to a dynamic sub-procedure (by invoking a process interface from a reusable sub-process task). In TIBCO Business Studio, a graft step is implemented as follows:



1. Create a process interface that specifies the events and their parameters that must be present in processes created using that interface. For more information about creating process interfaces, see *TIBCO Process Modeling User's Guide*.
2. Create the reusable sub-process task, and instead of a specific sub-process, select the process interface that you created.



All sub-processes that you wish to graft to the graft step at runtime must implement the process interface referenced in the reusable sub-process task.

3. Create an array data field to hold the names of the sub-processes that will be returned by the external application.



When you have an array in a step in a procedure, you need to initialize the array before you can deploy the procedure.

4. In the sub-process call task, set the runtime identifier to the array data field that holds the list of sub-processes returned by the external application.

5. On the **Advanced** tab, you can configure the reusable sub-process call task as described in [Reusable Sub-Process Call Task Validation on page 109](#):

Properties Problems Diagram Fragments

Independent Sub-Process

General

Description

Interface

Scripts

Map To Sub-Process

Map From Sub-Process

Appearance

Extended

Advanced

Property	Value
<input checked="" type="checkbox"/> BPMN Task Extensions	
<input checked="" type="checkbox"/> Duration Calculations	
<input checked="" type="checkbox"/> iProcess Task Properties	
<input checked="" type="checkbox"/> Dynamic Sub-Process Task	
Is Graft Step	true
Return Status Array Field	
<input checked="" type="checkbox"/> Runtime Sub-Process Validation	
Halt Process On Invalid Interface	false
Halt Process On Invalid Interface	false
Halt Process On Invalid Sub-Process	false

Reusable Sub-Process Call Task Validation

You can configure dynamic sub-procedures and graft steps as follows:

iProcess Property	Description
Is Graft Step	Set this property to true if you want the task to be considered a graft step.
Runtime Sub-Process Validation	<p>The settings in this category determine what happens as a result of validation in the runtime environment:</p> <ul style="list-style-type: none"> • Halt Process on Invalid Interface Sub-procedures that do not use the same process interface (in the iProcess, sub-procedure parameter template). Select this option to stop the process if iProcess finds parameters that are not in the sub-procedure parameter template being used by the dynamic sub-procedure call. • Halt Process on Invalid Interface Version Sub-procedures that use different versions of the same process interface (sub-procedure parameter template). Select this option to stop the process if iProcess finds some parameters that are not valid for the version of the template being used for this call. • Sub-procedure names are invalid Select this option to stop the process if iProcess cannot find one of the sub-procedures it needs to call.
Return Status Array Field	Select an array field that will be used to return an error return value. Refer to “Returning an Error Status” in the <i>TIBCO iProcess Modeler - Advanced Design</i> guide for more information about using arrays for this purpose.

Groups and Roles

The iProcess Modeler concept of a Role is different to that in BPMN. A BPMN role (as modelled in TIBCO Business Studio) represents a group of behaviors. For example, one individual could have the roles of second line support as well as project manager, and others can hold those roles too. In many cases there is no reason to choose one person over another if they both share the same role (for example, both are project managers). In the iProcess Modeler, a role corresponds to only one person. Therefore, one individual having the role of project manager prohibits anyone else having that role.

All types of user task participant in TIBCO Business Studio become iProcess users when exported or deployed to iProcess.

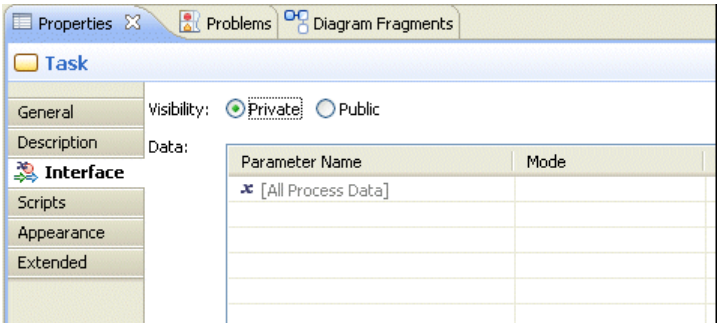
You can also use the iProcess meaning of Role as follows:

1. Add an Advanced tab entry for Participants of type Role named "Use iProcess Role semantics". This will be a Boolean and default to false.
2. On transformation to iProcess if this flag is present and true the participant name will be put in the Role Addressee field rather than Group Addressee.

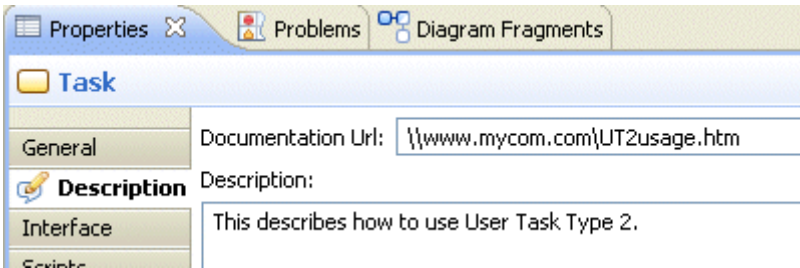
Public Steps and Events

Public events or tasks are made available to external processes or applications. For example, when a process with public events or tasks is deployed to iProcess, iProcess can publish information about case start and event trigger steps to an external process (or application) via SAL/TIBCO iProcess Objects interfaces.

In TIBCO Business Studio, you can specify whether an event or task is private or public on the **Interface** tab of the Properties view. For example:



You can then associate parameters with the event or task and specify whether the parameters are mandatory. On the **Description** tab, you can optionally provide a URL (equivalent to a usage URL).



Object Mapping

When you import a process into the TIBCO iProcess Modeler, the original objects (from the TIBCO Business Studio version) are mapped into objects that the TIBCO iProcess Modeler supports. This section describes the mapping between TIBCO Business Studio and TIBCO iProcess Modeler objects.

Project Objects

TIBCO Business Studio Object	Corresponding TIBCO iProcess Modeler Object
Project	None
Package	Upon deployment, the Package name becomes the library name.
Process	Procedure
Process Interface	I/O Parameter Template
Sub-Process (Process referenced by an reusable sub-process call or a Process that has formal parameters) Note: Embedded Sub-Processes cannot be exported to the iProcess Modeler or deployed to the iProcess Engine.	Sub-Procedure
Participant	All participants appear in the Users column in iProcess. An additional administrative task in the iProcess Engine is to create the required users and groups.

TIBCO Business Studio Object	Corresponding TIBCO iProcess Modeler Object
Data Field (Package and Process)	<p>Field. Package level data fields are created as fields in each Procedure. Process level data fields are created as fields only in that procedure.</p> <p>Text data fields must be between one and 255 characters or unspecified. If the length of a text data field is not specified, it becomes a memo field (a type of iProcess field that enables the user to enter unlimited amounts of text).</p> <p>A Date Time data field is exported as two separate date and time fields in the iProcess Engine (suffixed "_D" and "_T").</p>
Process Formal Parameter	<p>I/O parameter and field. A Date Time parameter is exported as two separate date and time fields in the iProcess Engine (suffixed "_D" and "_T").</p> <p>The mandatory flag on an process formal parameter is equivalent to the Required column in iProcess. This column is on the table that is displayed by selecting the Procedure > IO Parameters menu in iProcess when defining a sub-procedure. Using the mandatory flag means that the sub-procedure requires this parameter to always be supplied.</p>
Group	None

Activities and Tasks

TIBCO Business Studio Object	Corresponding TIBCO iProcess Modeler Object
Reusable Sub-Process	Sub-Procedure call
Service Tasks	Enterprise Application Integration (EAI) step with the appropriate type set
Script Task	<p>EAI Script step. If you plan on having someone else implement the script and export or deploy it, use the Text script type to describe the behavior of the script that you want the Solution Engineer to implement.</p> <p>You cannot export or deploy Script Tasks with the script type set to Text. For more information about implementing Script Tasks, see Working with Scripts on page 185.</p>
Task Type (None)	Complex router. Note: If the process is re-imported into TIBCO Business Studio, the complex router becomes a Gateway (this is expected and provides functionally equivalent behavior).
Receive Task	iProcess Event
User Task	Work Item Step
Reference Task	Step that is a duplicate of the referenced task. For example, if the reference task points to a user task, upon deployment the reference task is replaced with the specified user task.
User Task Participant	Work item addressee depending on the participant type (Performer Data Field = Field Addressee, all other types = User Addressee)
User Task Form URL	Formflow Form URL
User Task Parameter	Field instance on iProcess form or Formflow parameter

TIBCO Business Studio Object	Corresponding TIBCO iProcess Modeler Object
User Task Script	<p>Script object. If you plan on having someone else implement Open, Close and Submit scripts and export or deploy the process, use the Text script type to describe the behavior of the scripts that you want the Solution Engineer to implement.</p> <p>You cannot export or deploy user task scripts with the script type set to Text. For more information about implementing user task scripts, see Creating a Script for a User Task on page 199.</p>

Events

TIBCO Business Studio Object	Corresponding TIBCO iProcess Modeler Object
Start/End Event	Start/Stop
Timer Event (inflow)	iProcess Event with deadline link and auto withdraw deadline
Timer Event Script	<p>Deadline expression. If you plan on having someone else implement the deadline expression and export or deploy the process, use the Text script type to describe the desired behavior.</p> <p>You cannot export or deploy Timer Event scripts with the script type set to Text. For more information about implementing Timer Event scripts, see Timer Event Scripts on page 205.</p>
Timer Event (on Task boundary)	Defines deadline for iProcess step. In TIBCO Business Studio there are two options you can select (Withdraw Task on Timeout and Continue Task on Timeout). These are equivalent to the iProcess Withdraw flag.
Signal Event pair (throw and catch)	Withdraw link - the throw event becomes a complex router and the catch event defines the Withdraw link between the complex router and the task to which it is attached.

Gateways

TIBCO Business Studio Object	Corresponding TIBCO iProcess Modeler Object
AND Gateway (branching)	Complex Router
Gateway with one input, one conditional output, and optionally one default output	Condition with the Expression defined by the condition on the conditional Sequence Flow:

TIBCO Business Studio Object	Corresponding TIBCO iProcess Modeler Object
Parallel Gateway (multiple unconditional input)	Wait
XOR Gateway with multiple unconditional input	Complex Router

Connecting Objects

TIBCO Business Studio Object	Corresponding TIBCO iProcess Modeler Object
Association	None
Sequence Flows from a Timer Event (on an Activity)	Deadlines Link
Text Annotation	Unattached Annotation
Data Object	None

Swimlanes

TIBCO Business Studio Object	Corresponding TIBCO iProcess Modeler Object
Pool	Not mapped
Lane	Lane

Field Mapping

TIBCO Business Studio Field	Corresponding TIBCO iProcess Modeler Field	Mapping
Activity/Task Name	Step Name	Truncated to 8 characters if needed
Label	Description	First 24 characters
Label	Extended Description	Second 24 characters

TIBCO Business Studio Field	Corresponding TIBCO iProcess Modeler Field	Mapping
Label	n/a	Character 49 onwards

Chapter 4 **Configuring User Tasks**

This section of the help describes how to elaborate a user task.




For information about creating user task scripts, see [Creating a Script for a User Task on page 199](#).

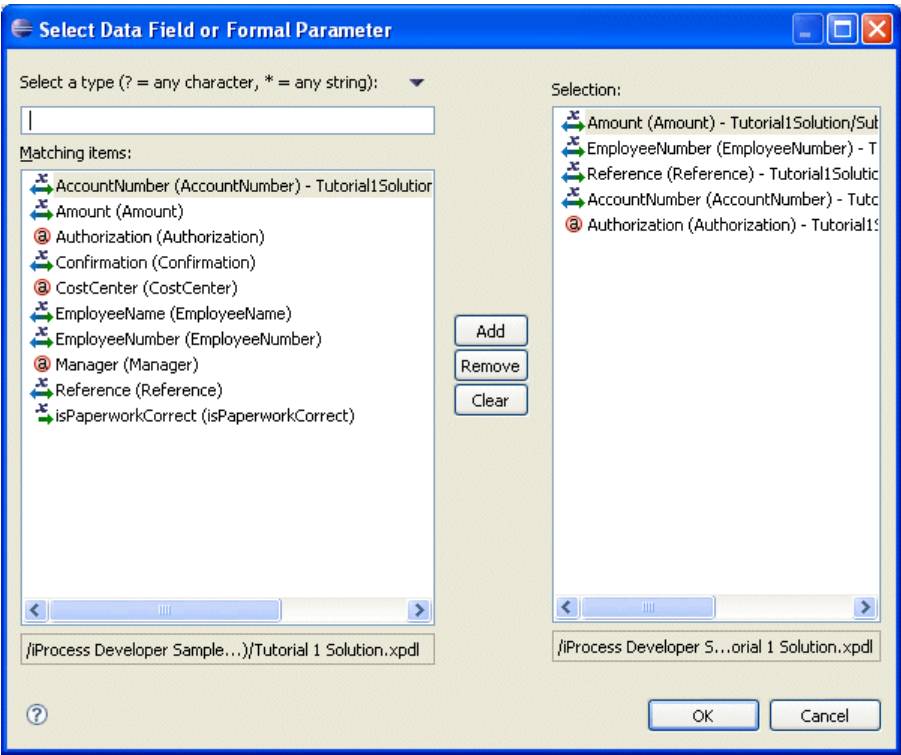
Topics

- [Adding Parameters and Data Fields, page 120](#)
- [Specifying a Form, page 122](#)
- [Using Performer Data Fields, page 124](#)

Adding Parameters and Data Fields

You can add parameters or data fields to user tasks as follows:

1. Click the user task that you want to add parameters to. In the Properties view, click the **Interface** tab.
2. Select whether you want the visibility of the Event to be private or public (see [User Task Properties on page 259](#)).
3. Click  to add new process data to the event or task.



- The wildcard ? returns all matching data fields or formal parameters. Use the * wildcard to restrict the results (for example, *2 to return all data fields or parameters ending in 2. Note that the wildcard * by itself does not return any results; it only works in conjunction with a string.
- To select several parameters in the **Select Data Field or Formal Parameter** dialog, press the **Ctrl** key and click the desired data field or parameter.



- The process data displayed depends on what type of event or task is selected. Most events and tasks can have both data fields and formal parameters associated with them, however Receive Tasks and events of type None or Message can have only formal parameters associated with them.
 - Data fields can contain an external reference to a primitive type in the business object model. For more information about creating primitive types, see *TIBCO Business Studio Business Object Modeler User's Guide*.
 - Data fields or parameters with spaces in their names cannot be used for mapping (for example, mapping to parameters in a web service).
4. The process data you select is added to the table of data. Select whether you want the data to be mandatory.



- The mandatory setting on the **Interface** tab for a formal parameter overrides the mandatory setting in the Properties view for the formal parameter. This allows complete freedom in designing the process - you can define a formal parameter as mandatory in one place in a process, and optional in another.
5. Use the space provided if you want to add an optional usage description of the process data. Selected parameters also display their mode (In, Out, or In/Out). You can change the mode of data fields by selecting from the drop-down list.
 6. Click **OK** when you have finished selecting parameters and data fields.
 7. Save the Package that contains the process.

Specifying a Form

The primary way of designing a form in TIBCO Business Studio is to use TIBCO Business Studio Forms. For more information see *TIBCO Business Studio Forms User's Guide*.

There are several options for forms on the **General** tab for a user task:

The screenshot shows the 'Task' properties dialog in TIBCO Business Studio, specifically the 'General' tab. The dialog has a sidebar with tabs: General, Description, Interface, Resource, Scripts, Appearance, Extended, and Advanced. The 'General' tab is active. It contains several input fields and checkboxes. The 'Label' field is 'User Task', and the 'Name' field is 'UserTask'. Under 'Activity Markers', there are checkboxes for 'Standard Loop', 'Multiple Instance Loop', and 'Ad-Hoc'. The 'Participants' field contains 'Worker' and has a 'Clear' button. The 'Activity Type' dropdown is set to 'User Task'. On the right side, there are three radio buttons: 'No Form URL', 'User Defined URL...', and 'Form...'. The 'Form...' radio button is selected. Below these radio buttons, there is a text field labeled 'Form:' containing 'Untitled.form' and a link labeled 'Open Form'.

Select one of the following options:

- **No Form URL** Select this option if you have not created a specific form for the user task. At runtime a standard iProcess form is created, in which required fields are marked in red, and optional fields in blue (for more information about standard iProcess forms, see the TIBCO iProcess Modeler set of guides).
- **User Defined URL** Select this option if you want to point to a specific URL that you have defined (for example, if you authored a form outside of TIBCO Business Studio). Manually enter the URL in the field that is displayed when you select this option. TIBCO Business Studio cannot validate the URL, so ensure it is correct. Upon import to iProcess Modeler, a user task with a user defined URL becomes a step with a Form type of **Formflow Form**. If you select this option but do not specify a URL, upon export to iProcess a standard iProcess form is created.

Refer to the *iProcess Workspace Browser Configuration and Customization Guide*.

- **Form** Use this option if you have created a form using TIBCO Business Studio Forms. Either automatically create a form (in which case the **Form** field is completed automatically) or browse to select a form from a **Forms** special folder. Upon deployment/import to the iProcess Engine, the task becomes a step with a Form type of Formflow Form.

About URLs

If you specify a user-defined URL, you can use either of the following types:

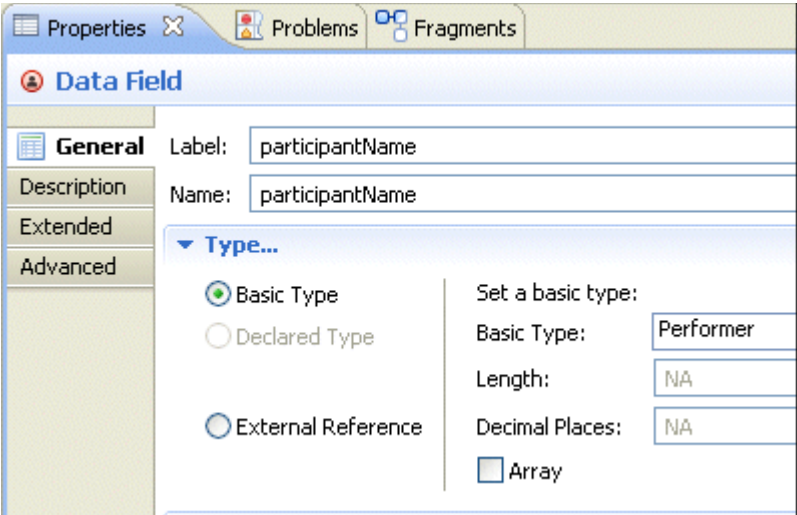
- **Relative** (the default) - for example, **JSPFormExample/JSPForm.jsp**. Specifying a relative URL assumes that browser client (either your own custom client or that provided by TIBCO), has been configured to specify the "base" URL for the form's location. For information about configuring TIBCO iProcess Workspace (Browser), see *TIBCO iProcess Workspace (Browser) Configuration and Customization*.

This has the effect that all JSPs in all processes on all nodes accessed using this client must be in the same web application. The benefit of using this type of URL is that when moving from development to user acceptance testing and then to production, the base URL (including the host name) needs to be changed in just one central place.

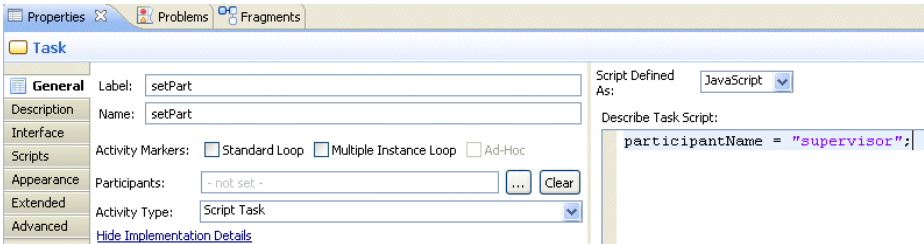
- **Absolute** - for example, **http://www.acme.com/mywebapp/myjsp.jsp**. Absolute URLs allow each step of each process to specify a separate host, application and protocol. This method limits flexibility because that URL must be constant.

Using Performer Data Fields

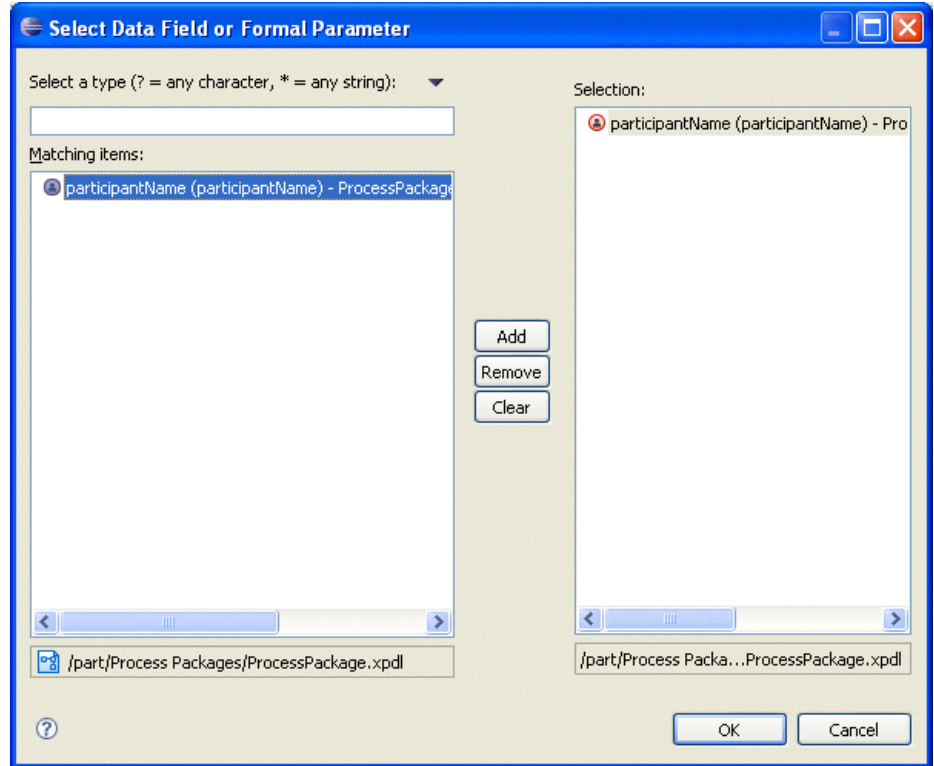
You can use a performer data field in place of a participant. The advantage of doing this is that you can dynamically assign a value to the performer data field (and therefore the participant that references it). For example, you could define a Performer data field called **participantName**:



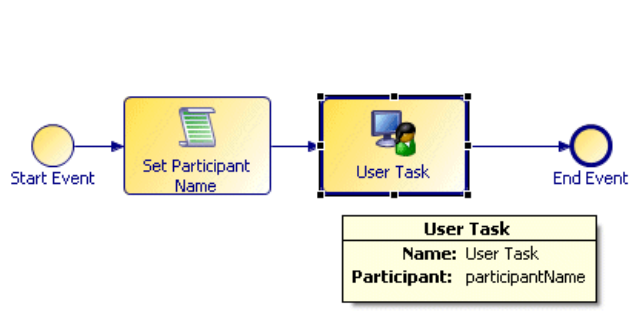
The value of this data field could then be set in a Script Task:



Then, when assigning a participant to a subsequent user task, you can select the Performer data field.



The full process looks like this:



When the process is executed, the user task will have the participant **supervisor**.

Converting a Participant into a Performer Data Field

Often a business analyst may have defined participants that while sufficient for the modeling of the process, need to be converted to performer data fields so they can be utilized as described in [Using Performer Data Fields on page 124](#).

1. Select the participant that you want to convert, and select **Convert Participant to Data Field**.
2. Note that the participant is removed from the list of participants and added to the list of data fields as a performer data field.

Chapter 5

Configuring Service Tasks

This chapter describes how to make various types of service call from a Service Task.

Topics

- [Working with Service Registries, page 128](#)
- [Working with WSDL Files, page 131](#)
- [Configuring a BusinessWorks or Web Services Service Task, page 145](#)
- [Using the Mapper, page 149](#)
- [Sending an Email, page 156](#)
- [Making a Database Call, page 160](#)
- [Calling Java Code, page 172](#)
- [Configuring POJO in TIBCO Business Studio, page 177](#)
- [Configuring TIBCO iProcess Conductor Service Tasks, page 180](#)

Working with Service Registries

If you plan to add a WSDL file to your project from a service registry, you can create a new service registry entry either before importing the WSDL file, or as part of the import process. This section describes how to add a UDDI registry before importing the WSDL file.


Adding a UDDI Registry

1. Select **New > Other**.
2. Expand **Business Modeling > Service Registry**, select **Configure Service Registry** from the list of wizards and click **Next**.
3. Enter the details of the Web Service registry:
 - **Name** - enter the name that you want to be displayed for the registry in the Service Explorer.
 - **Query Manager (Inquiry) URL** - the URL used to retrieve information about the services and businesses of the registry.
 - **Lifecycle Manager (Publish) URL** - the URL used for publishing services and businesses to the registry.
4. Click **Finish**.

Viewing a Registry

1. Select **Window > Show View > Other**.
2. Expand **Web Service Registries** and select **Registries**.
3. The Registries view opens and you should see any UDDI Registries that you have added.

Creating a Registry Search

1. Click the Add Search button () or right-click the Registry and select **Add Search**.
2. Select the type of search you want to perform (either for a business or for a service) and click **Next**.

3. Enter the service search criteria:

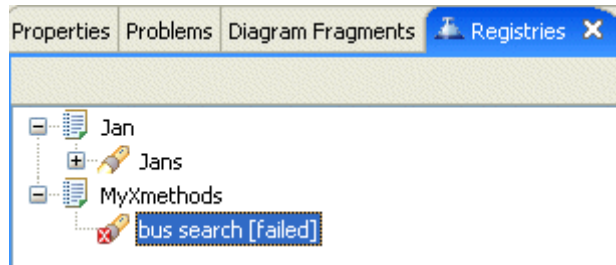
- **Name** This is the name you want displayed in the Registries view for your search.
- **Search Criteria** You can use a percent sign (%) as a wildcard to specify search criteria. For example, specifying **c%** would return all businesses or services that start with the character **c**.



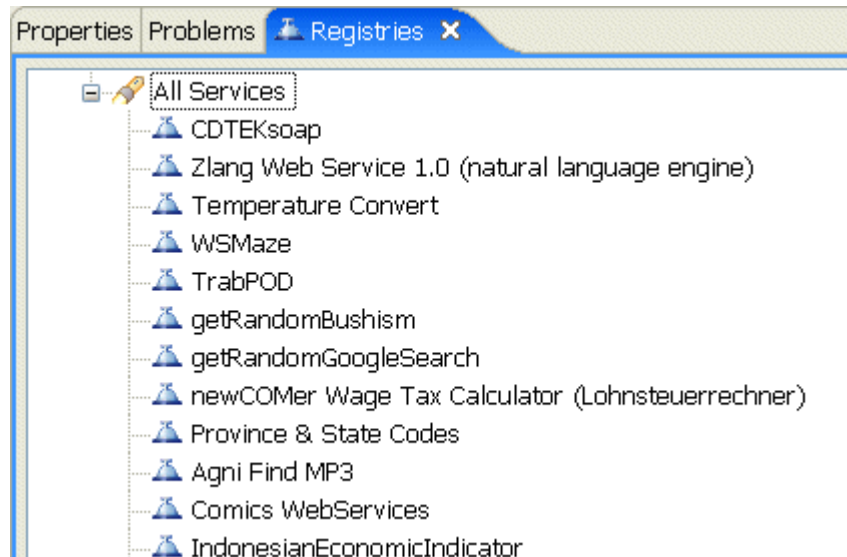
Not all registries support this wildcard syntax; however the TIBCO ActiveMatrix Service Registry does.

4. Click **Finish**.

If the search is successful, the results are displayed in the Registries view. If the search is not successful, a message is displayed and you should check the error log for more details:



When you expand the search in the Registries view, the results are displayed. For example:



Search results are preserved for subsequent browsing, but may be refreshed.

Changing the Properties of a Registry or Search

You can change the properties of a Registry or Search that you have created as follows:

1. Do one of the following:
 - Double-click the Registry or Search.
 - Right-click the Registry or Search and select **Properties**.
 - Select the Registry or Search and select **File > Properties**.
2. From the resulting **Properties** dialog, change the settings as necessary, then:
 - Click **Apply** to effect any changes you have made.
 - Click **OK** to exit the dialog.
 - Click **Cancel** to exit the dialog without applying your changes.

Working with WSDL Files

There are several options for getting a WSDL file into your project. This section describes:

- [Generating a New WSDL File](#)
- [Copying a WSDL File](#)
- [Dragging a WSDL File From the Registry](#)
- [Importing a WSDL File/Service Description](#)

WSDL File Requirements

Data Transport/Call Style

- The web service must use one of the following data transport mechanisms.
 - Simple Object Access Protocol (SOAP) requests over the Hypertext Transfer Protocol (HTTP) - (SOAP/HTTP)
 - Extensible Markup Language (XML) text using a Java Message Server (JMS) - (XML/JMS)

The main difference between using SOAP/HTTP and XML/JMS is that SOAP/HTTP uses Web Services Description Language (WSDL) source to determine how the text is sent. However, when using XML/JMS you must define your own XML schema for sending data between iProcess and an external application. Then when you import the XML schema into TIBCO Business Studio, a WSDL file is created (see [Importing a WSDL File/Service Description on page 139](#)).

- The web service call style must be either Document Literal or Remote Procedure Call (RPC) Encoded.

Parameter Mapping

- The web service must have at least one input and output parameter mapping.

- The web service can have the following types of request and response parameters:
 - Simple types
 - Arrays of simple types
 - Complex types (including complex types containing complex types)
 - Arrays of complex types

Message Exchange Patterns

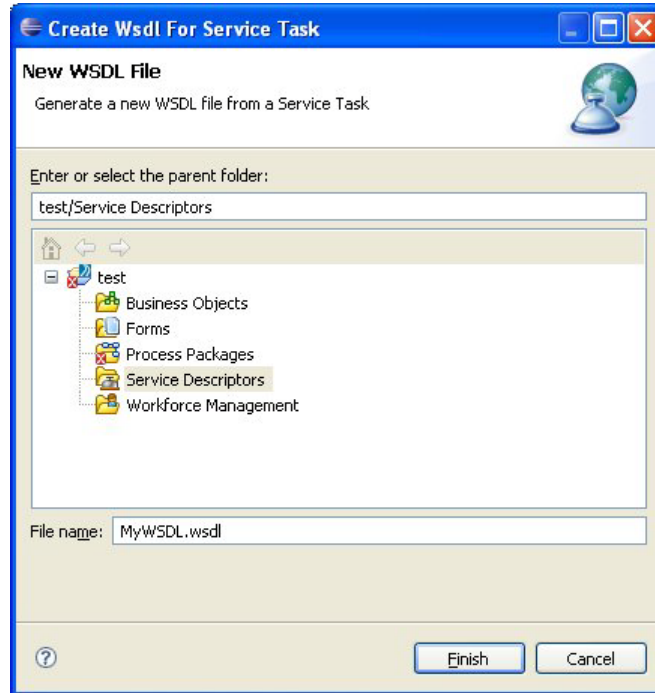
The only supported message exchange pattern is Synchronous Request/Response.

Generating a New WSDL File

If you already have a WSDL file for the service you want to call, import it as described in [Importing a WSDL File/Service Description on page 139](#). If you do not already have a WSDL file for the service you want to call, you can automatically generate one from the Properties view for the service task. The generated WSDL document is abstract (containing a port type and operation), and created with the parameters from the **Interface** tab (or all parameters if none are selected):

1. On the **General** tab in the Properties view for the service task, click **Generate**.

2. In the **Create WSDL for Service Task** dialog, select the parent folder in which to save the WSDL file, and enter a name for the WSDL file:



3. Click **Finish**. The WSDL file is created in the specified folder:



In addition, the service task is configured to reference the generated WSDL file:

Service Type:	Web Service
Operation:	<div>Select Clear Import WSDL Generate</div>
Port Type:	ProcessPackageProcess
Operation Name:	ServiceTask
Port Name:	
Service Name:	
Transport:	SOAP over HTTP
Endpoint Resolution:	
WSDL:	<div><input type="radio"/> Use local: <input checked="" type="radio"/> Use remote:</div>
Location:	MyWSDL.wsdl
Endpoint Name:	
Security Profile:	



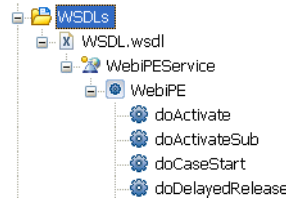
- Only concrete WSDL files can be deployed to iProcess. Use another software tool to modify the abstract WSDL file before deploying it.
- Generation of a WSDL file from a service task is performed only when you click the **Generate** button. This means that after generating the WSDL file, any changes to the parameters of the task will not be reflected in the WSDL file unless you explicitly regenerate the WSDL file. This is different from generated services created by TIBCO Business Studio for receive tasks or message events, where the WSDL files are generated every time the process is saved.
- The Endpoint name must be resolved (see [Web Services Endpoint Resolution on page 147](#)).

Copying a WSDL File

If you already have a WSDL file, either in the file system or received via e-mail, you can copy and paste it into a folder in the project. You can also designate a special folder to hold the WSDL file (this allows you to expand the WSDL file in the Project Explorer to see the operations available).

1. Locate the WSDL file, select it and press **Ctrl + C** to copy it.

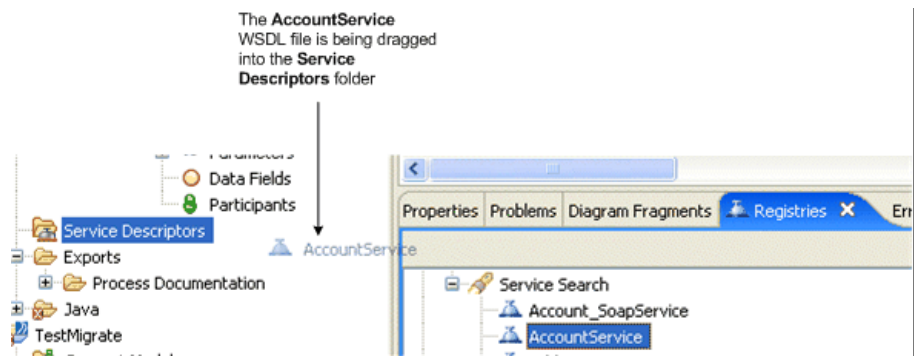
2. In the Project Explorer select the folder in the project where you want to copy the WSDL file, the press **Ctrl + V**. The WSDL file is pasted into the correct location. If you have not yet created a folder, you can do so as follows:
 - a. Right-click where you want to create the folder and select **New > Folder**.
 - b. Enter the parent directory and folder name, then click **Finish**.
 - c. Select the newly-created folder, right-click and select **Special Folders > Use as Service Descriptors Folder**. This enables you to expand the WSDL file in the Project Explorer. For example:



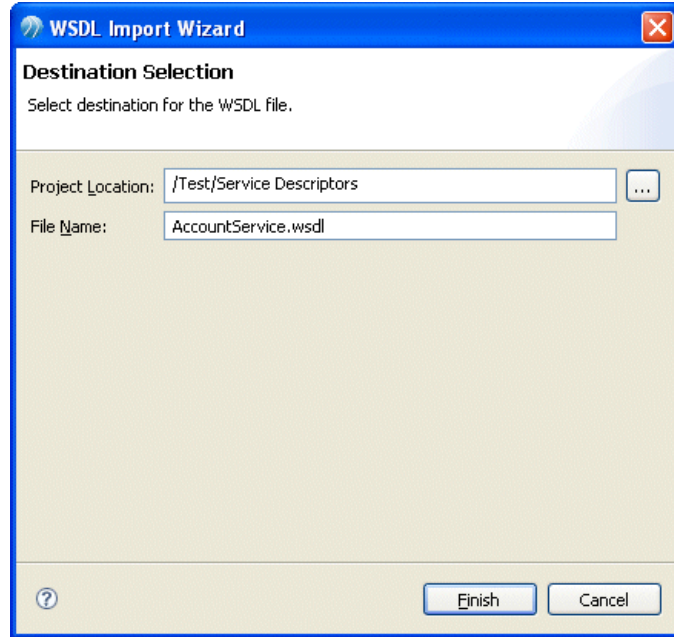
Dragging a WSDL File From the Registry

You can drag WSDL files returned from a Registry search and drop them into a **Service Descriptors** folder in the Project Explorer as follows:

1. Drag the WSDL file to the **Service Descriptors** folder.



2. When you do this, the WSDL Import wizard is launched with the **Project Location** set to the location where you dropped the WSDL file.



3. Click **Finish**. The WSDL file is imported.

Creating a JMS Server (TIBCO BusinessWorks Only)

If you are importing a WSDL file using a BusinessWorks live link, you must first create a JMS server as described in this section.

1. Right-click **Deployment Servers**, select **New > Server**.
2. Enter a server **Name** (to identify the server within TIBCO Business Studio).
3. Select **JMS Server** as the **Runtime** and click **Next**.

4. The following dialog is displayed:

New Server

Runtime Server Parameters

✖ "Target Queue Name" cannot be empty.

Runtime Server Parameters:

Host: localhost

Port: 7222

JNDI Name: QueueConnectionFactory

Target Queue Name:

Username:

Password:

☐ Save password

⚠ Saved passwords are stored on your computer in a file that is difficult, but not impossible, for an intruder to read.

Test Connection

Repository:

Repository Type: Workspace

< Back Next > Finish Cancel

Enter the runtime server parameters as follows:



When using a BusinessWorks live link, the JMS provider information can be found in either of the following locations:

- In the JMS Administration Utility (from UNIX, navigate to the **\$SWDIR\jmsadmin** directory and enter the **jmsadmin.sh** command, or from Windows, click **Start > TIBCO iProcess Server (Windows) > JMS Administration Utility**).
- In the iProcess Service Agent area of the **BusinessWorks Step Definition** dialog when defining an iProcess BusinessWorks step in an iProcess Engine procedure.

For more information see the *TIBCO iProcess BusinessWorks Connector User's Guide*.

- **Host** Use **localhost** if the server is on your local machine; otherwise enter the machine name or IP address of the computer where the JMS server is installed.
 - **Port** The port number that the JMS server uses for communication. By default the Port is **7222**.
 - **JNDI Name** The JNDI name of the JMS server.
 - **Target Queue Name** Enter the JMS destination queue for the TIBCO Enterprise Message Service that handles messages from TIBCO BusinessWorks and the TIBCO iProcess Engine. Note that you cannot specify queue names with hyphen characters (-); if you do TIBCO Business Studio will be unable to retrieve the WSDL file.
 - **Username** Username for connecting to the server.
 - **Password** Password that corresponds to the specified username.
 - **Save Password** Select this checkbox to avoid entering the password each time you connect.
 - **Repository Type** Workspace is selected; you cannot change this setting.
5. Click the **Test Connection** button to verify the details you entered. Click **Finish**. The new server is created and displayed in the Project Explorer.

Connecting to a JMS Server (TIBCO BusinessWorks Only)

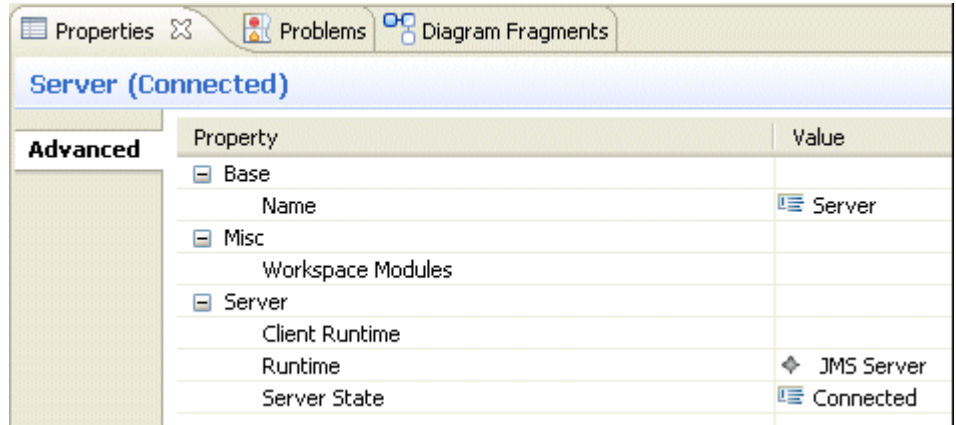
After creating a JMS server, you must connect to it before you can import a WSDL using BusinessWorks (see [Importing a WSDL File/Service Description on page 139](#)) You can connect to the JMS server you have created as follows:

1. In the Project Explorer, expand **Deployment Servers**.
2. Right-click the server name of the server you are connecting to and select **Connect**.



The username and password you entered when you created the server is authenticated on the deployment server to prevent you from deploying a process to a server which you do not have authorization to use.

- When you have connected, the Properties view for the server displays Connected as the Server State:



If you encounter any problems connecting to a Server, check the error log by selecting **Help > About TIBCO Business Studio**. From the resulting dialog, click **Configuration Details** then click **View Error Log**. Also restart TIBCO Business Studio before attempting to reconnect to the Server.

Importing a WSDL File/Service Description

There are several ways to import a WSDL file into a project:

- from a file or URL
- from a UDDI registry
- from TIBCO BusinessWorks



If you are using XML/JMS, you can also import XML service descriptions for both the input and output operations. TIBCO Business Studio then creates a WSDL file based on the XML service descriptor that you specified.

To import a WSDL file, do the following:

- Right-click the **Service Descriptors** folder into which you want to import the WSDL file and select **Import > Service Import Wizard**.

2. Select one of the following import methods:
 - **Descriptor for XML over JMS** Use this method if you are using the XML/JMS transport mechanism, and have created an XML or XSD document that describes the expected input and output to the service.
 - **Import from a File** Use this method to browse the file system for the WSDL file.
 - **Import from a URL** Use this method to specify a URL that resolves to the location of the WSDL file.
 - **Import from a UDDI Registry** Use this method to select a WSDL file from a UDDI registry.
 - **Import from BusinessWorks 5.3+** Use this method to import a WSDL file using the BusinessWorks live link feature.



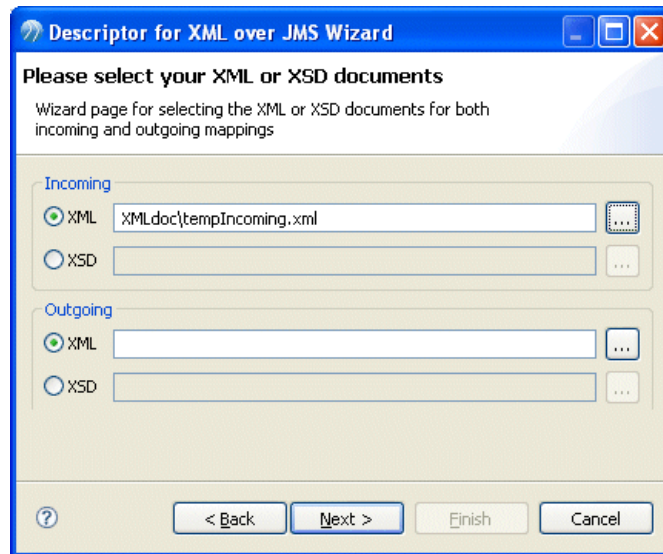
To use the TIBCO BusinessWorks live link invocation method, the iProcess Engine Service Agent must be running and the JMS provider transport information must be configured. For more information, see the *TIBCO iProcess BusinessWorks Connector User's Guide*.

Click **Next** and proceed as follows:

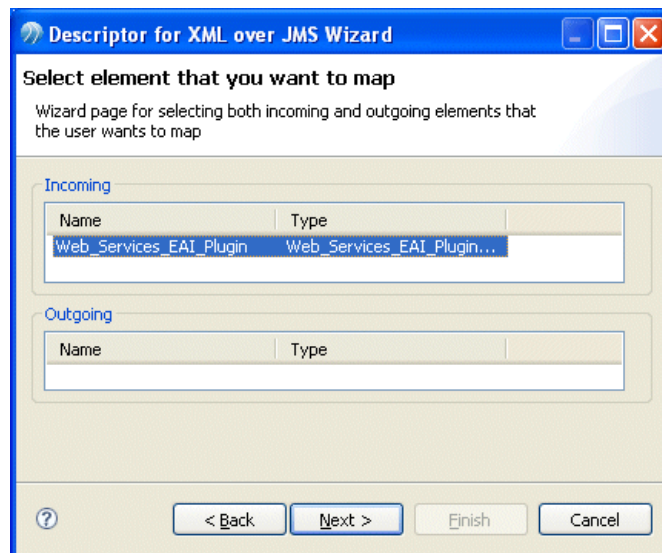
- If you chose **Descriptor for XML over JMS**, continue with [Importing an XML/XSD Service Description on page 141](#).
- If you chose **Import from a File** or **Import from a URL**, continue with [Importing From a File or URL on page 142](#).
- If you chose **Import from a UDDI Registry**, continue with [Importing From a UDDI Registry on page 143](#).
- If you chose **Import from BusinessWorks 5.3+**, continue with [Importing From BusinessWorks 5.3+ on page 143](#).

Importing an XML/XSD Service Description

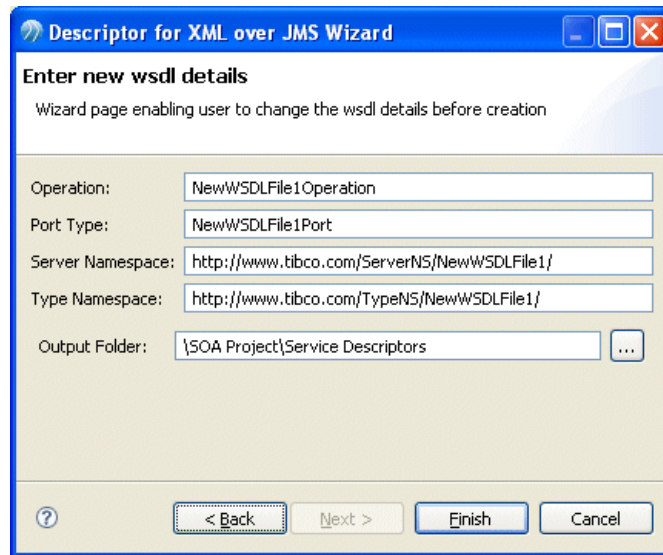
1. Browse to select the location of the XML or XSD documents that describe the input and output operations and click **Next**.



2. Select elements that you want to include in the WSDL file and click **Next**.



- Review the WSDL file details, make any necessary changes, and click **Finish**.



There may be slight delay while the WSDL file is generated in the specified output folder.

Importing From a File or URL

- Depending on the option that you chose in the previous step, do the following:
 - If you chose **Import from a File**, browse to specify the **Location** of the WSDL file.
 - If you chose **Import from a URL**, enter the URL for the WSDL file.
 Click **Next**.
- Browse to select the **Project Location** (the folder in your project where you want to store the WSDL file), and if necessary change the name of the WSDL file. Select the **Overwrite existing resources** checkbox if you want to replace any existing WSDL files with the same name.
- Click **Finish**. If the WSDL file is located remotely, there may be slight delay while the WSDL file is imported.

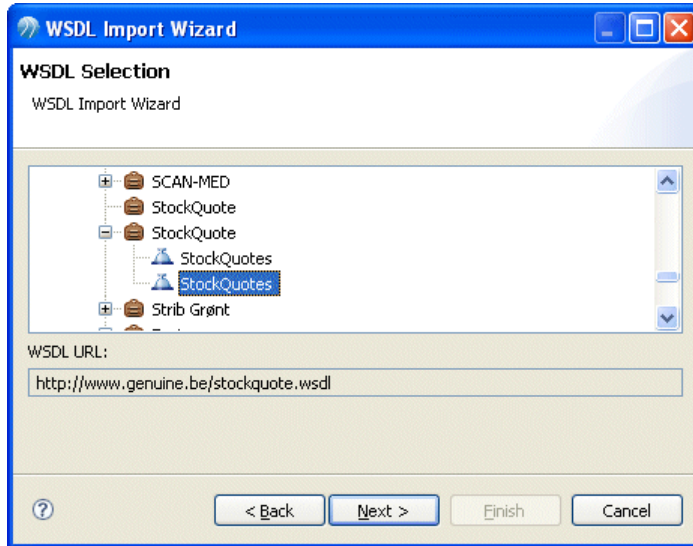
Importing From a UDDI Registry

1. If you chose **Import from a UDDI Registry**, the dialog lists any existing UDDI registries that you have added (see [Adding a UDDI Registry on page 128](#)).



To add a new registry, right-click in the blank area of the dialog and select **Add Registry**. You can also add registry searches in this dialog using the right-click menus.

2. Expand a registry and select a WSDL and click **Next**. For example:

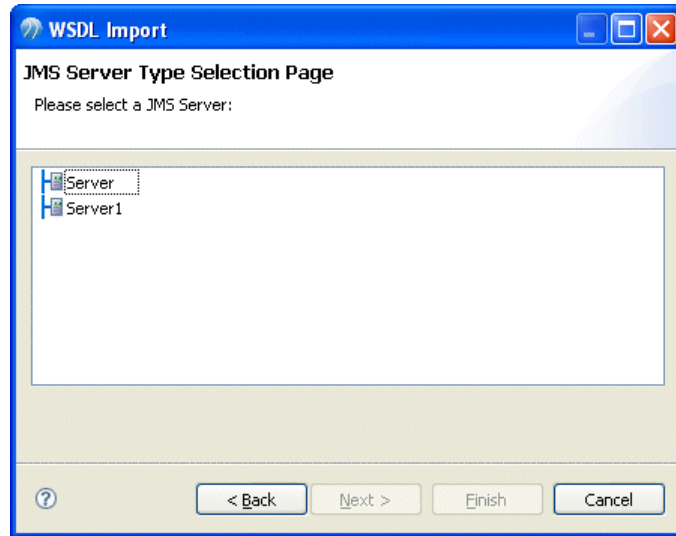


3. Browse to select the **Project Location** (the folder in your project where you want to store the WSDL file), and if necessary change the name of the WSDL file. Select the **Overwrite existing resources** checkbox if you want to replace any existing WSDL files with the same name.
4. Click **Finish**. There may be slight delay while the WSDL file is imported.

Importing From BusinessWorks 5.3+

1. If you chose **Import from BusinessWorks 5.3+**, ensure that you have created a JMS server and are currently connected to it (see [Creating a JMS Server \(TIBCO BusinessWorks Only\) on page 136](#) and [Connecting to a JMS Server \(TIBCO BusinessWorks Only\) on page 138](#)).

2. The following dialog allows you to select a JMS server that you have previously configured.



Select a server and click **Next**.

3. The JMS server parameters are displayed. Click **Next**.
4. In the **Destination Selection** dialog, select the **Service Descriptors** folder in the project into which you want to import the WSDL file. If the WSDL file already exists in the location you select, you can overwrite it by selecting the **Overwrite existing resources** checkbox.
5. Click **Finish**. There may be slight delay while the WSDL file is imported.

Validating a WSDL File

In addition to the validation provided by TIBCO Business Studio, there is additional validation available from the Eclipse Web Tools Platform (WTP). To use this additional validation, do one of the following:

- Right-click the WSDL file and select **Validate**.
- Highlight the project, right-click and select **Properties**. In the resulting dialog, click **Validation** and select the **Add Validation Builder to Project** check box.

Configuring a BusinessWorks or Web Services Service Task

You can associate a WSDL file with a Service Task. You can then specify that the Service Task performs a web services operation at runtime.

1. Select the Service Task.
2. On the Properties view for the Service Task, on the **General** tab, select either **BW Service** or **Web Service**.



When configuring a BusinessWorks service task, you must ensure that the required JMS destination has been created in iProcess. To do this, use the JMS Administration Utility (from UNIX, navigate to the `$SWDIR\jmsadmin` directory and enter the `jmsadmin.sh` command, or from Windows, click **Start > TIBCO iProcess Server (Windows) > JMS Administration Utility**).

For more information see the *TIBCO iProcess BusinessWorks Connector User's Guide*.

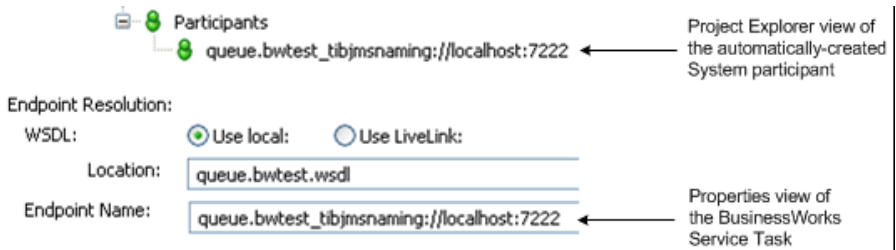
3. Click **Select** to select an operation. This opens the **Operation Picker** dialog. Depending on whether you selected an abstract or concrete WSDL file, the **Port Type**, **Operation Name**, **Port Name**, **Transport** and **Service Name** are populated automatically.



- If the operation you want to select is not available in the operation picker for a Web Services Service Task, the most likely cause is that the WSDL file is incorrect. Either contact the person that provided the WSDL file or examine it in the WSDL editor. A common cause of this problem is that the **targetNamespace** does not match the SOAP address. Also, ensure that the WSDL file is located in the special **Service Descriptors** folder.
- If you associate a WSDL file with a Service Task and subsequently delete the WSDL file from the **Service Descriptors** folder in the Project Explorer, the association between the WSDL file and the Task is broken. If you modify the process, a validation rule is run and an error message informs you of this. However, if you do not modify the process, there is no way of knowing that the association has been broken.

Additionally, if you are creating a BusinessWorks Service Task, when you select an operation, TIBCO Business Studio automatically creates a System

participant name comprised of the JMS queue name and JMS provider URL, and populates the endpoint name with the System participant. For example:

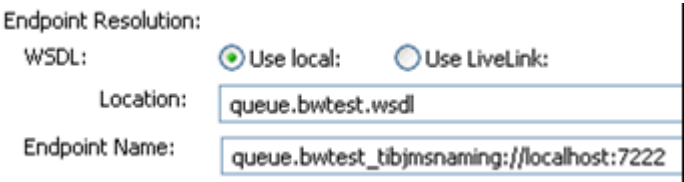


- 4. If you selected an abstract WSDL operation, you can select the **Transport**, either **SOAP/HTTP** or **XML/JMS**. Note, however that an abstract WSDL file cannot be deployed to iProcess.
- 5. Specify the **Endpoint Resolution** for the WSDL file depending on the type of step you are creating:
 - [BusinessWorks Endpoint Resolution on page 146](#)
 - [Web Services Endpoint Resolution on page 147](#)
- 6. Continue by either clicking the **Advanced** tab and configuring the parameters as described in [Service Task Properties \(Web Service/BusinessWorks Service\) on page 261](#), or by creating auditing scripts (see [Creating Audit Scripts on page 202](#)).

BusinessWorks Endpoint Resolution

Do one of the following:

- Specify a local WSDL endpoint. For example:



With the **Use local** option, you must ensure that the **Endpoint Name** matches the JMS destination that was created in iProcess, renaming the System participant if necessary.

- Use a WSDL file that will be obtained in iProcess using a BusinessWorks live link by selecting **Use LiveLink**, and specifying a System participant as a

logical **Endpoint Name**. At deployment, you must map the logical Endpoint Name to an actual endpoint in the runtime environment. For example:

Endpoint Resolution:

WSDL: ☐ Use local: ☒ Use LiveLink:

Location:

Endpoint Name: ... 



(Export only) If you want to obtain the WSDL file using a BusinessWorks live link in the runtime environment, note the following:

- If the process has the **Modeler** destination component included in the selected destination environment, an error is generated in the Problems view, and you cannot export the process.
- If the process has the **Engine** destination component included in the selected destination environment, the export of the Service Task properties may not be correct (no validation rules in TIBCO Business Studio prohibit this type of export).

Web Services Endpoint Resolution


Do one of the following:

- Specify a local WSDL endpoint. For example:

Endpoint Resolution:

WSDL: ☒ Use local: ☐ Use remote:

Location:

Endpoint Name: ... 

- Use a remote WSDL file by selecting **Use remote** and either specifying a System participant as a logical **Endpoint Name**, or specifying an explicit URL for the WSDL file. If you use a logical **Endpoint Name**, at deployment, you

must map the logical Endpoint Name to a URL alias in the runtime environment. For example:

Endpoint Resolution:	
WSDL:	<input type="radio"/> Use local: <input checked="" type="radio"/> Use remote:
Location:	This is taken from the Alias URL at Run-time
Endpoint Name:	iPE URL alias
Security Profile:	



(Export only) If you want to obtain the WSDL file with a URL alias defined in the runtime environment, note the following:

- If the process has the **Modeler** destination component included in the selected destination environment, an error is generated in the Problems view, and you cannot export the process.
- If the process has the **Engine** destination component included in the selected destination environment, the export of the Service Task properties may not be correct (no validation rules in TIBCO Business Studio prohibit this type of export).

Security Profiles

If you are calling a local, statically-defined web service (with the **Use local** option) or a remote WSDL file specified by an explicit URL, you can use a security alias in the runtime environment. To do this, specify a logical **Security Profile** name in TIBCO Business Studio. At deployment, you must map the logical Security Profile name to an actual Security Profile alias in the runtime environment.

This option is not available if you are obtaining a remote WSDL file from a URL alias (by using a System participant as a logical **Endpoint Name**). In that case, configure the security profile using the Security Profile Administrator provided with the TIBCO iProcess Web Services Plug-in.

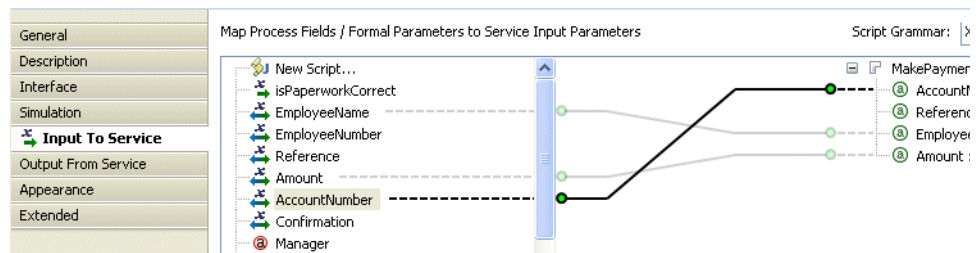


For more information about creating Security Profile aliases using the TIBCO iProcess Web Services Plug-in, see *TIBCO iProcess Web Services Plug-in User's Guide*.

Using the Mapper

This section describes how to use the mapper to map parameters to and from the service.

1. Click the **Input to Service** tab. On the left of the tab are the parameters and data fields and on the right are the formal parameters that the web service expects.
2. Expand the web service. Highlight a parameter on the left and drag the pointer from the parameter to the formal parameter on the right to create a mapping. For example:



- You cannot map data fields or parameters with spaces in their names.
- When passing negative values from iProcess to a web service or POJO step, one space is reserved for the negative sign. For example, a service that expects a decimal number of length 3 with 2 decimal places, allows numbers from -99.99 to 999.99.
- If you attempt to map data fields or parameters to parameters of a different type (for example, mapping a Date field to an Integer), an error is generated in the Problems view.

3. Complete the output mapping in a similar fashion using the **Output From Service** tab.



Web service operations that return more than one parameter are not supported in the **iProcess Engine/iProcess Modeler** Destination Environments.

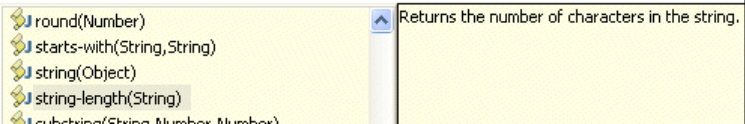
4. Save the Package that contains the process.

Array to Array Mapping

You can perform array to array mappings. To do this, you must first have a server that is i11.3 (or later). You must then configure the iProcess engine component version in the destination, as the default does not allow this mapping. Select **Window >Preferences**, go to the **Destinations** page and select the iProcess engine component version "**11.3+**"

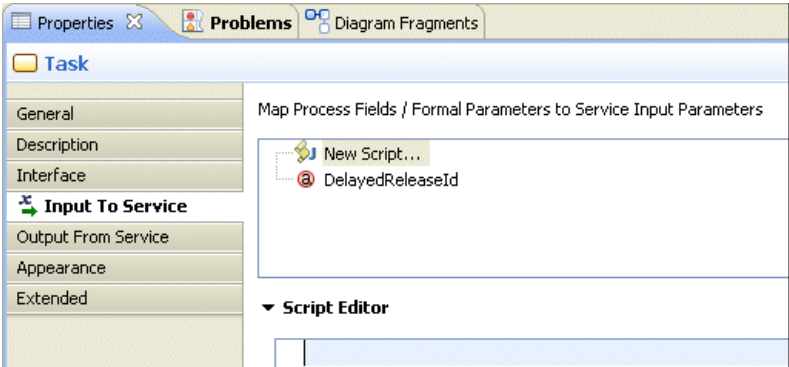
Applying Scripts to a Mapping

You can use any of the following to modify a mapping:

- XML Path Language (XPath) expressions
- Extensible Stylesheet Language Transformations (XSLT)
- Scripts that you enter are validated and any errors are displayed in the Problems view. You can also use content assist by pressing **Ctrl+Space**. For example:
- You cannot map an array to an array using XPath. In this case you will need to use XSLT.

To add a script to a mapping, do the following:

1. In either the **Input To Service** or **Output To Service** tab, click **New Script** and position the cursor in the Script area:



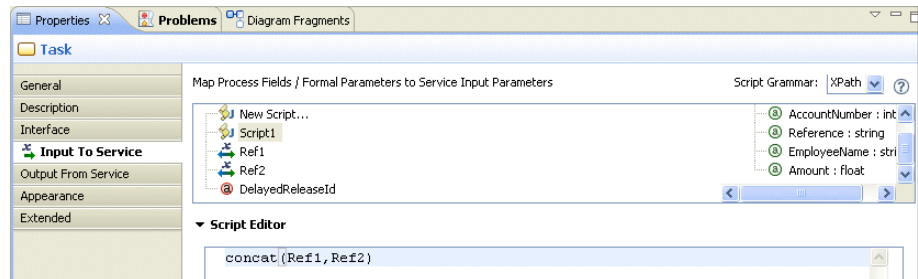
2. When you begin typing, a script called **Scriptn** is created.

3. Enter your script as follows:
 - To apply an XPath expression to a mapping, select **XPath** from the **Script Grammar** drop-down list, and enter the expression in the Editor section of the window. The expression is limited to a single line.
 - To apply an XSLT instead of selecting mappings, select **XSLT** from the **Script Grammar** drop-down list. You then have the option of either specifying a URL where the XSLT is located, or pasting the XSLT into the Editor area of the window.

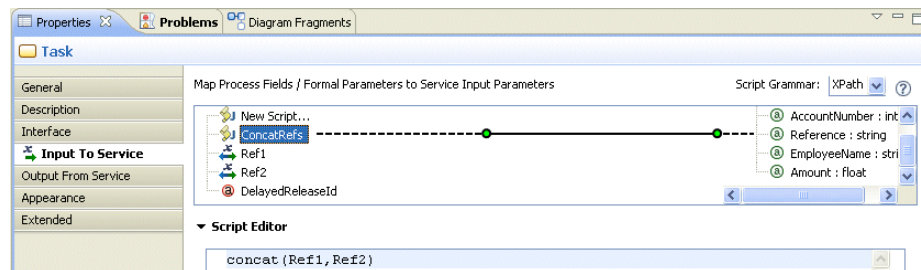


If you create XPath mappings for a process with an **iProcess** destination selected, if you can change the script grammar to XSLT, the script is converted to XSLT.

For example:



4. Give the script a meaningful name by right-clicking the script name, and selecting **Rename**.
5. Map the script by dragging from the left side of the mapper. For example:



Deleting Mappings

To delete a mapping, click the mapping and select **Delete mappings**.



Changing Mappings

To change a one-to-one mapping, drag to end of the mapping to the new parameter.



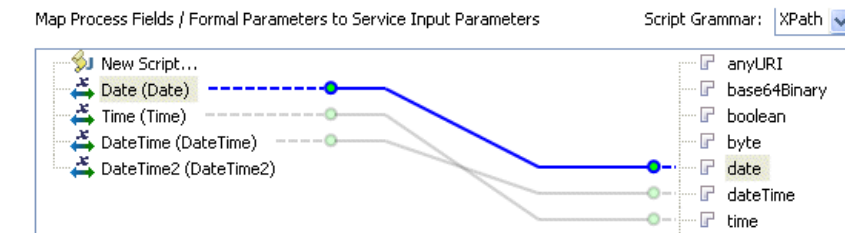
You cannot use this method when several parameters are mapped to a single parameter. If that is the case, delete the mapping and create a new one.

Mapping Date and Time Parameters

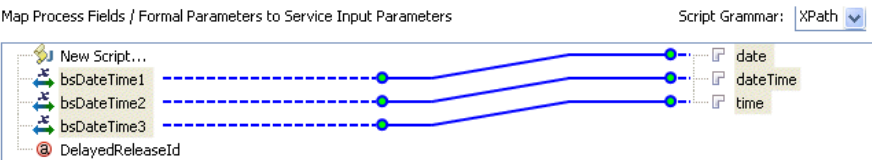
When using the **Input To Service** and **Output From Service** tabs on a web services service task, use the guidelines listed in this section.

Input Mapping

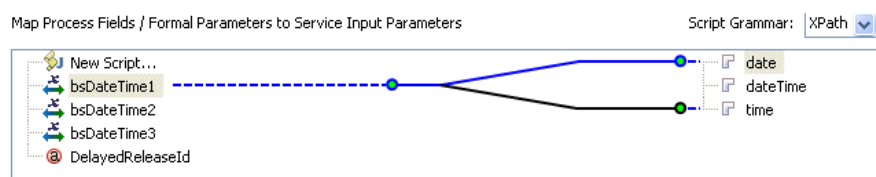
TIBCO Business Studio provides Date, Time, and Date Time types of formal parameter that can be mapped to the corresponding service input parameters as follows:



The following example shows mapping TIBCO Business Studio Date Time formal parameters to a web service Date, Time, and Date Time input parameters. All of these mappings are valid:



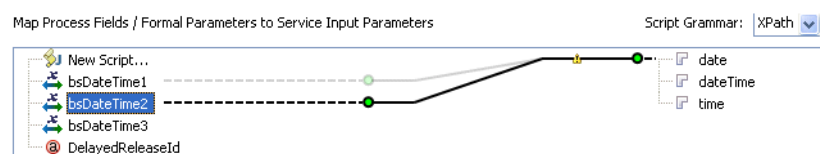
You can also map a TIBCO Business Studio Date Time formal parameter to a web service Date and Time input parameter as shown in the following example. In this case, the Date Time is split into a Date and Time in the web service:



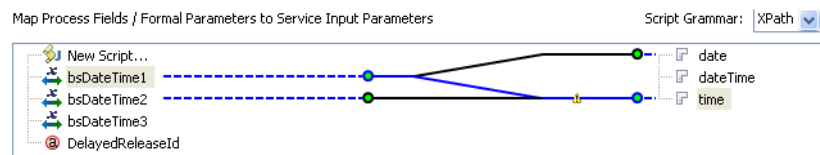
Unvalidated Mappings

There is no validation to prevent you from creating the following mappings, however they may not produce the desired results. If you do create such a mapping, BPMN and destination-specific warnings are displayed:

- Combining two Date Time Fields and mapping them to either a Date, Time, or Date Time in the web service:



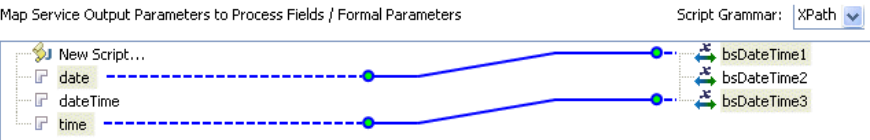
- Although you can split a TIBCO Business Studio Date Time Field and map it to a web service Date and Time, you cannot then map another TIBCO Business Studio Date Time Field to the web services parameters that have already been mapped:



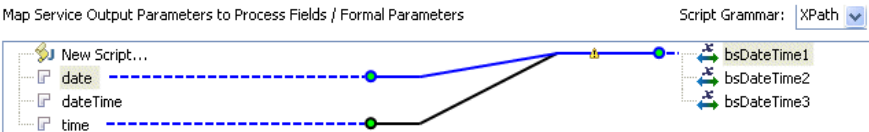
In this case, the mapping of **bsDateTime2** is invalid.

Output Mapping

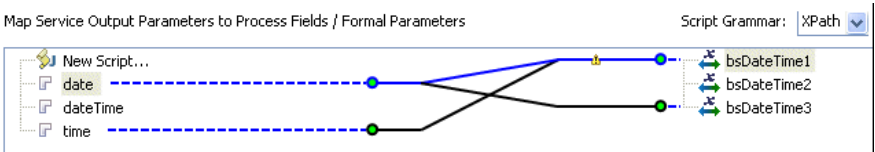
You can map a web service Date, Time, or Date Time actual parameter to a TIBCO Business Studio Date Time as follows:



You can combine a web service Date and Time into a single TIBCO Business Studio Date Time, but the order is significant – the Date must be mapped before the Time:



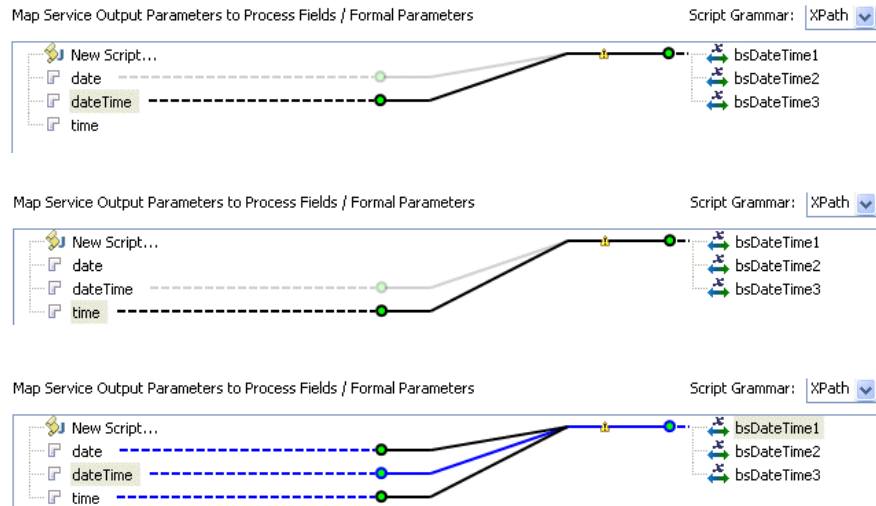
The following mapping is also valid:



The Date and Time are combined into **bsDateTime1**, and the Date is also inserted into the Date part of **bsDateTime3**.

Invalid Mappings

You can combine a web service Date and Time and map to a TIBCO Business Studio Date Time field, however you cannot combine a web service Date and Date Time, Time and Date Time, or Date Time and Date Time as shown in the following examples:



Sending an Email

A Service Task can be configured to send an email message as follows:

- 1. Select the Service Task, then on the **General** tab of the Properties view for the Service Task, select the **E-Mail** option from the **Service Type** drop-down list:

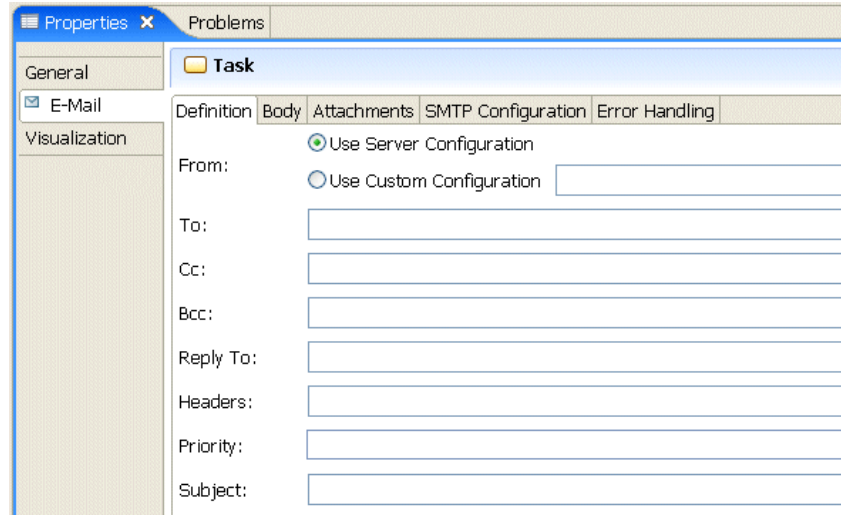
The screenshot shows a configuration window for an E-Mail service task. It contains the following fields:

- Service Type:** A drop-down menu with "E-Mail" selected.
- To:** A text field containing the placeholder "%Manager%".
- Subject:** An empty text field.
- Body:** A large, empty text area with a vertical scrollbar on the right side.

Below the form, there is a blue hyperlink labeled [More Details...](#).


- 2. Enter an email address for the recipient in the **To:** field, a subject and the body for the message. This is the minimum configuration necessary to send an email message. For further options, click **More Details** on the **E-Mail** tab and continue to specify further parameters.

3. On the **E-Mail** tab, you can specify further parameters for the **Definition** of the message:



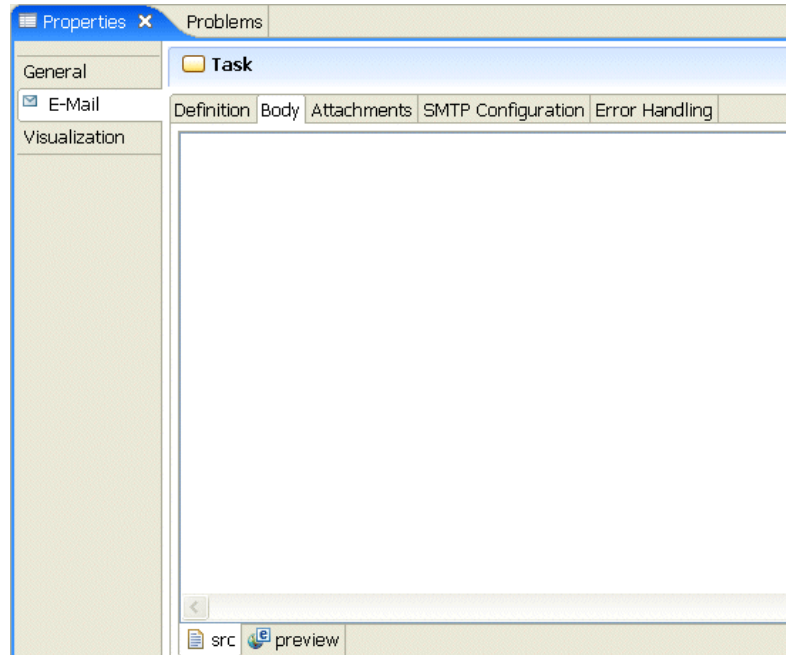
The screenshot shows the 'Properties' dialog box for a task named 'Task'. The 'E-Mail' tab is selected in the left sidebar. The 'Definition' sub-tab is active, showing fields for 'From:', 'To:', 'Cc:', 'Bcc:', 'Reply To:', 'Headers:', 'Priority:', and 'Subject:'. At the top of the 'Definition' sub-tab, there are two radio buttons: 'Use Server Configuration' (selected) and 'Use Custom Configuration' (unselected). The 'Use Custom Configuration' option is followed by a text input field.



All of the parameters on this tab can be specified using a data field or parameter. Click the  button to select the data field or parameter.

- **From:** Either **Use Server Configuration** to use the address of the server from which the email is sent or **Use Custom Configuration** to specify a data field or parameter.
 - **To:** Specify the recipient of the email either as an explicit email address or by selecting a data field or parameter.
 - **Cc:** Specify the recipients to whom you want to send a copy of the email either as explicit email addresses or by selecting a data field or parameter. Their email address is visible to other recipients of the email.
 - **Bcc:** Specify the recipients to whom you want to send a copy of the email either as explicit email addresses or by selecting a data field or parameter. Their email address is not visible to other recipients of the email.
 - **Reply to:** Use this parameter to specify a different email address to which recipients of a message can reply. Alternatively, select a data field or parameter.
 - **Headers:** Use this parameter to specify additional information in the header of the email.
 - **Priority:** Select a priority from the drop-down list (**Normal**, **High** or **Low**) or select a data field or parameter.
 - **Subject:** Select the Subject line for the email message or select a data field or parameter.
4. Click **Body** to specify the main text of the message. At the bottom of the text area are two buttons, **src** and **preview** which allow you to alternate between

viewing the source of the body text and previewing how it will look to the recipient.



5. Click **Attachments** to specify a document to be attached to the message:
 - **Field Contents:** Use this option to attach the contents of a parameter or data field to the email message.
 - **Files:** Use this option to browse the file system and attach a file to the email message.
6. Click the **Scripts** tab and add any audit scripts you require (see [Creating Audit Scripts on page 202](#)).

Making a Database Call

This section describes how to configure a service task to establish a JDBC connection and run stored procedures on a DB2, Oracle, or SQL Server database.

To use a service task to run a stored procedure:

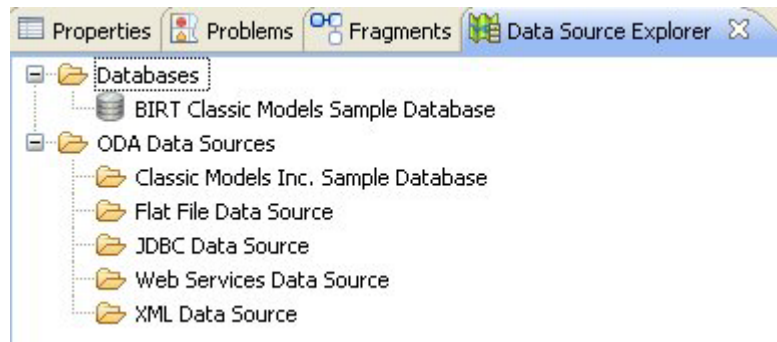
1. Define a database connection profile (if one does not already exist). A connection profile contains the connection property information needed to connect to the target database. See [Creating a Database Connection Profile on page 160](#).
2. Configure the service task to call the database and define the operation you wish to perform. See [Configuring the Service Task on page 164](#).
3. Map input and output parameters defined in the SQL query or stored procedure to corresponding data fields or parameters in the process.

Creating a Database Connection Profile


Create a database connection profile as follows:

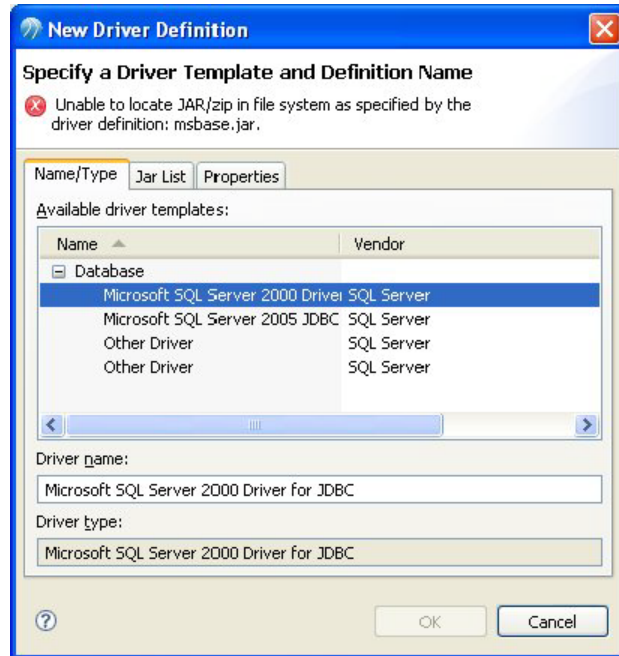
1. Select **Window > Show View > Other > Data Management > Data Source Explorer**.

The Data Source Explorer view is displayed:

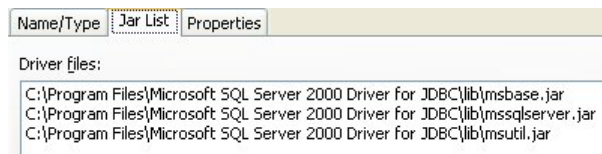


2. Right-click **Databases** and select **New**.
3. Select the type of database connection you want to create. Add a description (optional), and click **Next**.

4. In the resulting dialog, the **Drivers** drop-down list displays the JDBC drivers that have been defined for this database type. If a suitable driver has already been defined, select it; otherwise you must create one:
 - a. Download and install the JDBC drivers for the selected database connection.
 - b. Click the **New Driver Definition** button .
 - c. Select a driver definition template. This populates the **Driver name** and **Driver type** fields.



- d. Click the **Jar List** tab. Remove any default JAR files and click the **Add JAR/Zip** button to browse to the location of the JAR files that you installed as part of the JDBC driver installation, add them to the list of JAR files and click **OK**. For example:



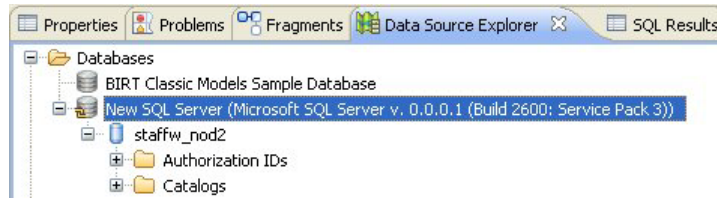
5. In the **New Connection Profile** dialog, enter the necessary database details:

Field	Description
Database	Depending on the platform, specify the database name (SQL Server and DB2) or the Oracle System ID (SID) for the database to which you want to connect.
Host	Specify the IP address or host name where the database is located.
Port Number	Specify the port used to access the database.
(DB2 only) Use client authentication	Allows you to use client-side authentication rather than database authentication.
User name	Specify the database username for the connection.
Password	Specify the database password for the username you specified.
Save password	Select this checkbox to avoid entering the password each time you connect.
Connection URL	Displays the connection URL that is built up automatically from the connection information entered in the dialog.
(Oracle only) Catalog	Select the appropriate Oracle catalog that you want to use (or accept the default, User).
Connect when the wizard completes	Select to automatically connect after creating the connection profile.
Connect every time the workbench is started	Select to automatically connect when starting TIBCO Business Studio with the current workspace.

6. Click the **Test Connection** button to verify the details you entered. Click **Finish**.

If the database is running and you selected the **Connect when the wizard completes** checkbox, you are connected and can view the newly created

connection in the Data Source Explorer view. Otherwise connect manually as described in [Connecting to the Database on page 163](#).



Connecting to the Database

If you do not connect automatically, connect as follows:

1. Select **Window > Show View > Other > Data Management > Data Source Explorer**.

The Data Source Explorer view is displayed.

2. Right-click the database to which you want to connect and select **Connect**.

Working Offline

The Data Source Explorer view supports an offline mode that allows you to work with the database schema without being connected to the database. To do this:

1. Create the database connection profile as described in [Creating a Database Connection Profile on page 160](#), and ensure that you are connected to the database.
2. Right-click the database connection in the Data Source Explorer view and select **Save Offline**. This creates a local copy of the database schema.
3. Disconnect from the database
4. Select **Work Offline**. This connects to the local copy of the database schema.

Configuring the Service Task

1. Select the Service Task, then on the **General** tab of the Properties view for the Service Task, select the **Database** option from the **Service Type** drop-down list:



Service Type: Database

Operation: ☐ Query ☒ Stored Procedure

Name:



Running a stored procedure is the only supported operation for the iProcess Engine; you cannot run a query using a database service task.

2. Select the **Stored Procedure** operation.
3. If you do not have a database connection follow this step; otherwise continue with [step 4](#).

Enter the name of the stored procedure in the following format:

[owner.]stored_procedure_name

The owner can be omitted for SQL server if the stored procedure is in the default iProcess database, and is owned by the default iProcess background user (swpro). The owner is required for all Oracle databases.





The following characters are invalid and must not be used in the name of the stored procedure:

' , '\t', '\n', '^', '?', '.', null (ascii code 0), ':', '@'

The full stop is permitted just once and is assumed to imply the separation of procedure owner from procedure name.


4. If you have a connection to the database, select the stored procedure as follows:

If a system participant already exists:

- a. Click  to the right of the **Participants** field. The **Select Participants** dialog is displayed.
- b. Select the appropriate system participant, then click **Add** to add them to the **Selection** list.
- c. Click **OK**. The selected participant is displayed in the **Participants** field.
- d. Click  in the right-hand pane.
- e. The **Pick Stored Procedure** dialog is displayed, listing the stored procedures that are available on this database. Select the stored procedure that you want to execute and click **OK**.
- f. The stored procedure (prefixed by the owner) is displayed in the **Name** field.

Configure the **Advanced** tab for the database connection as described in [Configure the Advanced Connection Properties on page 165](#).

If a system participant does not already exist:

- a. Click  in the right-hand pane. The **Create System participant** dialog is displayed.
- b. Select the database connection profile that you want to use, then click **OK**. A system participant is created (with the same name as the selected database connection profile) and assigned as the participant for the service task.
- c. The **Pick Stored Procedure** dialog is displayed, listing the stored procedures that are available on this database. Select the stored procedure that you want to execute and click **OK**.
- d. The stored procedure (prefixed by the owner) is displayed in the **Name** field.

Configure the **Advanced** tab for the database connection as described in [Configure the Advanced Connection Properties on page 165](#).

Configure the Advanced Connection Properties

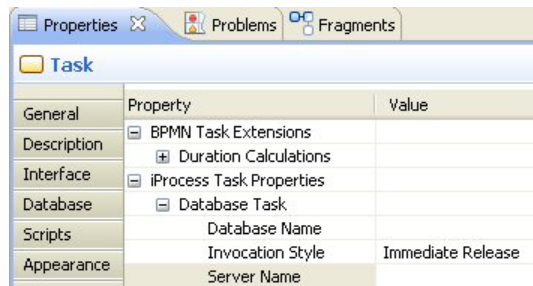
If the stored procedure is located in the default iProcess database, you do not need to configure the advanced properties of the database service task.

Configure the **Advanced** tab if either of the following is true:

- **SQL Server** The stored procedure is in the same SQL Server instance as the default iProcess database, but in a different database.

- **Oracle** The stored procedure is in a remote Oracle database different to the default iProcess database (connected using db links).

The database section of the **Advanced** tab has the following advanced properties for iProcess destinations:



Depending on the database you are connecting to, enter the following:


- **SQL Server** For **Database Name**, specify the database that contains the stored procedure that you want to run. The database must be on the same machine and in the same SQL Server instance as the default iProcess database. Select the desired invocation style (see [Invocation Styles on page 76](#)). Do not enter a value for **Server Name**.
- **Oracle** For **Database Name**, specify the db link name for the database that contains the stored procedure that you want to run. Select the desired invocation style (see [Invocation Styles on page 76](#)). Do not enter a value for **Server Name**.
- **DB2** Executing stored procedures on a remote database is not supported. Information entered here will be preserved in the procedure definition but will be ignored by the runtime plug-in, which will assume that the stored procedure is located in the local database instance.

After configuring the **Advanced** tab, define the input and output parameters required for the stored procedure.

Mapping Existing Process Data


This section describes how to map the stored procedure parameters to existing process data. If you can connect to the database, TIBCO Business Studio can automatically create a business object model based on the stored procedure return parameters as described in [Creating a Business Object Model From Returned Data on page 168](#).

1. Click **Define procedure parameters...** on the **General** tab, or click the **Database** tab to define the parameter mappings between parameters in the stored procedure and TIBCO Business Studio data fields or parameters.

2. If you are connected to the database, the **Parameter** and **Type** fields are automatically populated from the stored procedure. Add parameters as follows:
 - a. Click the  button to add a parameter.
 - b. Select from the drop-down list whether the parameter is input to the stored procedure (**IN**), output to the iProcess Engine (**OUT**) or both (**INOUT**).



You can remove or reposition the parameters using the **Move Up**, **Move Down**, and **Remove** buttons.

- Click  to display the **Select Data Field or Formal Parameter** dialog. Select the data field or formal parameter you want to map to the stored procedure parameter. In addition to basic types, you can also select a data field or parameter that is an external reference to a business object model.



There is no validation in TIBCO Business Studio to ensure that the data types of data fields you select match the data types of parameters returned by the stored procedure. You must ensure that the data types of the data fields and the stored procedure return parameters match.

For example:

Parameter	Type	Data Field
* Acct1	IN	DEBITACCOUNT
* Amount	IN	AMOUNT
* Bal	OUT	BALANCE
* Rcode	OUT	[Return Code]
* DRID	OUT	[Delayed Release Id]

In this example, the stored procedure takes the input parameters **Acct1** and **Amount** and maps them to the Data Fields **DEBITACCOUNT** and

AMOUNT. Similarly, the stored procedure outputs the parameters **Bal**, **Rcode**, and **DRID** to **BALANCE**, **Return Code**, and **Delayed Release Id**.




There are two special fields that are used by the iProcess Engine when the process is deployed and executed - **Delayed Release ID** and **Return Code**. These are not data fields and can only be defined once in a service task.

- If this task is set up for delayed release, you need to use the **Delayed Release ID** field so that the ID can be passed to the stored procedure for later use when it needs to execute the task.
 - The **Return Code** field enables you to check if the stored procedure has been called successfully. See [Return Code](#) below for more information about this field.
4. Click the **Advanced** tab to optionally add delayed release information and a corresponding condition (see [Reference on page 253](#)).
 5. Click the **Scripts** tab and add any audit scripts you require (see [Creating Audit Scripts on page 202](#)).

Creating a Business Object Model From Returned Data

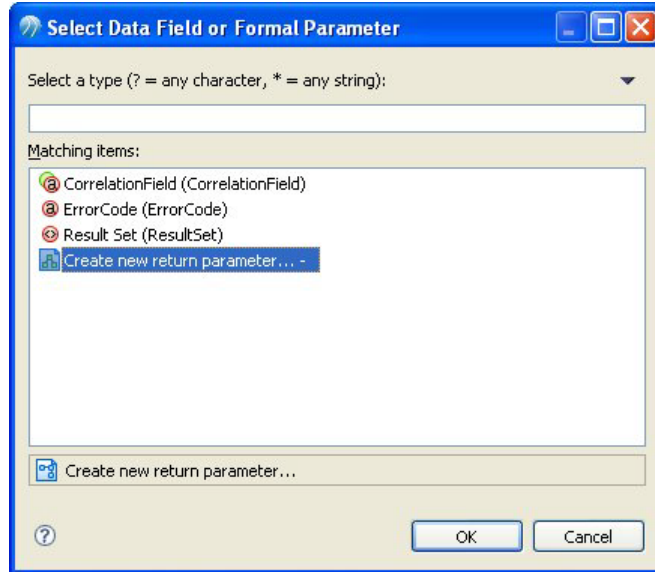
This section describes how to automatically create a business object model that is based on the stored procedure return parameters. You must have access to the database to do this.

1. Click **Define procedure parameters...** on the **General** tab, or click the **Database** tab to define the parameter mappings between parameters in the stored procedure and TIBCO Business Studio data fields or parameters.
2. If you are connected to the database, the **Parameter** and **Type** fields are automatically populated from the stored procedure. If you need to add more parameters, do so as follows:
 - a. Click the  button to add a parameter.
 - b. Select from the drop-down list whether the parameter is input to the stored procedure (**IN**), output to the iProcess Engine (**OUT**) or both (**INOUT**).




You can remove or reposition the parameters using the **Move Up**, **Move Down**, and **Remove** buttons.

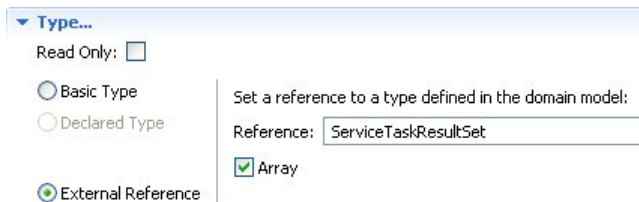
3. Click  to display the **Select Data Field or Formal Parameter** dialog. Select **Create new return parameter** and click OK.



This creates a data field called **ResultSet**:

Parameters:	Parameter	Type	Data Field
	✖ Parameter 1	IN	ResultSet 

The data field **ResultSet** is defined as an external reference that points to a business object model:



If a connection exists, the business object model is populated by introspection of the database return parameters. If no connection exists, you must to edit the business object model to match return parameters from the stored procedure.



There are two special fields that are used by the iProcess Engine when the process is deployed and executed - **Delayed Release ID** and **Return Code**. These are not data fields and can only be defined once in a service task.

- If this task is set up for delayed release, you need to use the **Delayed Release ID** field so that the ID can be passed to the stored procedure for later use when it needs to execute the task.
- The **Return Code** field enables you to check if the stored procedure has been called successfully. See [Return Code](#) below for more information about this field.

- 4. Click the **Advanced** tab to optionally add delayed release information and a corresponding condition (see [Reference on page 253](#)).
- 5. Click the **Scripts** tab and add any audit scripts you require (see [Creating Audit Scripts on page 202](#)).

Return Code

This field controls the runtime behavior of the database service task when the stored procedure it calls returns. The writer of the stored procedure uses the return code to specify the desired iProcess background behavior.

The Return Code field is a special field that, by default, is not visible to any other Task within the process. You can however map the Return Code field to a data field in TIBCO Business Studio, thus making it available to other Tasks in the process. The possible return conditions are as follows:

Return Code	Description
1	Indicates successful completion of a stored procedure if the calling database service task is part of a withdraw action. If 1 is returned during a normal EAI step call (not as part of a withdraw action), it is treated as an error.
2	Indicates successful completion of the stored procedure and that its part of the workflow transaction can be committed (if the other tasks in the transaction are successful). Any delayed release settings defined for the task are taken into account (in contrast to Return Code 3).

Return Code	Description
3	Indicates successful completion of the stored procedure, but treats the task as delayed release. Instead of using the delayed release settings defined for the task, the iProcess background waits for an APPRELEASE command to be issued before the case continues. All return parameters are valid and can be used immediately in other Tasks.
-1	The stored procedure detected an error. This causes the iProcess background to abort the workflow transaction. The stored procedure executes, but its effects are rolled back.
Other values	Error (equivalent to -1).

Calling Java Code

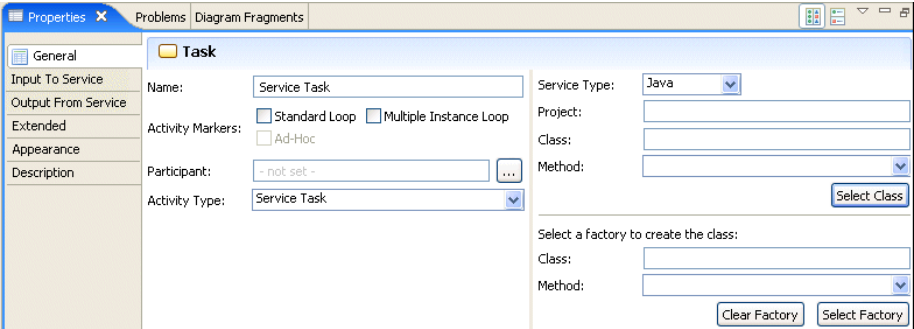
You can call Java code using a Service Task as follows:

1. Create a Java project in your workspace that contains code that you want to call as well as any dependencies.

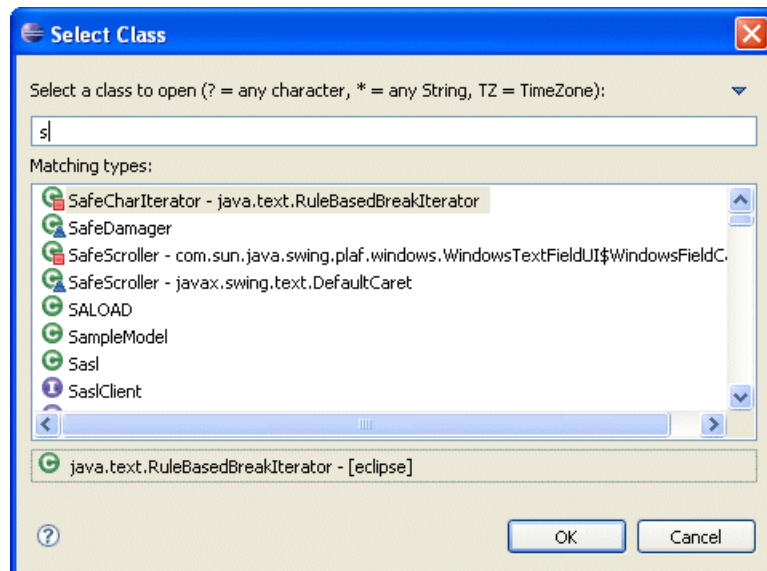


For more information about creating or importing Java projects, see the *Java Development User Guide* in the Eclipse documentation.

2. Select the Service Task, then on the **General** tab of the Properties view for the Service Task, select the **Java** option from the **Service Type** drop-down list:



3. Click **Select Class**. The **Select Class** dialog is displayed:



An alternative to selecting the Class is to select a factory to create the Class. To do this, click **Select Factory**. If a factory is available, selecting it populates the **Class** and **Method** fields.

Select the appropriate class that you want to use. If you begin typing, the matching classes are displayed. In the previous example, the character **s** was entered, and all classes starting with **s** are displayed.

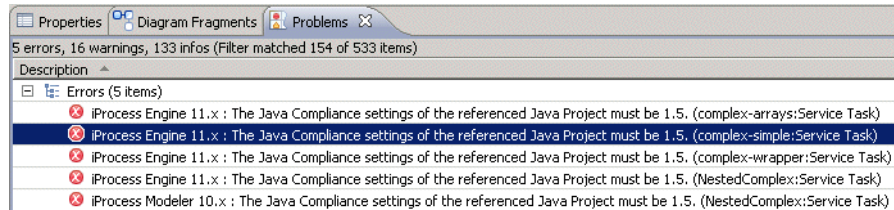
4. Select from the drop-down list the **Method** that you want to use. For example:

Service Type:	Java
Project:	EAIPOJO
Class:	com.tibco.mortgage.MortgageService
Method:	<div> <div>getName () - String</div> <div>getName () - String</div> <div>makePayment (AccountHolder, AccountHolder, CurrencyAmount) - PaymentRef</div> <div>makePayment (String, String, String, String, BigDecimal) - String</div> <div>setName (String) - void</div> </div>
Select a factory to	
Class:	
Method:	
<div>Clear Factory</div> <div>Select Factory</div>	

5. Click the **Scripts** tab and add any audit scripts you require (see [Creating Audit Scripts on page 202](#)).

Notes

- The Java compiler compliance level must be set to version 1.5 for iProcess destinations, otherwise problems such as the following are displayed:



You can right-click these problems and select **Quick Fix**, or manually change the Java compiler compliance level. Note, however, that even after the problem is fixed, the problems continue to be displayed in the Problems view until you clean the workspace by select **Project > Clean**.

- You must use a factory static method to instantiate the Plain Old Java Objects (POJO).
- You must define a constructor if you have input data. The input data that you call and the return types are constrained to the following (and arrays of the following):
 - primitives
 - wrapper classes for primitives
 - `Java.util.Date`
 - `Java.math.BigDecimal` or `Java.math.BigInteger`
 - `Java.lang.String`



JavaBeans and arrays are treated as pass by reference. They must also be "true" JavaBeans in terms of public getters and setters.

Complete the Parameter Mapping

Click the **Input To Service** and **Output From Service** tabs to complete the input and output mapping between any parameters or data fields in your process and the Java code.

Mapper Restrictions and Notes

- You cannot map data fields or parameters with spaces in their names.

- If you attempt to map data fields or parameters to parameters of a different type (for example, mapping a String to an Integer), an error is generated in the Problems view.
- If numbers are being mapped, you must perform a one to one mapping.
- Date Time data fields are deployed/exported as two separate date and time fields in the iProcess Engine (suffixed "_D" and "_T"). They are then recombined for use in Java.
- Floats and doubles are limited in precision and TIBCO Business Studio validates the mapping of these data types as follows:

Data Type	Process to Java	Java to Process
Float	6 maximum	6 minimum
Double	15 maximum	15 minimum

From process to Java, error are generated if these limits are exceeded. From Java to process, warnings are generated if these minimum values are not adhered to.

- Class loaders and threads have an adverse effect on EAI Java steps and therefore should be avoided.
- Use of system exit is prohibited.
- POJO steps in the iProcess Engine are immediate release, therefore withdraw is not supported.
- Mapping of process Decimal numbers to Java Integer numbers can lead to a loss of precision (there is a validation rule for this).
- Mapping of Java Decimal numbers to process Integer numbers can lead to a loss of precision (there is a validation rule for this).
- There is a validation rule that checks that all the input parameters for the POJO service have been mapped.

Java Deployment

Deployment is described in [Deploying to the iProcess Engine on page 219](#). However, note the following:

- The iProcess Engine must be running, and any JARs and dependent libraries must be imported into *SWDIR/eaijava/libs/repository/user*.
- After deployment, you must stop and start the iProcess Engine if any JARs have changed (or if you are deploying for the first time).



There is no facility in the iProcess Modeler to edit an EAI POJO step.

Configuring POJO in TIBCO Business Studio

The `SWDIR\ei\java\pojo.properties` configuration file allows you to change the run-time behavior of POJO in the iProcess Engine.



- You should only modify this file if you have specific requirements that require changes to the default configuration. If you do modify this file, make a copy as a safeguard.
- Ensure that you are aware of the limits of the data types you are using. For example, if a value greater than 32767 is set for the input BigInteger and the field is mapped to a short, the case can fail if it tries to use the default value.
- Be aware that a default value may cause a user to believe that they have set a particular value when it is actually the default setting.

Changing the Exception Mode

This setting determines how strict the conversion between iProcess Engine fields and Java will be. The following line allows you to change the default exception mode (0):

```
ConversionExceptionMode = 0
```

The following table lists allowable values and their effect.

Value	Effect
0	Conversion anomalies will throw ConversionException causing the EAI_POJO Step to fail.
1	Same effect as 0, except truncation and field length errors will not throw an exception.
2	Same effect as 1, but additionally floating point rounding and loss of precision will not throw exceptions.

Changing Conversion Defaults Between iProcess and Java

These settings allow you to control how conversion to and from iProcess Engine fields are handled when the values are null or empty (depending on the data type).

Input

The default setting converts iProcess Engine fields according to type:

```
ApplyDefaultsForEmptyInput = true
```

```
INPUT.TEXT.java.lang.String      =
INPUT.NUMERIC.java.math.BigInteger = 0
INPUT.NUMERIC.java.math.BigDecimal = 0
INPUT.DATE.java.util.Date         = 01/01/1970
INPUT.TIME.java.util.Date          = 00:00
```

If you set **ApplyDefaultsForEmptyInput** to **false**, iProcess Engine fields are passed unchanged. See [Usage Notes](#) for more information.

Output

The default setting converts to iProcess Engine fields according to type:

```
ApplyDefaultsForEmptyOutput = true
```

```
OUTPUT.TEXT      =
OUTPUT.INTEGER   = 0
OUTPUT.DECIMAL   = 0
OUTPUT.DATE      = 01/01/1970
OUTPUT.TIME      = 00:00
```

If you set **ApplyDefaultsForEmptyOutput** to **false**, the mapped iProcess Engine fields are not updated, leaving them unchanged.

Usage Notes

- Quotes should not be used when defining values associated with INPUT.TEXT.java.lang.String or OUTPUT.TEXT.
- An example of a valid empty string is:

```
INPUT.TEXT.java.lang.String =
```

The following is invalid:

```
INPUT.TEXT.java.lang.String = ""
```
- Any string value can be used as a default including embedded spaces.
- Values associated with the following fields all have numeric values (including the date/time related properties):

```
INPUT.NUMERIC.java.math.BigInteger
INPUT.NUMERIC.java.math.BigDecimal
OUTPUT.INTEGER
OUTPUT.DECIMAL
```

- Date/time related properties can be set to the current date/time by specifying -1 as a value.

- INPUT.TEXT.java.lang.String default value will NOT be used if the iProcess Engine field that it came from was an "empty" text field. This is because empty strings are valid. The default value will however be used if the iProcess Engine field contains null.
- In contrast, if a numeric iProcess Engine field contains null or is empty, the default value will be used depending on the type (BigInteger or BigDecimal). Similarly, a null or empty iProcess Engine Date or Time field will use the default INPUT.DATE.java.util.Date or INPUT.TIME.java.util.Date value.
- Default INPUT substitution will not occur unless:
`ApplyDefaultsForEmptyInput = true`
- Default OUTPUT substitution will not occur unless:
`ApplyDefaultsForEmptyOutput = true`
- In contrast to inputs, if an output has a null value and `ApplyDefaultsForEmptyOutput = false`, the iProcess Engine fields are not updated.
- As with INPUT.TEXT.java.lang.String, the value associated with OUTPUT.TEXT applies only to null fields and not empty fields because empty strings are valid.

Configuring TIBCO iProcess Conductor Service Tasks

From within TIBCO Business Studio you can configure the following types of service tasks for use with TIBCO iProcess Conductor:



For more information about these types of EAI steps, see *TIBCO iProcess Conductor Concepts Guide* and *TIBCO iProcess Conductor Implementation*.

- **Orchestrator steps** Orchestrator steps are used to sequence the process components in an execution plan. They also enable you to send iProcess data to the TIBCO iProcess Conductor and for the TIBCO iProcess Conductor to process the data before sending the resulting data back to the TIBCO iProcess Engine.

An Orchestrator step sends a message to the iProcess Conductor indicating that it has been reached. Depending on the message type, it can then wait for a message from the iProcess Conductor before it releases. This enables the iProcess Conductor to make sure that any tasks that have defined dependencies must have those dependencies satisfied (including the transfer of input and output data) before being allowed to proceed.

- **Order steps** These steps can be used to do the following:
 - Provide an interface that enables data from a specific order request or order amendment to be used in iProcess fields and vice versa.
 - Progress the status of an order within the TIBCO iProcess Conductor based on the current stage in the fulfillment process.
 - Pass the name of the event step to the TIBCO iProcess Conductor so that it knows which step in the process to notify when an order amendment is received.



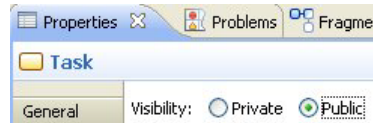
Use a single Order step to change the status of an order *or* update order parameters. Do not use a single Order step to do both a status change and update. This is because the status change will fail if the update fails and this may not be the intended result. To perform a status change and an update, use two consecutive order steps; one that performs a status change and one that performs an update.

The Transformation step can be implemented in TIBCO Business Studio using a script task of the type **Transform** (see [Creating a Script Task, page 186](#)).

Orchestrator Service Tasks

Configure an orchestrator service task as follows:

1. Select the service task, then on the **General** tab of the Properties view for the service task, select the **IPC Orchestrator** option from the **Service Type** drop-down list.
2. On the **General** tab, set the visibility of the task to **Public**.



3. From the Message Type list, select the type of message you want to send ([Activate Sink](#), [Start Plan](#), or [Task Complete](#)).

Activate Sink

If you select the **Activate Sink** message type, use the **Interface** tab to specify the process data that is passed between the TIBCO iProcess Conductor and the TIBCO iProcess Engine:


- Parameters with a Mode of **In** appear on the **Message Payload Fields** tab when deployed or exported to iProcess. These represent data that is passed from TIBCO iProcess Engine to TIBCO iProcess Conductor.
- (Optional) Parameters with a Mode of **Out** appear on the **Orchestrator Response Fields** tab. These represent data that is passed from TIBCO iProcess Conductor to TIBCO iProcess Engine.





The Plan Id Field and Task Id are created upon deployment or export to the TIBCO iProcess Engine.

Start Plan

Configure a **Start Plan** message as follows:

1. If you select the **Start Plan** message type, specify the following:
 - **Graft Step Name** Press Ctrl+Space to select from a list of reusable sub-processes. The reusable sub-process that you select should be one that is set up to be a graft step (see [Graft Steps on page 106](#)). This is used to orchestrate the required execution plan.
 - **Execution Plan Override Field** Click  to select a data field or parameter that contains the unique identifier for the execution plan or execution plan template.
 - **Callback Task** Press Ctrl+Space to select from a list of receive tasks. The receive task is used to pass the execution plan ID back to the fulfillment process (this was achieved in iProcess Modeler using an ad-hoc orchestration event step).

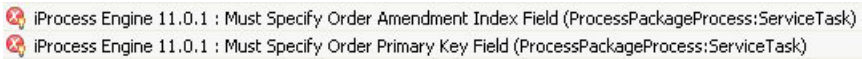
- 2. Optionally specify classification values for the execution plan:
 - To create groups and values in TIBCO Business Studio, select the **Designtime** option. Click  to add classification values, and click into the table cells to change the **Group Name** and **Value Code**.
 - To specify a data field or parameter to represent the classification value in the runtime environment, select the **Runtime** option and click  to select a data field or parameter.

Task Complete If you select the Task Complete message type, use the **Interface** tab to specify the process data that is passed from the TIBCO iProcess Engine to the TIBCO iProcess Conductor. You can only specify parameters with a Mode of Out (these appear on the **Orchestrator Response Fields** tab, and represent data that is passed from TIBCO iProcess Conductor to TIBCO iProcess Engine).

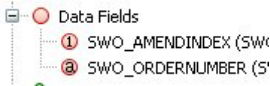
Order Service Tasks

Configure an order service task as follows:

- 1. Select the Service Task, then on the **General** tab of the Properties view for the Service Task, select the **iPC Order** option from the **Service Type** drop-down list.
- 2. Provide data fields or parameters for the **Order Field Primary Key Field** and **Order Amendment Index Field**. The easiest way to do this is from the Problems view:
 - a. Locate the following problems in the Problems view:

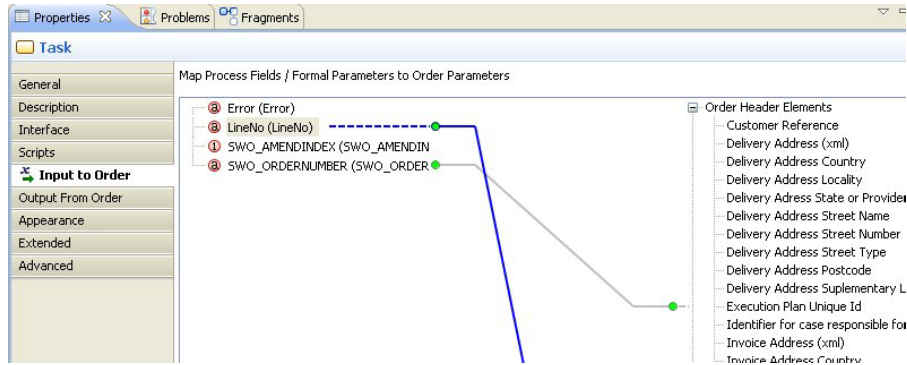

 - b. Right-click one of the problems in the Problems view and select Quick Fix.
 - c. Repeat these steps for the **Order Primary Key Field** and the **Order Amendment Index Field**.

By performing the quick fix, the required data fields are created:



3. Select the desired transition for the task:
 - **Execution** Sets the status of the order from **Feasibility** to **Execution**.
 - **Feasibility** Sets the status of the order from **Draft** to **Feasibility**.
 - **Feasibility Failed** Sets the status of the order from **Feasibility** to **Feasibility Failed**.
 - **Plan Development** Sets the status of the order to **Plan Development**, which means that the fulfillment process has notified iProcess Conductor Order Management that it is at the stage where it is waiting for the TIBCO iProcess Decisions Plug-in to select an execution plan template on which to base an execution plan.
4. In the **Ad-Hoc Event Step** field, press Ctrl+Space to select from a list of receive tasks. The receive task is used for iProcess Conductor Order Management to notify of order amendments.
5. In the **Event Case** field, specify the data fields from which the order step should get the process name and case number.
 - Select **This Procedure** to use the default iProcess fields **SW_CASENUM** and **SW_PRONAME**.
 - Select **Specify Fields** to specify different data fields for the case number and process name. Then, click to select the data field you have defined for the case number and process name, respectively.
6. In the **Order Line State Change** field, you must specify:
 - A data field for **Line Number Selection** that specifies the location in the XML code where you want the status to change. The value that you specify is then inserted into an XPath expression to locate the value in the XML that you want to change.
 - The **Order Line Status** that you want the order step to set. You can select either **Cancelled** to indicate that the cancel operation performed on the order line is complete, or **Complete** to indicate that the order line has orchestrated and has completed.
7. In the Exception Handling section, you can either:
 - Select the **Fail Step on XML Exception** checkbox. This box is selected by default. Leaving this selected means that the order service task causes the process to stop and logs the messages to the **eaijava** log files in the **\SWDIR\logs** directory, or
 - De-select the **Fail Step on XML Exception** checkbox. In the **Exception Field Name** field, specify the data field that you want to populate with the message from the non-fatal exception.

8. In the **Input to Order** and **Output to Order** tabs, map the process data and order parameters. For example:



Chapter 6

Working with Scripts

This chapter describes working with scripts.



The JavaScript script grammar is only available with iProcess destination environments and when the Solution Design capability is selected:



✓ Business Analysis

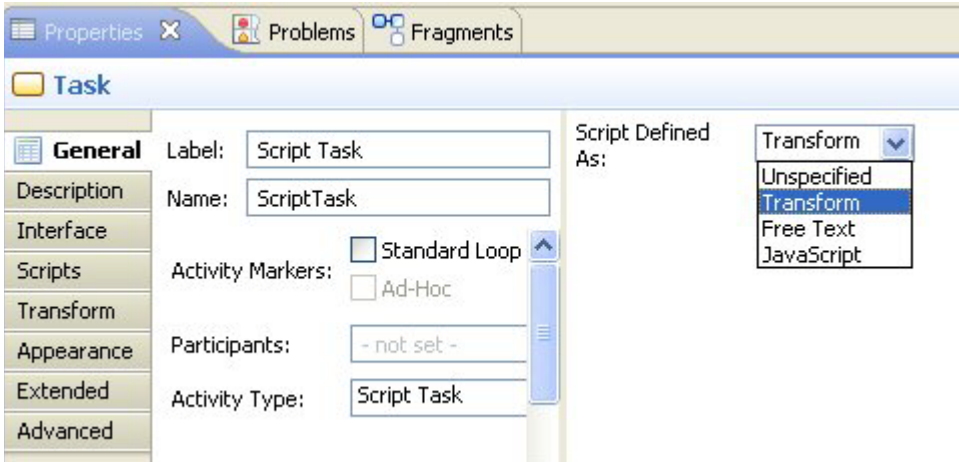
✓ Solution Design

Topics

- [Creating a Script Task, page 186](#)
- [Using iProcess Expressions and Functions, page 194](#)
- [Creating a Script for a User Task, page 199](#)
- [Creating Audit Scripts, page 202](#)
- [Associating a Script with a Conditional Flow, page 204](#)
- [Timer Event Scripts, page 205](#)
- [Customizing JavaScript Presentation Preferences, page 207](#)
- [Customizing XPath Presentation Preferences, page 209](#)

Creating a Script Task

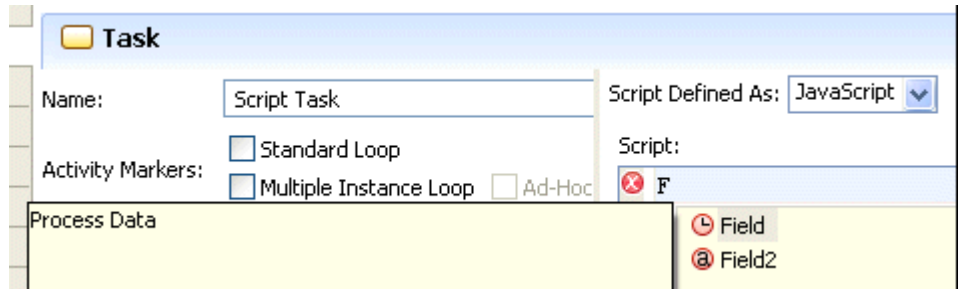
In the Properties view for a script task, you can select from the available script grammars, and enter a script that will be executed at runtime. The script grammars displayed depends on the destination environment and the Eclipse capability you have selected. For example, for the iProcess Engine destination environment with the Solution Design capability selected, you can enter JavaScript or a transform (either XPath or XSLT):



On the **General** tab, there may a text description of the required script. This description will be converted to comments if you select **JavaScript** from the **Script Defined As** list. If you select **Unspecified** from the **Script Defined As** list, the description is lost. You can however recover the description by pressing **Ctrl + Z**.

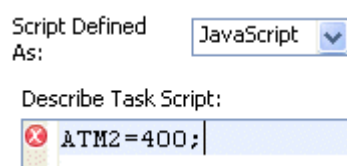
Entering JavaScript

The **Script** area supports the usual text editing assistance such as color syntax highlighting, content assist and error markers. For example, if you want to specify a data field called **Field**, enter the character "F", then press Ctrl+Space. All matching data fields are displayed:

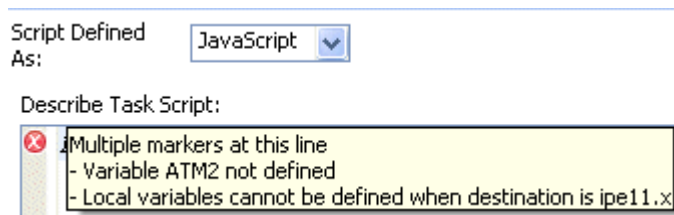


Although TIBCO Business Studio allows you to create process data (for example, a data field) that has spaces in its name, such names are not displayed in content assist.

You can then select the desired data field from the list and continue entering JavaScript:

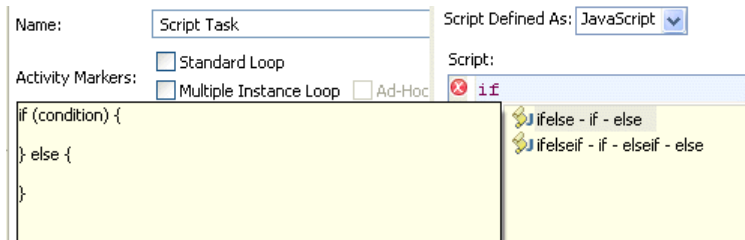


Note that in this case there is an error marker next to the line. This is because validation has reported an error in the Problems view. You can position the cursor over the error marker to display the reason for the error:



When these errors are corrected, the error marker and entries in the Problems view are removed.

Content assist also provides templates for common JavaScript constructs. For example, if you enter `if`, then press `Ctrl+Space`, you can use the following template to construct an if else:



When using the colon character in a JavaScript conditional expression, ensure that you insert spaces before and after the colon. For example, the expression `FIELD2==4?PARAM1:PARAM2;` is not valid. The corrected expression is `FIELD2 == 4 ? PARAM1 : PARAM2;`

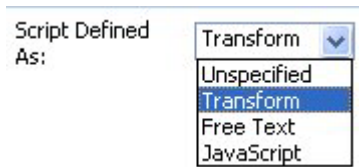
iProcess Script is supported for backward compatibility via the TIBCO iProcess JavaScript class library. Within the packaging/export phase this JavaScript is converted to iProcess Script for execution by the iProcess Engine. If you have selected the **iProcess Engine** or **iProcess Modeler** destination environment, you can view the available list of iProcess Script functions by typing **IPE** and pressing `Ctrl+Space`.



Some iProcess functions expect parameters of the data type VarType. TIBCO Business Studio Text, Decimal Numbers, Integer Numbers, and the Date or Time portion of a Date Time can all be passed to a VarType parameter.

Transforming Data (XPath and XSLT)

To apply an XPath expression or XSLT to process data, select the **Transform** option on the **General** tab for the script task:



Using XPath or XSLT, you can perform the following types of mappings:

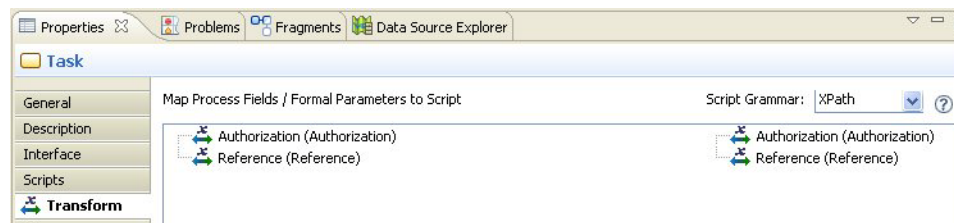
- **Simple type to simple type** A value from one parameter is mapped to another. At runtime, the value of the source field is inserted into the target field.

- **Simple type to complex type** A simple type is mapped to a data field that is an external reference to the business object model. At runtime, an output XML is created and inserted into a memo field in iProcess.
- **Complex type to simple type** A data field that is defined as an external reference to the business object model is mapped to a simple type. At runtime, an output XML is created and inserted into a memo field in iProcess.



For iProcess destinations, a script task can have multiple simple input mappings, or one complex mapping as input.

After changing the **Script Defined As:** field to **Transform**, the **Transform** tab is displayed:



This tab allows you to transform data either by using XPath expressions or by using XSLT by selecting the appropriate option from the **Script Grammar** drop-down list.

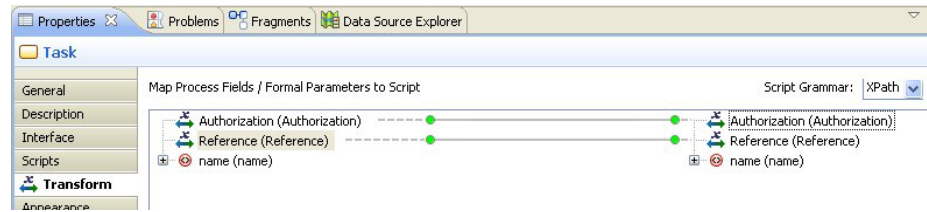
XPath Transformation

When you select **XPath** as the script grammar, you can enter XPath expressions as described in this section.

Simple Type to Simple Type

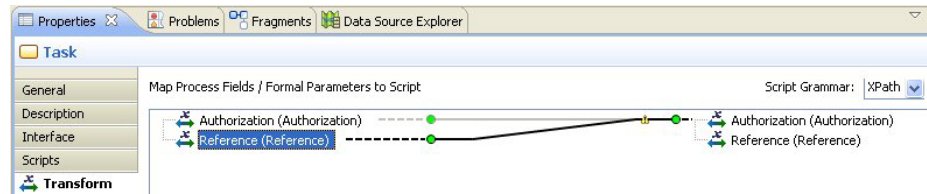
To transform simple types with an XPath expression, do the following:

In the **Transform** tab, drag the pointer from a data field or parameter on the left side to one on the right to create a mapping:



If you change the script grammar from **XPath** to **XSLT**, the script is converted to XSLT. However, this is a one-way operation; if you change to XSLT, then back to XPath, the original mappings and scripts are lost.

You can concatenate parameters as follows:

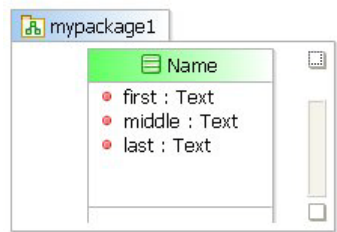


Simple Type to Complex Type

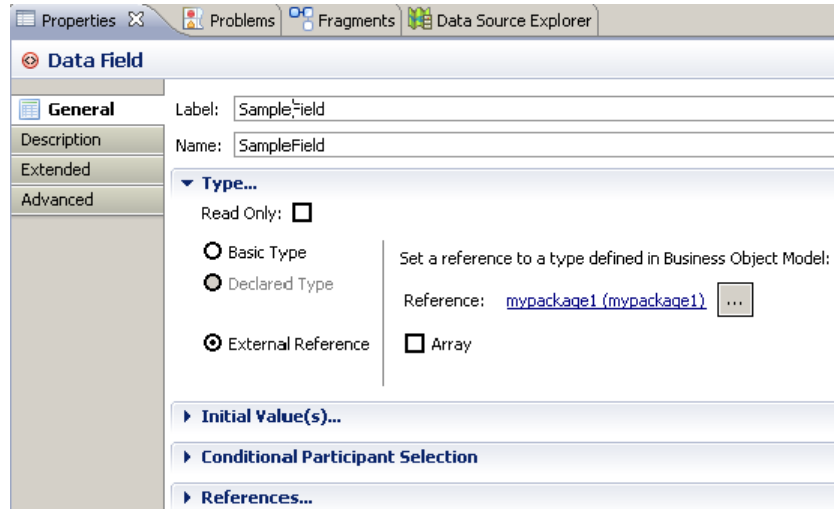
Complex data types are represented in TIBCO Business Studio as data fields or parameters that are external references to business objects.

For example:

1. Create the following class in a business object model:



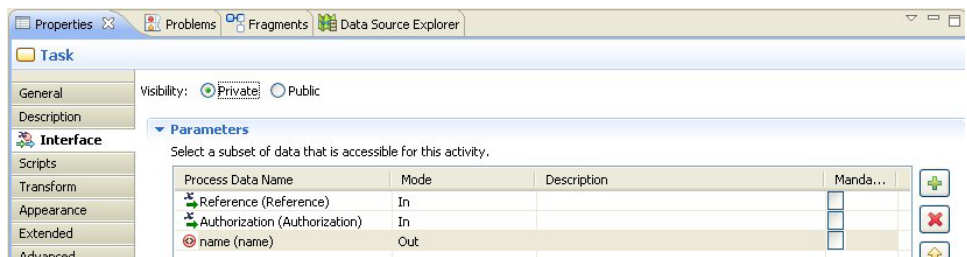
2. Create a Data Field in the process that has an external reference to a class in the Business Object Model.



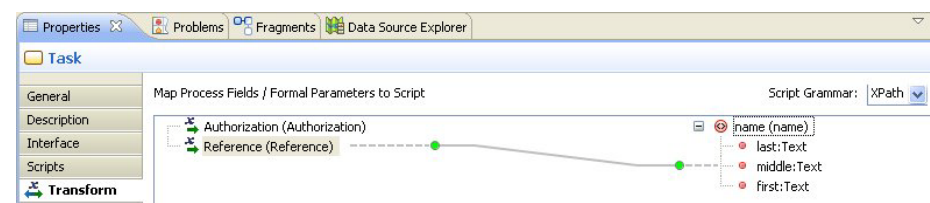
3. Map in the script Transform task using the visual mapper to XSLT.
4. Make sure that the special fields (data fields that have an external reference to a class in the Business Object Model) are only used in the script task. The way to do it is to go through all the other tasks in the process and instead of the default Interface generated (all fields) select the used ones and assure no Business Object Model-related fields are there. You have to do this also to the END task.

After this is done, then the error disappears and the process can be deployed.

On the **Interface** tab, the following parameters and data fields are selected for the script task:

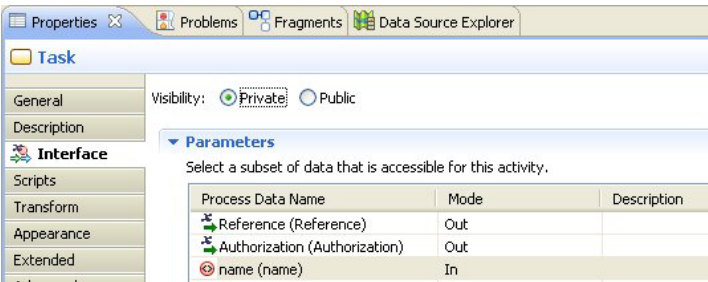


On the **Transform** tab, a simple to complex mapping can be created as follows:

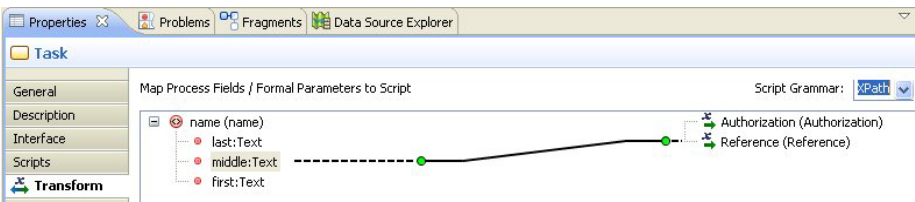


Complex Type to Simple Type

Using the same business object model class as the previous example, but with the **Interface** tab configured as follows, you can create a complex to simple type mapping:



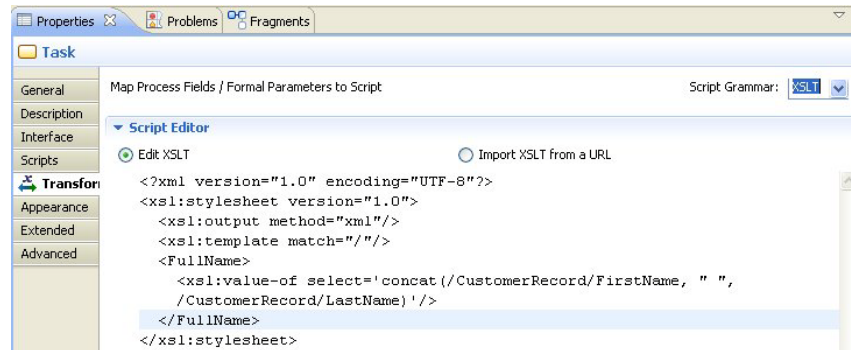
On the **Transform** tab, the mapping is as follows:



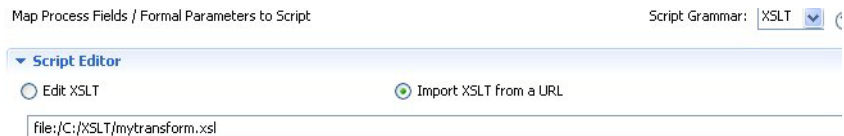
XSLT Transformations

When you select **XSLT** as the script grammar, you can use XSLT as follows:

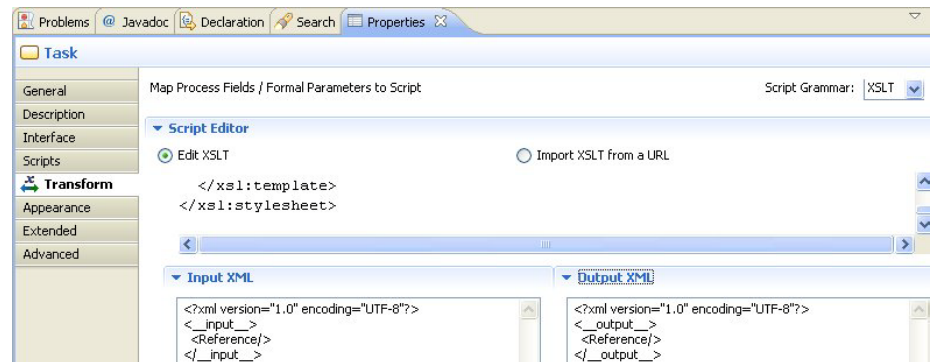
1. Select one of the following options for specifying XSLT:
 - To enter the XSLT directly, select the **Edit XSLT** option and enter the transformation in the area provided. For example:



- To specify a remote XSLT, select the **Import XSLT from a URL** option and specify the URL to the XSLT.



2. You can view the input and output XML by expanding the **Input XML** and **Output XML** sections:



Using iProcess Expressions and Functions

This section shows how to use iProcess expressions and functions within TIBCO Business Studio.

iProcess Script File Functions

When using iProcess Script file functions (for example, `IPEFileUtil.FILEDELETE`), you should ensure that you correctly specify any directory paths. In a path, any single backspace character (`\`) is interpreted by the parser as an invalid string character. For this reason, specify two backslash characters (`\\`) rather than one when specifying paths. For example:

```
IPEFileUtil.FILEDELETE("C:\\Temp\\myfile.txt")
```

Obtaining iProcess User Attributes

To obtain attributes of users like Starter, Group, and User, you can use the following Util classes.

- Starter --> `iPEStarterUtil` (for Script Tasks)
- User --> `iPEUserUtil` (for user task scripts)
- Group --> `iPEGroupUtil` (for user task scripts)

These Util classes have a **`getAttribute(String attributeName)`** function which allows you to get the value of the passed attribute which has been defined by the System Administrator. For example:

```
IPEStarterUtil.GETATTRIBUTE("Name");
```

This would return **`SW_STARTER:NAME`**.

iProcess Functions With Optional Parameters

There are several iProcess functions with optional repeating parameters. For example, **`IPEProcessUtil.CASESTART0`** has four required parameters, after which you can specify the following pair of parameters:

- *fieldname (text)* is used to start the case with data. The argument consists of sets of pairs of arguments. Fieldname is the first argument in the pair.
- *fieldvalue (anytype)* is used to start the case with data. The argument consists of sets of pairs of arguments. Fieldvalue is the second argument in the pair.

Although the iProcess Engine expects these optional parameters as pairs (*fieldname1* and *fieldvalue1*, *fieldname2* and *fieldvalue2*, and so on), there is no validation in TIBCO Business Studio to ensure that they are passed as pairs. Therefore a set of three optional parameters (*fieldname1* and *fieldvalue1*, *fieldname2*) would appear valid in TIBCO Business Studio, but would not be valid in the iProcess Engine.

The functions **IPEProcessUtil.TRIGGEREVENT()** and **IPEGENERALUTIL.SWITCHVAL()** also have optional repeating parameters that must be specified in pairs.

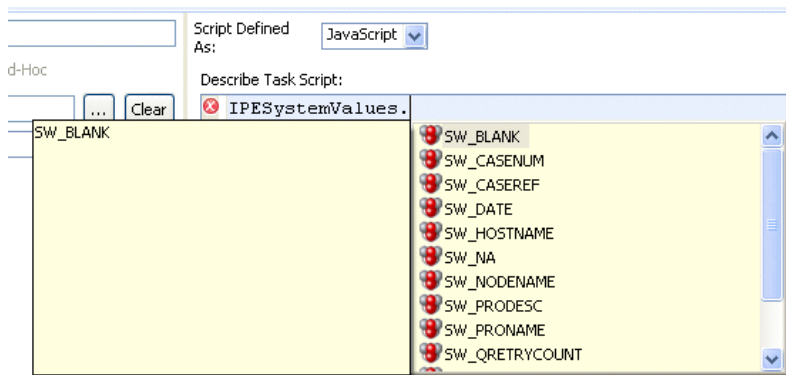
For more information, see the *TIBCO iProcess Expressions and Functions Reference Guide*.

Using iProcess Array Functions

For iProcess Array Functions (FINDARRELEMENT, NEXTARRELEMENT), the array field name must be in quotation marks and must specify a data field that is an Array. For example, if the array field name is CUSTOMER, then it must be entered in the expression as "CUSTOMER" and CUSTOMER must be a data field of the type Array.

Using iProcess System Fields

Key words, or system values, are special iProcess words that have particular values and meanings. In TIBCO Business Studio, the read-only iProcess system fields are displayed in content assist. For example:



The following table details each of the system values listed in content assist.



For more information about the iProcess system fields, see [TIBCO iProcess System Fields](#) on page 283.

Key Word	Meaning
SW_CASEDESC	The case description of the current case as entered by the user starting the procedure or can be calculated.
SW_CASENUM	The case number of the current case, allocated sequentially by iProcess.
SW_CASEREF	The case reference number of the current case in the format x-yy, where x is the number of the procedure and yy is the number of the case.
SW_DATE	The system date.
SW_HOSTNAME	The name of the host node for the procedure.
SW_NA	Not Assigned - a field has no value.
SW_NODENAME	The name of the iProcess Engine.
SW_PRODESC	The description of the procedure (up to 24 characters).
SW_PRONAME	The name of the procedure (up to 8 characters).
SW_QRETRYCOUNT	The number of times that a message in a message queue has failed. The field's value is 0 the first time a message is processed, and is incremented each time the message fails. For example, if a BG process is processing a message and SW_QRETRYCOUNT = 2, this means that the BG is attempting to process the message for the third time.
SW_STEPDESC	The description of the step (up to 24 characters).
SW_STEPNAME	The name of the step (up to 8 characters).
SW_TIME	The system time on the iProcess Engine.

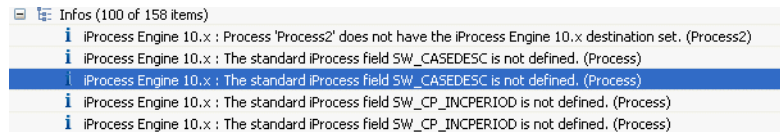
There are also certain key words that apply just to sub-procedures. These key words are not displayed in content assist, but you can create data fields for them:

Key Word	Meaning
SW_MAINCASE	The top level procedure's case number.
SW_MAINPROC	The top level procedure's name.

Key Word	Meaning
SW_MAINHOST	The host where the top level procedure resides.
SW_PARENTCASE	The parent procedure's case number.
SW_PARENTPROC	The parent procedure's name.
SW_PARENTHOST	The host where the parent procedure resides.
SW_PARENTREF	Internal information on the parent given in a text string as follows: pname^pnum count^ccrnum^step name^step description^call depth

Informational Errors

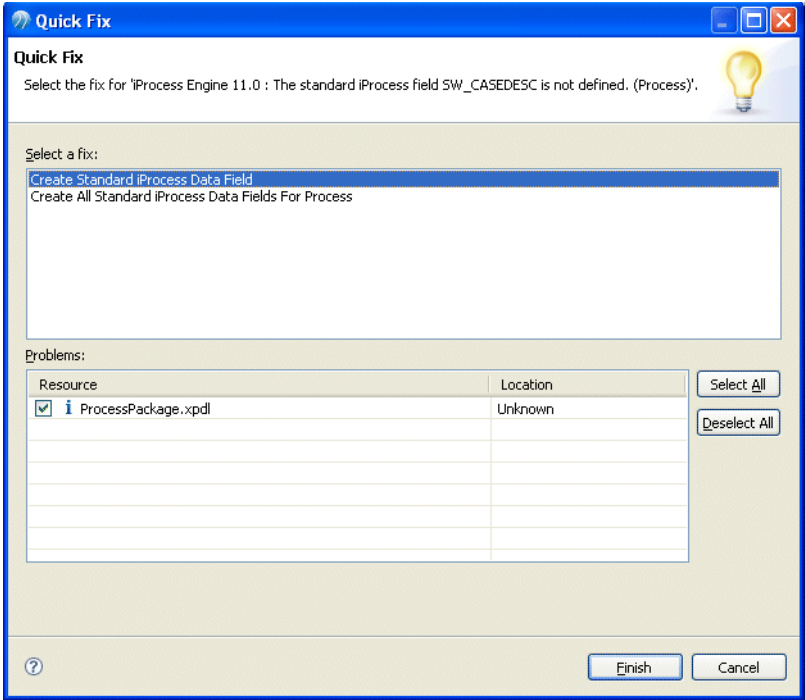
If data fields are not defined for the standard iProcess fields listed in the previous section, the Problems view displays information messages similar to the following:



To correct these errors, do the following:

1. Right-click and select **Quick Fix**.

2. The resulting dialog allows you to create a data field for the selected iProcess field, or for all iProcess fields:



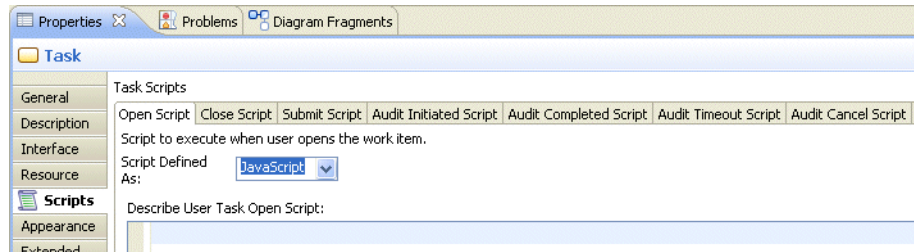
Select a quick fix (either **Create Standard iProcess Data Field** or **Create All Standard iProcess Data Fields For Process**).

3. Select the Packages to which you want to apply the quick fix and click **Finish**.

Creating a Script for a User Task

To add a script to a user task, do the following:

1. Click the user task to which you want to add a script.
2. In the Properties view for the user task, click the **Scripts** tab.



3. Depending on the Destination Environment and Capability selected, there are several script types in the **Script Defined As** list. Select one of the available script types and enter your script in the area provided. You can press Ctrl+Space for Content Assist. When the Task is exported, the corresponding

script objects are created in the iProcess procedure to be used by the Work Item Step (called by the Initial, Keep, and Release) Form Commands:

The screenshot shows the 'Step Definition' dialog box with the 'Definition' tab selected. The 'Form Commands' section includes 'Initial', 'Keep', and 'Release' fields. The 'Release' field contains the command: `winrun("c:\program files\microsoft office\office\winrun",3)`. The 'Step Priority' section includes a 'Base Priority Value (1-999)' field with the value `SW_CP_VALUE`. Below it is the 'Automatic Priority Escalation' section with four sub-fields: 'Increment (negative to raise):' with value `SW_CP_INCREMENT`, 'Number of increments (-1=unlimited):' with value `SW_CP_NUMINC`, 'Increment Period:' with value `SW_CP_INCPERIOD`, and 'Period Type ("m"ins, "h"ours, "d"ays):' with value `SW_CP_PERIODTYP`. The 'Permissions' section at the bottom has three checkboxes: 'Forward' (unchecked), 'Edit' (unchecked), and 'Ignore Case Suspend' (unchecked). At the very bottom are 'OK', 'Cancel', and 'Help' buttons.



Calls to script objects are not created when a process with Task scripts is deployed from TIBCO Business Studio to the iProcess Engine. However, when the same process is exported to the iProcess Modeler, then the calls to script objects are correct.

4. The following table shows the types of Task scripts that you can create. Also displayed on the **Scripts** tab are audit scripts (see [Creating Audit Scripts on page 202](#)).

TIBCO Business Studio User Task	iProcess Procedure Form Command	Result in iProcess Procedure
Open Script	Initial	This command is run when the work item form is opened from the user's Work Queue.
Close Script	Keep	This command is run when the form is returned to the user's Work Queue.

TIBCO Business Studio User Task	iProcess Procedure Form Command	Result in iProcess Procedure
Submit Script	Release	This command is run when the form is released.

5. Once you have created the desired scripts, save the Package that contains the process.



Scripts on a user task have certain foreground functions that are not available on Script Tasks.

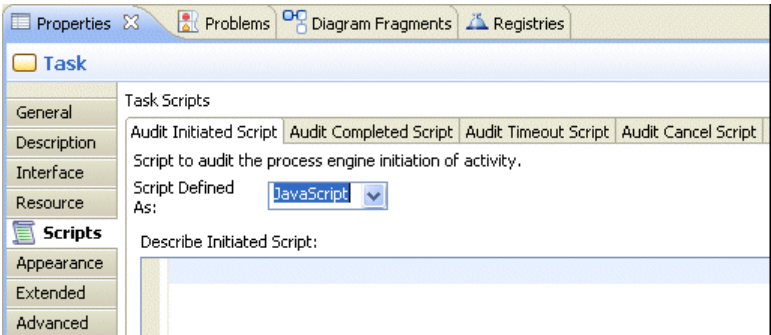
Creating Audit Scripts

For all types of tasks in TIBCO Business Studio you can create auditing scripts. However, for iProcess destinations, auditing scripts only apply to Service Tasks and Script Tasks.

1. Click the **Scripts** tab.
2. Click one of the horizontal tabs. For example:



Only the **Audit Initiated Script** and **Audit Completed Script** tabs are supported for iProcess destinations.



3. In the **Script Defined As** field, select one of the following options: free text, Unspecified or JavaScript.



The area where you describe the script supports the usual text editing assistance such as color syntax highlighting, content assist and error markers. For example, if you want to specify a data field called **Field**, enter the character "F", then press Ctrl+Space. All matching data fields are displayed.

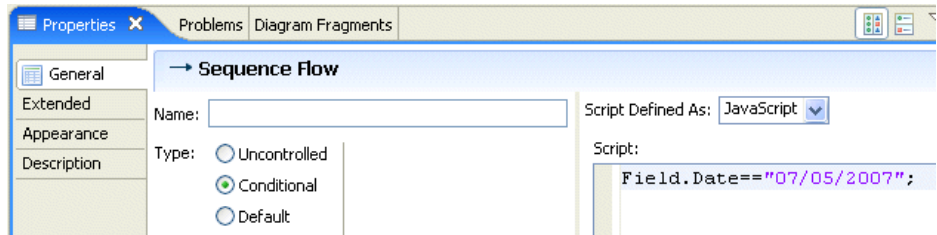
4. Enter the expressions that will be written in the audit trail. Note that only one line of expression that evaluates to a string is allowed.

TIBCO Business Studio User Task	Result in iProcess Procedure
Audit Initiated Script	Defines the expression for the value of an EAI Step Call-Out Initiated message in a case audit trail.

TIBCO Business Studio User Task	Result in iProcess Procedure
Audit Completed Script	Defines the expression for the value of an EAI Step Call-Out Completed message in a case audit trail.
Audit Timeout Script	Script that the process engine runs when the activity times out. No iProcess equivalent.
Audit Cancel Script	Script that the process engine runs when the activity is cancelled. No iProcess equivalent.

Associating a Script with a Conditional Flow

Scripts are associated with a Conditional Sequence Flow in the Properties view. Use a script to set the conditions that determine whether a conditional sequence flow is followed. Only one statement that evaluates to a Boolean value is allowed if you are exporting/deploying to the iProcess Engine. At runtime, this causes the Sequence Flow to be followed only if the condition is met:



The **Script:** area supports the usual text editing assistance such as color syntax highlighting, content assist and error markers (see [Creating a Script Task on page 186](#) for more information).



On the **General** tab, there may a text description of the required script. This description will be converted to comments if you select **JavaScript** from the **Script Defined As** list. If you select **Unspecified** from the **Script Defined As** list, the description is lost. You can however recover the description by pressing **Ctrl + Z**.

Timer Event Scripts

Scripts can be added to Start or Intermediate events in the Properties view for the event. Depending on the Destination Environment selected for the process, there are several script types available from the **Script Defined As** list:

- **Text** - the Business Analyst or person who created the process can use this field to enter text that describes the desired behavior for the script.



On the **General** tab, there may be text comments describing the required script. These comments will be lost if you change the select **JavaScript** from the **Script Defined As** list. If you need to save these comments, copy them before changing the script type.

- **Constant Period** - this allows you to specify the timeout period after the event is initiated using the following time units. For the iProcess Engine Destination Environment, this Constant Period that you specify is translated into a Deadline Expression:

Script Defined As: Constant Period

Specify timeout as offset from event initiation:

Years:	<input type="text" value="0"/>	Hours:	<input type="text" value="0"/>
Months:	<input type="text" value="0"/>	Minutes:	<input type="text" value="0"/>
Weeks:	<input type="text" value="0"/>	Seconds:	<input type="text" value="0"/>
Days:	<input type="text" value="0"/>	Micro Seconds:	<input type="text" value="0"/>

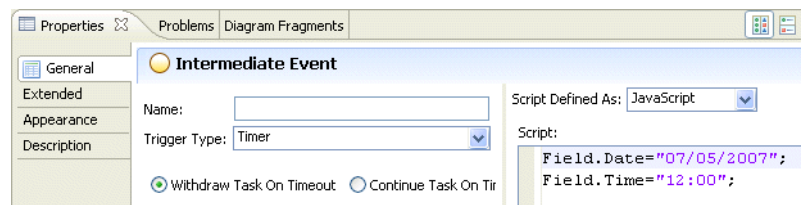


Not all time units are available for all Destination Environments. For example, Seconds and Micro Seconds are not valid in an iProcess Deadline specification, so you cannot specify these units for a process that will be deployed to the iProcess Engine.

- **JavaScript** - this script type is available for the iProcess Engine, Modeler, and Simulation Destination Environments. It allows you to enter JavaScript statements in the space provided.

The script that you specify is limited to two statements. If you specify only one statement, it must evaluate to either a Date or a Time. If you specify two statements, one must evaluate to a Date and the other to a Time. The script area cannot be empty if you want to export or deploy to the iProcess Engine (it can however be empty for the iProcess Modeler Destination Environment).

For example:

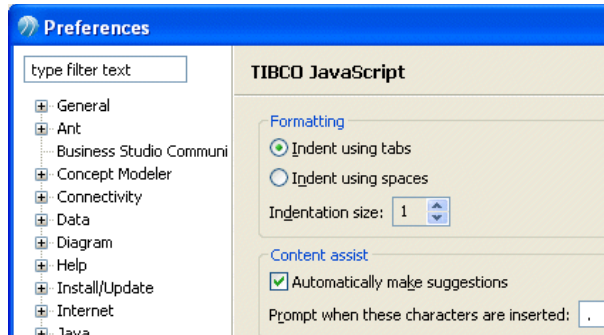


In this case, the event will be fired at the Date and Time specified. If only the Date were present, the event fires immediately when the deadline is created (in the runtime environment) on the Date specified. If only the Time were specified, the event would fire at the specified time on the current date.

Customizing JavaScript Presentation Preferences

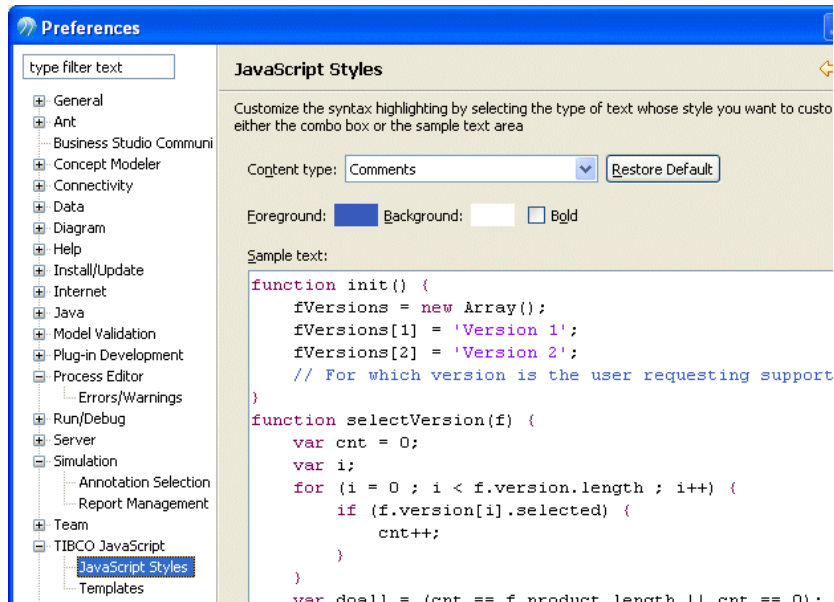
You can customize some of the settings for formatting, content assist, and the templates that are used by JavaScript. To customize the JavaScript settings, do the following:

1. Select **Window > Preferences**.
2. Select **TIBCO JavaScript**. The following dialog is displayed:



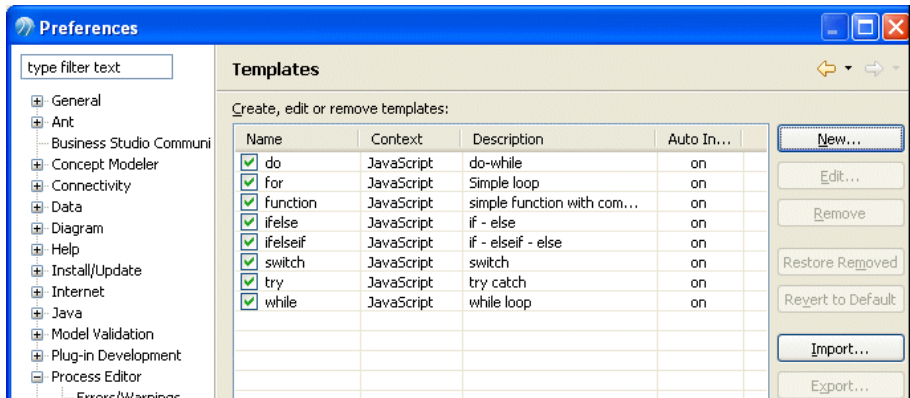
Make any desired changes to the **Formatting** or **Content Assist** sections, then click **Apply**.

3. To customize JavaScript styles, expand **TIBCO JavaScript** and select **JavaScript Styles**. The following dialog is displayed:



From the **Content type** drop-down list, select the type of text whose behavior you want to modify. Make the desired changes and click **Apply**.

- 4. To change the installed templates, expand **TIBCO JavaScript** and select **JavaScript Templates**. The following dialog is displayed:

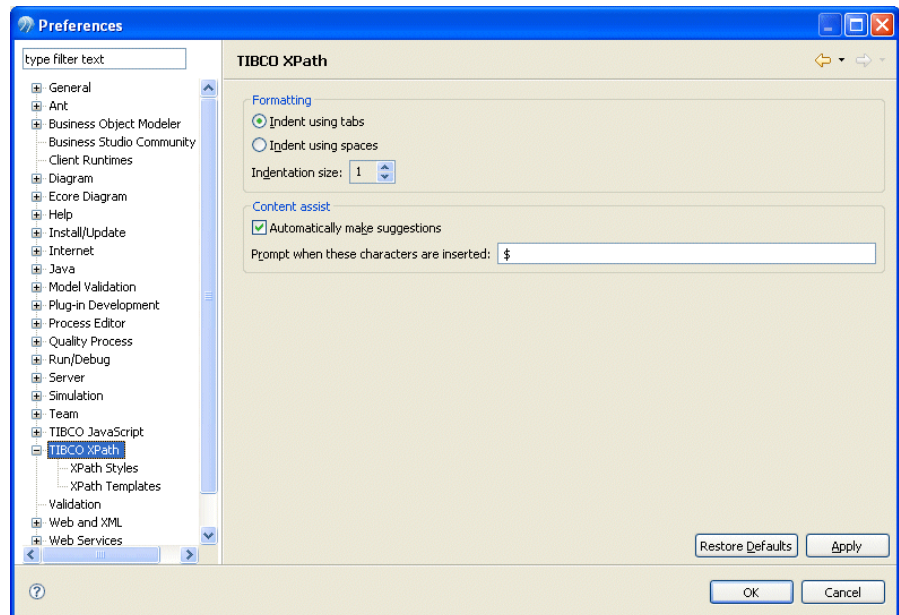


You can edit, modify, create, or import templates using the controls provided. When you have finished making changes, click **Apply**.

Customizing XPath Presentation Preferences

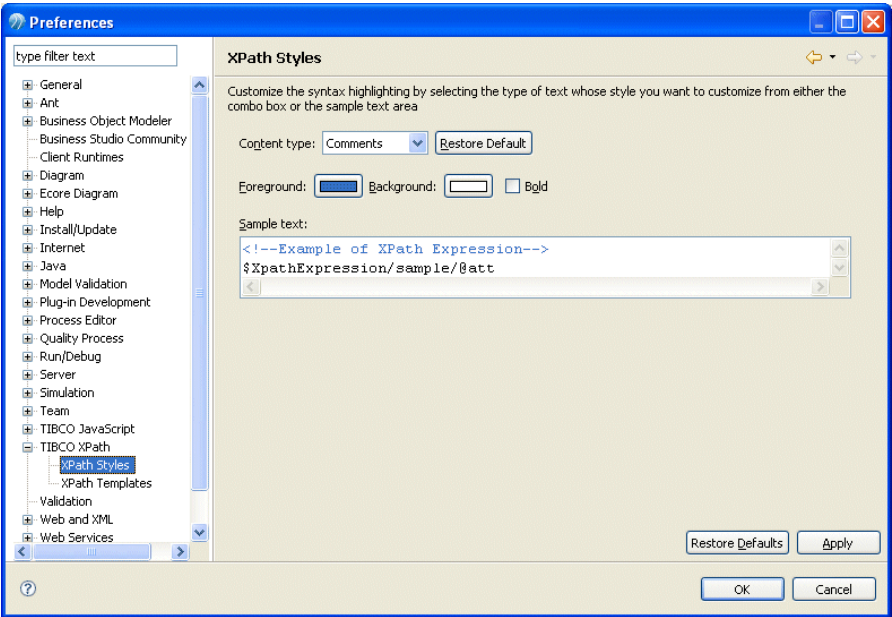
You can customize some of the settings for formatting, content assist, syntax highlighting, and the templates that are used by XPath scripts. To customize the XPath settings, do the following:

1. Select **Window > Preferences**.
2. Select **TIBCO XPath**. The following dialog is displayed:



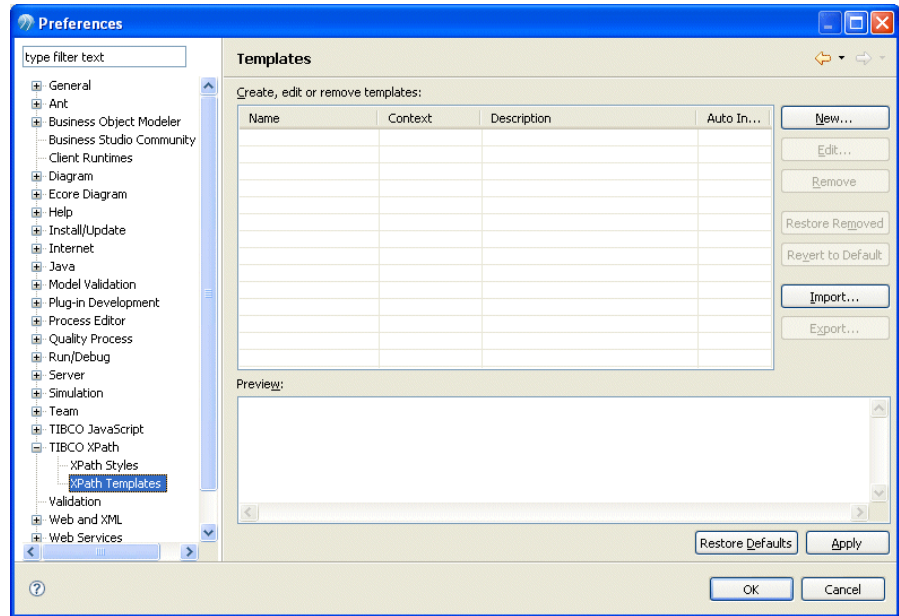
Make any desired changes to the **Formatting** or **Content Assist** sections, then click **Apply**.

3. To customize XPath styles, expand **TIBCO XPath** and select **XPath Styles**. The following dialog is displayed:



From the **Content type** drop-down list, select the type of text whose behavior you want to modify. Make the desired changes and click **Apply**.

4. To change the installed templates, expand **TIBCO XPath** and select **XPath Templates**. The following dialog is displayed:



You can edit, modify, create, or import templates using the controls provided. When you have finished making changes, click **Apply**.

Chapter 7 **Validating Processes**

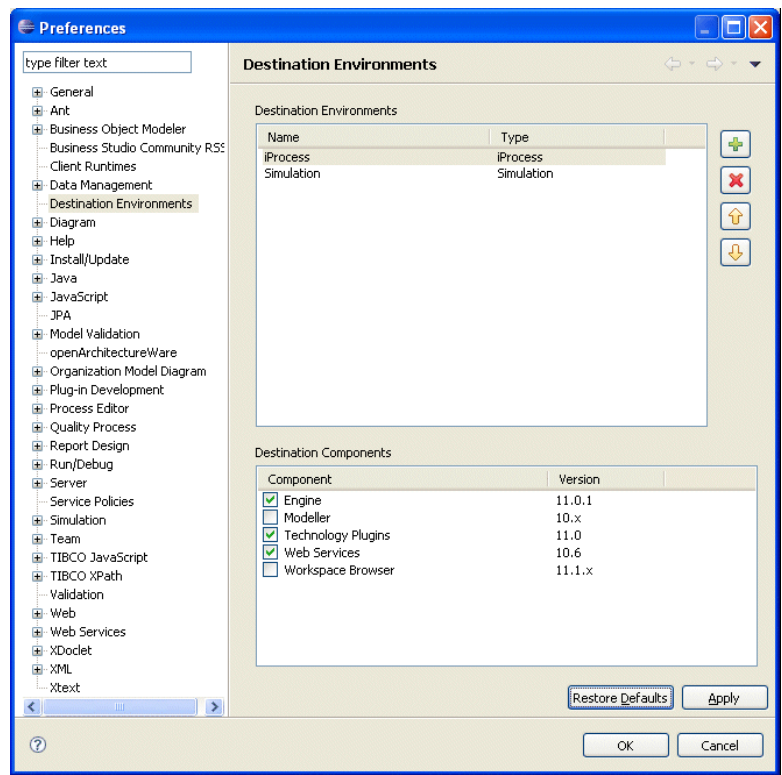
This chapter describes the validation that TIBCO Business Studio performs on processes, and how to correct problems.

Topics

- [Working with Destination Environments, page 214](#)
- [Correcting Validation Errors, page 217](#)
- [Turning off Validation, page 218](#)

Working with Destination Environments

By default, TIBCO Business Studio provides two destination environments, iProcess and Simulation. The specific "destination components" that make up these destination environments can be customized in the Preferences. For example, select **Window > Preferences**, and select **Destination Environments**. Highlighting the **iProcess** destination environment displays the following:



By default the iProcess destination environment contains the Engine, Technology Plug-ins, and Web Services destination components.

You can also create your own destination environments, and import or export your destination preferences as shown in [Tutorial 1: Working with Destination Environments on page 19](#).



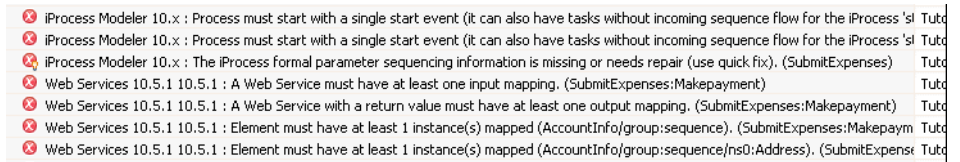
You can delete destination environments using the Preferences dialog. However any process that uses that destination will report a BPMN error in the problems view (**The destination does not exist in this workspace, please create it using the Destination Environments preference page or remove it from the process**).

Destination Components

Either when you create a process, or afterward on the **Destinations** tab, you can select a destination environment that contains destination components. Validation is performed according to the destination components contained in a destination environment.

- If you select **Engine**, you can choose from versions **10.x**, **11.0**, or **11.0.1**. When you save your process, TIBCO Business Studio performs validation to ensure that the process can be directly deployed to the selected version of the TIBCO iProcess Engine. Version 11.0 and Version 11.0.1 allow the use of process interfaces that are converted to I/O templates upon deployment to iProcess (see [Deploying to the iProcess Engine on page 219](#)).
- If you select **Modeler 10.x**, when you save your process, TIBCO Business Studio performs validation to ensure that the process can be imported into the TIBCO iProcess Modeler. For more information see the *TIBCO Business Studio Process Modeling User's Guide*.
- If you select **Technology Plugins**, you can choose from versions 10.7 or 11.0. When you save your process, TIBCO Business Studio performs validation to ensure that the process can be directly deployed to the TIBCO iProcess Engine with these versions of the Technology Plugins.
- If you select **Web Services**, you can choose from versions 10.5.1 or 10.6. When you save your process, TIBCO Business Studio performs validation to ensure that the process can be directly deployed to the TIBCO iProcess Engine with these versions of the TIBCO iProcess Web Services Plug-ins.
- If you select **Workspace Browser**, when you save your process, TIBCO Business Studio performs validation to ensure that the forms in the process can be directly deployed to version 11.1.x of the TIBCO iProcess Workspace (Browser).

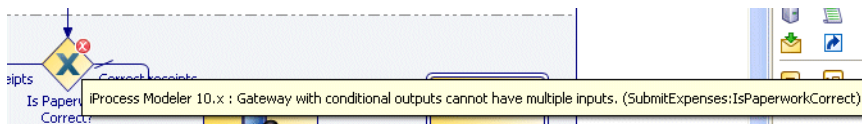
Any error messages resulting from this validation will be displayed in the Problems View and indicate which parts of your process need to be changed. For example:



The screenshot shows the Problems View with the following error messages:

- ❌ iProcess Modeler 10.x : Process must start with a single start event (it can also have tasks without incoming sequence flow for the iProcess 'sl' Tut...
- ❌ iProcess Modeler 10.x : Process must start with a single start event (it can also have tasks without incoming sequence flow for the iProcess 'sl' Tut...
- ❌ iProcess Modeler 10.x : The iProcess formal parameter sequencing information is missing or needs repair (use quick fix). (SubmitExpenses) Tut...
- ❌ Web Services 10.5.1 10.5.1 : A Web Service must have at least one input mapping. (SubmitExpenses:Makepayment) Tut...
- ❌ Web Services 10.5.1 10.5.1 : A Web Service with a return value must have at least one output mapping. (SubmitExpenses:Makepayment) Tut...
- ❌ Web Services 10.5.1 10.5.1 : Element must have at least 1 instance(s) mapped (AccountInfo/group:sequence). (SubmitExpenses:Makepaym Tut...
- ❌ Web Services 10.5.1 10.5.1 : Element must have at least 1 instance(s) mapped (AccountInfo/group:sequence/ns0:Address). (SubmitExpenses Tut...

Problem markers are also displayed against objects in the Process Editor. Positioning the pointer over the problem marker displays the reason for the error. For example:



When you import a process into the TIBCO iProcess Modeler or deploy a process to the iProcess Engine, the original objects (from the TIBCO Business Studio version) are mapped into objects that the TIBCO iProcess Modeler and TIBCO iProcess Engine support. For more information about these mappings, see [Object Mapping on page 112](#).

Correcting Validation Errors

Any problems that result from validation are shown in the Problems view. To correct the problem do one of the following:

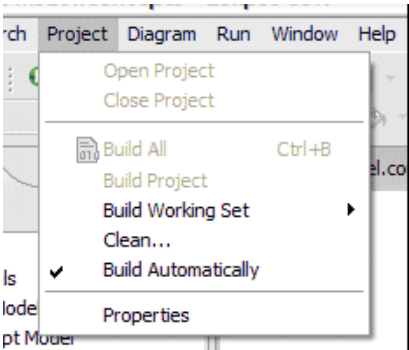
- Right-click the problem and select **Quick Fix** (if enabled for the current problem). This gives you the option of having TIBCO Business Studio correct the problem for you.
- Right-click the problem and select **Go To**. This displays the process in the Process Editor, allowing you to correct the problem.



You can customize the validation that is performed in the Process Editor. For more information, see *TIBCO Business Studio Process Modeling User's Guide*.

Turning off Validation

When **Project > Build Automatically** is deselected, in-memory validation is disabled:



When this is done, any existing problems remain in the Problems view and markers remain against the resources. To clear these problems, select **Project > Clean**, and select the relevant projects. Subsequently, you can select **Project > Build All** to rebuild all processes, causing any applicable problem markers to reappear.

Chapter 8

Deploying to the iProcess Engine

This describes how to package and deploy a process.

Topics

- [Deployment Overview, page 220](#)
- [Preparing a Process for Deployment, page 222](#)
- [Synchronizing your Step Index, page 223](#)
- [Creating a New Workspace Server, page 231](#)
- [Connecting to a Server, page 232](#)
- [Deploying a Module, page 235](#)
- [Starting a Case From Within TIBCO Business Studio, page 240](#)
- [Opening the TIBCO iProcess Workspace \(Browser\), page 242](#)
- [Managing Deployed Modules, page 244](#)
- [Disconnecting from the Server, page 246](#)

Deployment Overview

TIBCO Business Studio allows you to deploy a resource (represented in TIBCO Business Studio as a **Module**) on a local or remote system (represented in TIBCO Business Studio as a **Server**). In the iProcess Engine, a Module corresponds to a process with a Destination Environment that includes the **Engine** destination component, and a Server represents a running TIBCO iProcess Engine where you want to deploy the process.

The iProcess Engine must be running; you cannot start and stop the Server from within TIBCO Business Studio. However, you can connect and disconnect from the Server. Once connected, you can manage Modules on the Server. For example, you can Release, Withdraw, and Undeploy Modules.



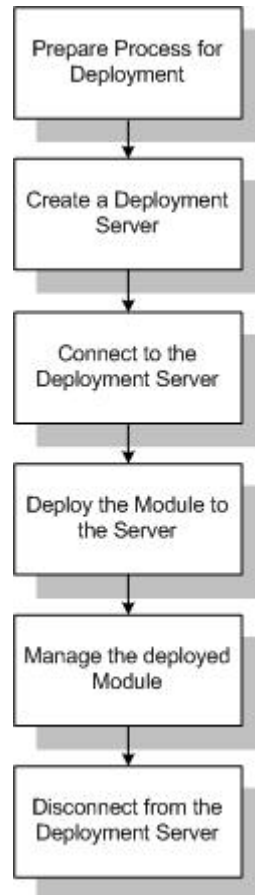
Direct Deployment to the TIBCO iProcess Engine is a one-way operation. Information can be lost if you deploy to the iProcess Engine, modify the process, re-import to TIBCO Business Studio, deploy again, and so on. This is because the TIBCO Business Studio process description contains more information than merely the execution information required for iProcess Modeler/iProcess Engine, and this extra information can be lost upon re-importing. For this reason, you should regard the import of an iProcess Modeler XPD into TIBCO Business Studio as a one-off activity to evaluate or migrate to TIBCO Business Studio.

When you create a deployment server, in addition to the connection details for the iProcess Engine, you can also specify connection details for the following optional components:

- **MBean Servers** If you plan to deploy a process that contains service tasks that use logical names (system participants) for any of the following, you must specify the MBean servers that will be used to retrieve the runtime names that will replace the logical names used within TIBCO Business Studio:
 - URL aliases for a WSDL file
 - BusinessWorks endpoint names
 - Security profiles
- **TIBCO iProcess Workspace (Browser)** You can specify connection details for the TIBCO iProcess Workspace (Browser) so that you can connect to the iProcess (Browser) from within TIBCO Business Studio.
- **Forms Deployment Servers** If you are developing forms using TIBCO Business Studio Forms, you can create a WebDAV deployment server for the deployment of forms so that when you deploy the process, the forms are also deployed.

Both the iProcess Workspace (Browser) and Forms deployment server can either be created as part of the main iProcess server, or separately, depending on your requirements. For example, if you are only responsible for developing forms, you can create a server that only contains the WebDAV server. However, if you are responsible for deploying processes and forms, you would create a server that includes iProcess and WebDAV connection information.

The following diagram shows the steps for deploying a process the first time:



Once you have deployed a process, if the required server has been registered and the Module created, you do not need to repeat all of these steps each time you deploy the process; simply connect and deploy a new version of the process.

Preparing a Process for Deployment

This section describes the steps that must be performed to prepare a process for deployment.



You cannot choose to import some processes in a process and sub-process family into TIBCO Business Studio, and then only partially re-deploy them. If you have an iProcess Modeler process in a 'family' (that is, related via sub-process invocation/process interface implementation) then to migrate to TIBCO Business Studio to edit the processes, you must import the entire process family and on the first occasion at least, re-deploy the entire family.

Package the Process

Packaging happens automatically and is a prerequisite to deployment. By default, **Project > Build Automatically** is selected. This means that when you save your Package (XPDL2), a version of it is also saved to XPDL1 ready for deployment to the iProcess Engine. The process is only available for deployment using the Deployment wizard if its Destination Environment includes the iProcess destination component (see [Working with Destination Environments on page 214](#)), and it has no errors.

If you choose to package your project manually (for example, because of resource or memory considerations), de-select **Project > Build Automatically**. When you want to package your project, select either **Project > Build All** or **Project > Build Project**.



De-selecting **Project > Build Automatically** turns off in-memory validation of the project.

Verify Package/Process Naming

The name of a process in TIBCO Business Studio is truncated to the first eight alphanumeric characters when it is deployed to the iProcess Engine. If you are planning to deploy a process to the iProcess Engine, you should ensure that the first eight alphanumeric characters of the process name in TIBCO Business Studio are unique and do not contain ^ (caret) or , (comma) characters.



If you deploy a process that has the same name as a process that has already been deployed, a new minor version is created. For this reason, you should be aware of the names of existing procedures in the target deployment environment to ensure that you do not unintentionally overwrite an existing procedure.

The Package Name of the process you are deploying becomes the Library Name upon deployment to the iProcess Engine.

Set the Destination and Correct Problems

To deploy a process to a TIBCO iProcess Server, do the following:

1. Create the process and set the Destination Environment to **iProcess**.
2. Resolve all of the problems reported in the Problems view.



TIBCO Business Studio prevents you from deploying a process with errors. If you are attempting to deploy a process for the first time and it contains errors, it will not be shown in the Deployment wizard and you cannot deploy it. If the process has already been deployed, the version that was previously deployed will be available in the Deployment wizard, but the version with errors will not be displayed/deployable.

3. If you cannot see the Deployment view, switch to the Solution Design capability using the following menu.



4. As described in [Package the Process on page 222](#), Packaging occurs automatically. However, if you have de-selected **Project > Build Automatically**, make sure that you select either **Project > Build All** or **Project > Build Project** to manually package your process.



You cannot deploy processes if you have turned off the **iProcess Module Builder** in the Properties of the project. To check this, right-click the project in the Project Explorer, select **Properties** and click **Builders**.

Synchronizing your Step Index

If you have an XPDL which you have previously imported into TIBCO Business Studio, it may have step sequence information missing. You will see an error reported on the process(es), and need to correct this as follows:

1. Right-click the error in the Problems view and select **Quick Fix**.

2. You will then be given the choice of running the quick fix, or running a wizard to correct your problem, depending on your process.
 - If there are no parallel joins in your process it will automatically assign random step indexes without further intervention.
 - If you have parallel joins it is important for the consistency of existing cases that the step index is consistent with that previously deployed so you will need to proceed to [step 3](#).
3. In the wizard:
 - If you do not have any live cases, select the appropriate option in the first wizard page and it will automatically assign step indexes..
 - Otherwise, if the process has been deployed and you have live cases, the next wizard page will assist you to extract a file from the server that contains the existing step indexes for tasks, and will assign these indexes to the tasks. If there is a mismatch between steps in the extract file and the tasks in you process you must verify that the correct file for your process was extracted from the server in order to continue .



If you have just performed an import using TIBCO Business Studio 3.4.0 then synchronizing the step indexes will not be an issue.

Creating a New iProcess Server

To create a new server, right-click **Deployment Servers**, select **New > Server**, and proceed with the steps in the following sections.

Specify the Server Name and Runtime

In the **Server and Server Runtime** dialog, enter the following:

1. Enter a **Server Name** (to identify the server within TIBCO Business Studio).
2. Accept **iProcess Engine Server** as the **Runtime** Environment.
3. Click **Next**.

Enter the Runtime Server Parameters

In the **Runtime Server Parameters** dialog, enter the following:

1. Enter the runtime server parameters.
 - **Host** Enter **localhost** if the server is on your local machine; otherwise enter the machine name or IP address of the computer where the iProcess Engine is installed.
 - **MBean Port** The port number that the iProcess Engine uses for the JMX engine. By default the Port is **10025**. This was configured during installation of the iProcess Engine and stored in the **SWJMXConfig.port** entry in the **SWDIR\etc\swjmx.properties** file. If you cannot determine the port number, contact your iProcess Administrator.



You now can run TIBCO Business Studio through a firewall, as long as you configure and open the necessary ports.

By default, TIBCO Business Studio uses port **10025** for the JMX engine and also uses port **10026** as an RMI port. The user will not be able to deploy if either of these ports are closed (which one could be for example if a customer was using a VPN).

So while **swjmx.properties** can define the JMX port, be aware you need to open that port as well as that port +1 as an RMI port.

- **Path** Specifies the path on the server to the JMX service, where the JMX objects are stored. The default for **Path** is **/server** (this is the default iProcess Engine setting). If you have changed the default location of the JMX service,

you must ensure that you also change **Path** in TIBCO Business Studio so that both settings match.

- **MBean Name** The MBean Name is set in the configuration file `SWDIR\etc\swjmx_config.xml`. TIBCO Business Studio uses the default name **TIBCO:iProcessDeployment=default**. If you have changed the default MBean Name setting for the iProcess Engine by editing the configuration file, you must ensure that you also change **MBean Name** in TIBCO Business Studio so that both settings match.
 - **MBean Connector** Reserved for future use.
 - **iProcess Node Name** Used for opening the TIBCO iProcess Workspace (Browser) from within TIBCO Business Studio if direct login is enabled in the workspace client. *Note that this setting is not tested by the Test Connection button on this dialog.* For more information about direct login, see *TIBCO iProcess Workspace (Browser) Configuration and Customization*.
 - **iProcess Objects Server TCP Port** Specify the TCP port of the iProcess Objects Server TCP port. Set this parameter only if you are opening the TIBCO iProcess Workspace (Browser) from within TIBCO Business Studio and direct login is enabled in the workspace client. *Note that this setting is not tested by the Test Connection button on this dialog.* For more information about direct login, see *TIBCO iProcess Workspace (Browser) Configuration and Customization*.
 - **Director** Specify whether the iProcess Objects Server is a director. Set this parameter only if you are opening the TIBCO iProcess Workspace (Browser) from within TIBCO Business Studio and direct login is enabled in the workspace client. *Note that this setting is not tested by the Test Connection button on this dialog.* For more information about direct login, see *TIBCO iProcess Workspace (Browser) Configuration and Customization*.
 - **Username** Valid iProcess Engine user with either the PRODEF or ADMIN permission that can connect to the iProcess Engine (for example, `IPEADMIN`).
 - **Password** Password for the user connecting to the iProcess Engine.
 - **Save Password** Select this check box to save the password that you entered. This means you do not need to enter a password when connecting to the server. However, for security reasons, you can choose not to save the password, in which case you must enter the password each time you connect to the server.
 - **Repository Type** Select **Workspace** to allow the Module to be deployed from your Eclipse workspace. The **Local Folder** option is not supported.
2. Click **Next** and continue with the following section.

Enter the MBean Server Configuration Parameters (Optional)

In the **Other Server Configurations** dialog, specify the details for any required MBean servers. You need to configure additional MBean servers if:

- You specified a logical name (System participant) as a URL alias for a WSDL file that will be obtained remotely. Upon deployment, this logical name must be mapped to an actual endpoint name in the destination environment. Use the **WS Config** tab to specify the connection that will provide this information.
- You specified a logical name (System participant) for a BusinessWorks endpoint name. In this case the logical name (System participant) must be mapped to an endpoint name in BusinessWorks. Use the **BW Config** tab to specify the connection that will provide this information.
- You have specified a logical name for a security profile. Upon deployment, this logical name must be mapped to an actual security profile in the destination environment. Use the **Security Profile Config** tab to specify the connection that will provide this information.

New Server

Other Server Configurations
Set parameters of the other servers hosting the WS and BW endpoint MBeans

WS Config | BW Config | Security Profile Config

WS MBean Server Parameters:

Connect to Server: ☐

Host:

MBean Port:

Path:

MBean Name:

MBean Connector:

Username:

Password:

☐ Save password

Saved passwords are stored on your computer in a file that is difficult, but not impossible, for an intruder to read.

Use the tabs at the top of the dialog to display the parameters for the relevant server (web service, BusinessWorks, or security profile configuration).

1. Select the **Connect to Server** checkbox so that when you connect to the deployment server, TIBCO Business Studio also connects to the MBean servers you are going to configure. Enter the runtime server parameters.
 - **Host** For web services, enter the machine name or IP address where Jetty is located (by default this is the same location as the iProcess Engine server).



If you are using URL aliases or security profiles, the hostname that you enter must match the server name in the **TibcoiProcessWorkspace\esai_websvcs\urlaliasmanager.properties** file, in the following property:

```
URLAliasManager.aliasHosts=servername:port
```

If you do not enter the exact string listed in this property, you cannot view the URL alias or security profile information in the EAI callout definition using TIBCO iProcess Modeler.

For BusinessWorks, enter the machine name or IP address where BusinessWorks is located.

- **MBean Port** The port number that the iProcess Engine uses for the JMX engine. For the **WS Config** and **Security Profile Config** tabs, the default Port is **10010**. For the **BW Config** tab, the default port is **10021**. This was configured during installation of the iProcess Engine. If you cannot determine the port number, contact your iProcess Administrator.
 - **Path** Specifies the path on the server to the JMX service, where the JMX objects are stored. The default for **Path** is **/server** (this is the default iProcess Engine setting). If you have changed the default location of the JMX service, you must ensure that you also change **Path** in TIBCO Business Studio so that both settings match.
 - **MBean Name** Do not change this setting.
 - **MBean Connector** Do not change this setting.
 - **Username** Enter the username that you want to use to connect to the server.
 - **Password** Enter the password that corresponds to the username you entered.
2. Test the connection by clicking the **Test Connection** button.
 3. If you want to enter iProcess Workspace (Browser) configuration details, click **Next**; otherwise click **Finish**.

Enter the Workspace (Browser) Server Configuration Details

In the Workspace (Browser) Server dialog, enter the connection details for connection to the TIBCO iProcess Workspace (Browser) and connection to the TIBCO Forms WebDAV server.



Web-based Distributed Authoring and Versioning (WebDAV) is a protocol used for publishing and managing content to web servers. TIBCO Business Studio Forms uses WebDAV to publish forms.

These parameters were entered during installation of the TIBCO iProcess Workspace (Browser). They are stored in the **config.xml** file in the following directory:

ClientInstallDir\JSXAPPS\ipc

where *ClientInstallDir* is the directory in which the iProcess Workspace (Browser) is installed. For more information, see *TIBCO iProcess Workspace (Browser) Configuration and Customization*.

New Server

Workspace (Browser) Server
Configuration for deploying Forms to and connecting to the Workspace (Browser)

iProcess Workspace (Browser) Server

Connect to server: ☒

Action Processor URL:

iProcess Client URL:

Forms Deployment URL:

Forms Deployment Username:

Forms Deployment Password:

☒ Save password

Saved passwords are stored on your computer in a file that is difficult, but not impossible, for an intruder to read.

1. Select the **Connect to Server** checkbox so that when you connect to the deployment server, TIBCO Business Studio also connects to the Workspace (Browser) server you are going to configure. Enter the server parameters as follows:
 - **Action Processor URL** Specify the URL to the Action Processor to which the iProcess Workspace (Browser) connects when it is started.
 - **iProcess Client URL** Specify the URL to the iProcess Workspace (Browser) client application.
 - **Forms Deployment URL** Specify base URL of the WebDAV server where forms are published. This can be found in the **webDAVRoot** parameter in the **config.xml** file (see *TIBCO iProcess Workspace (Browser) Configuration and Customization*).
 - **Forms Deployment Username** Username for connection to the WebDAV server (optional).
 - **Forms Deployment Password** Password for connection to the WebDAV server (optional).
2. Test the connection by clicking the **Test Connection** button.
3. Click **Finish**.

Creating a New Workspace Server

To create a server that connects to the iProcess Workspace (Browser), but not to an iProcess Engine, do the following:

1. In the Deployment view, right-click **Deployment Servers**, select **New > Server**.
2. Enter a name, select **iProcess Workspace (Browser) Server**, and click **Next**.
3. Enter the server parameters that are described in [Enter the Workspace \(Browser\) Server Configuration Details on page 229](#).
4. Click **Finish**. The new server is displayed in the Deployment view and you can connect to it as described in [Opening the TIBCO iProcess Workspace \(Browser\) on page 242](#).

Connecting to a Server

You can connect to a server you have created as follows:



If you encounter any problems connecting to a Server, check the error log by selecting **Help > About TIBCO Business Studio**. From the resulting dialog, click **Configuration Details** then click **View Error Log**. Also restart TIBCO Business Studio before attempting to reconnect to the Server.

1. In the Deployment view, expand **Deployment Servers**.



Any MBean Servers that you configured in [Set the Destination and Correct Problems on page 223](#) are not displayed in the Deployment view. However TIBCO Business Studio automatically connects to any MBeans servers that you configured with the **Connect to Server** checkbox selected.

2. Right-click the server name of the server you are connecting to and select **Connect**.



The username and password you entered when you created the server is authenticated on the deployment server to prevent you from deploying a process to a server which you do not have authorization to use.

3. When you have connected, the Properties view for the server displays Connected as the Server State:

The screenshot shows the 'Properties' view in TIBCO Business Studio. The 'Advanced' tab is selected, and the server 'Jan (Connected)' is chosen. The table below lists the properties of this server.

Property	Value
Base	
Name	Jan
Misc	
Workspace Modules	
Server	
Client Runtime	
Runtime	iProcess Engine Server
Server State	Connected

Troubleshooting

If you have problems connecting to an iProcess deployment server, use this section to troubleshoot the server connection.

- Ensure that you always start iProcess *before* TIBCO Business Studio.
- If you restart the deployment server while Studio is connected to the server, you must disconnect and then restart TIBCO Business Studio before reconnecting.
- Ensure use of the specified port is allowed through firewall settings.
- Enter **netstat** in a command window to:
 - Make sure that the JMX port defined in the *SWDIR/etc/swjmx.properties* is in a status of 'LISTENING' for incoming client requests.
 - Check that the JMX port is not in use, especially if this is a first time connection. If the port is already in use, change the JMX port settings on both the server and TIBCO Business Studio.
- Check passwords have not been changed by retyping them in the server properties - many companies enforce regular password changes.
- Check the Eclipse error log for additional information.

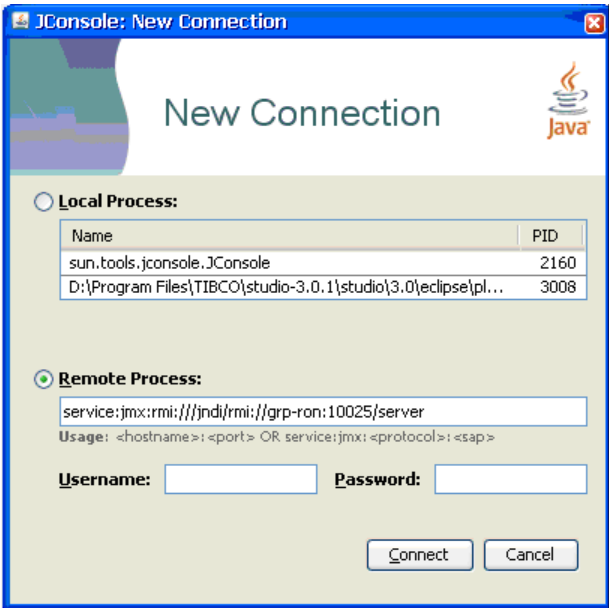
Using JConsole to Connect to an iProcess Server

You can connect to an iProcess server using the Java Management and Monitoring Console (JConsole) to diagnose problems. For example, you can use JConsole to ensure that JMX MBean server is running and presenting deployment MBeans.

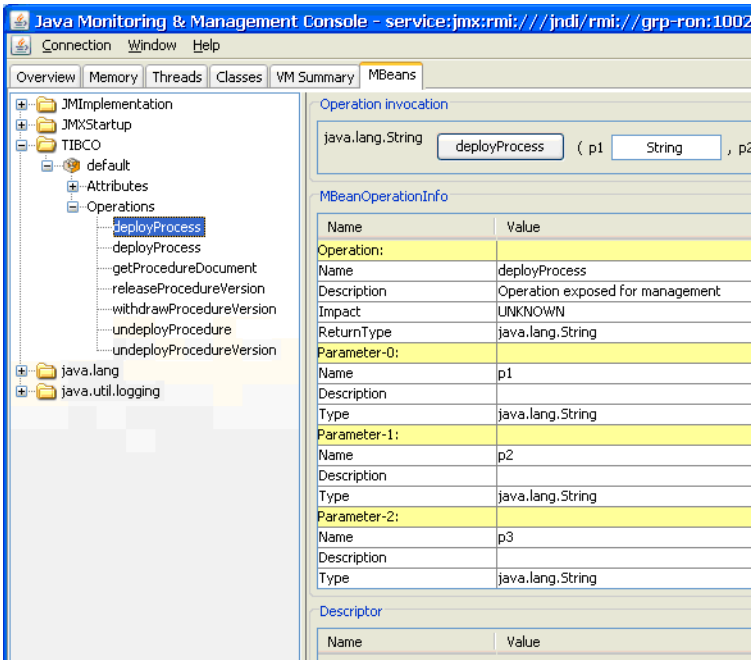
1. Start JConsole:
 - If the path to the **jconsole.exe** file is in your system path, enter **jconsole** in a command window.
 - Run **jconsole.exe** from *JDK_HOME\bin*, where *JDK_HOME* is the directory in which the JDK software is installed.
2. Connect using a remote process URL in the form:


```
service:jmx:rmi:///jndi/rmi://host:port/server
```

For example:



3. Use the tools provided. For example, the following tab shows MBeans information:

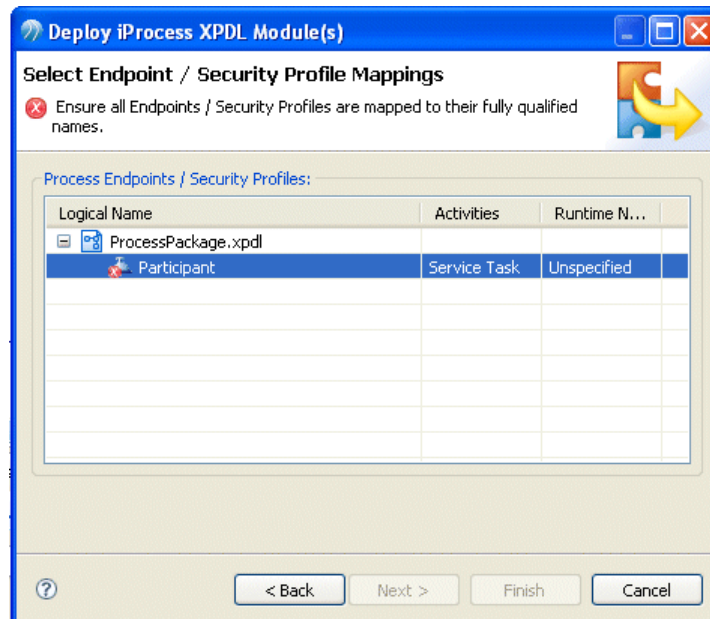


Deploying a Module

Once you have connected to a Server, you can deploy a process to an iProcess Engine either by dragging its package file onto the deployment server, or using the right-click menu on the server.

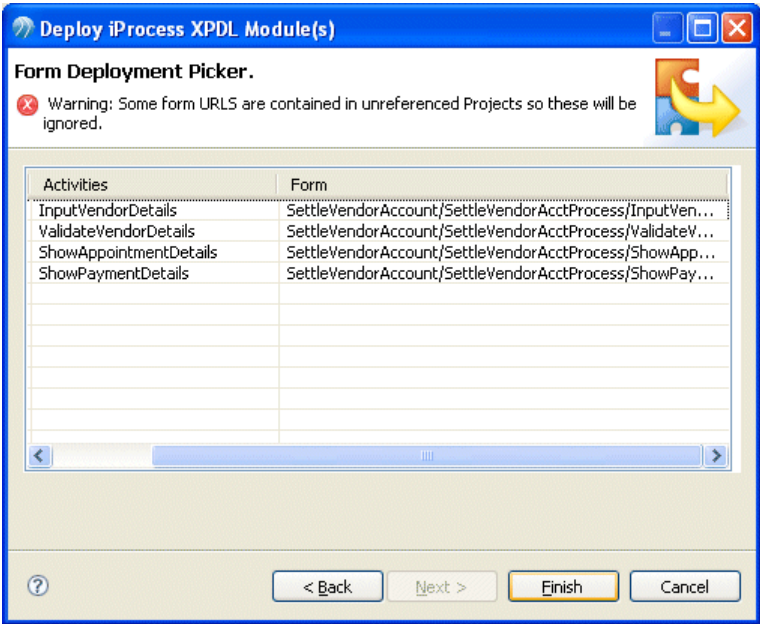
Drag and Drop Deployment

1. Ensure that you have prepared the process for deployment (including setting the destination environment).
2. Drag the XPDL file that contains the process from the Project Explorer to the Deployment view and drop it onto the connected server.
3. If any Service Tasks specify endpoint names or security aliases, the logical name (specified in TIBCO Business Studio) must be mapped to one of the runtime names retrieved from the server that was set up when the deployment server was created.



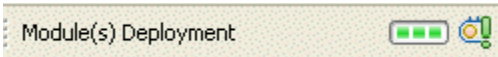
After completing any necessary mappings, click **Next**.

4. If the processes you are deploying contain user tasks that point to forms generated by TIBCO Business Studio Forms, the following dialog is displayed:



This dialog allows you to select which forms you want deployed (by default, all available forms are deployed). Click **Finish**.

5. The deployment progress is shown in the lower right of the TIBCO Business Studio window. When deployment finishes successfully, a message and symbol are displayed:

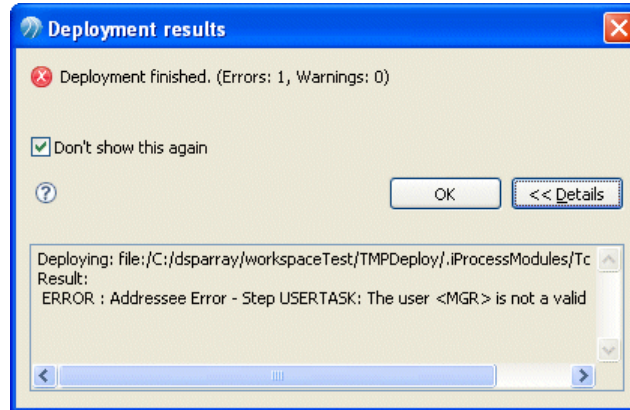


If errors are reported, the following symbol is displayed:



Clicking the symbol (either the exclamation displayed for a successful deployment or the error marker for a deployment that had errors) displays a

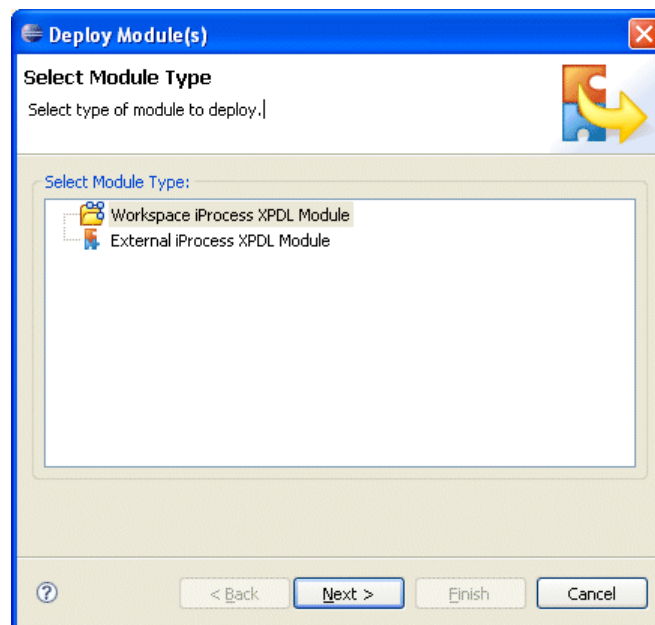
dialog that shows the results of the deployment. For example, the following deployment included a participant that was not defined as a user in iProcess.



6. After a successful deployment, right-click the server and select **Refresh**. The process appears in the Deployment view.

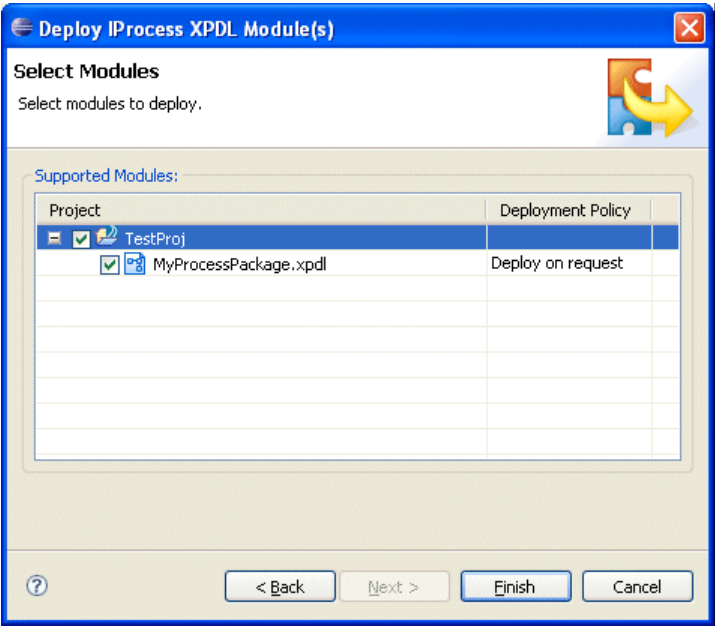
Deploying from the Server Menu

1. Right-click the Server on which you want to deploy and select **Deploy Module**. The following dialog is displayed:



2. Select **Workspace iProcess XPDL Module**, and click **Next**.

3. You are prompted to select the package that contains the process you want to deploy.



If the package and process you want to deploy is not displayed, ensure that you selected the correct destination environment.

You can also select the Deployment Policy for the Module - either **Deploy on request** which deploys the Module when you explicitly choose to do so, or **Deploy on save** which deploys the Module whenever a new version of the Package is saved/packaged.

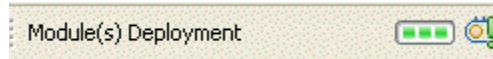


If you select **Deploy on save**, but have de-selected **Project > Build Automatically**, subsequent Save operations on the module do not trigger auto deployment. To deploy, you must right-click the Module and select **Deploy Module**. Then in the subsequent dialog click **Run Build**. When this is done, you can select Modules for deployment.

4. If any Service Tasks specify endpoint names or security aliases, the logical name (specified in TIBCO Business Studio) must be mapped to one of the runtime names retrieved from the server that was set up when the deployment server was created.

After completing any necessary mappings, click **Next**.

5. If the processes you are deploying contain user tasks that point to forms generated by TIBCO Business Studio Forms, a dialog is displayed that allows you to select which forms you want deployed (by default, all available forms are deployed).
6. Click **Finish**. The deployment progress is shown in the lower right of the TIBCO Business Studio window. When deployment finishes successfully, a message and symbol are displayed:



If errors are reported, the following symbol is displayed:



It is sometimes possible that a Module is deployed despite the process having Deployment error messages. For example, deploying an Email Service Task that specifies a recipient that does not exist on the deployment server results in deployment error messages. However, because the problem is easy to remedy, the Module is deployed despite the error.

Starting a Case From Within TIBCO Business Studio

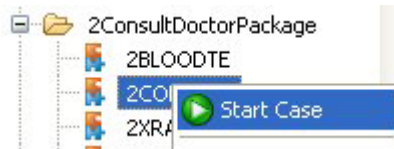
If you entered the connection details for TIBCO iProcess Workspace (Browser) as part of configuring an iProcess server, you can start cases from within TIBCO Business Studio.



Even if you can connect to the iProcess server and deploy processes, you should confirm the settings of runtime server parameters **iProcess Node Name** and **iProcess Objects Server TCP Port**. This is important because these settings are not required for deployment, but they are required to start a case as described in this section. To view these settings, right-click the server and select **Properties**. In the resulting dialog, select **Server General**.

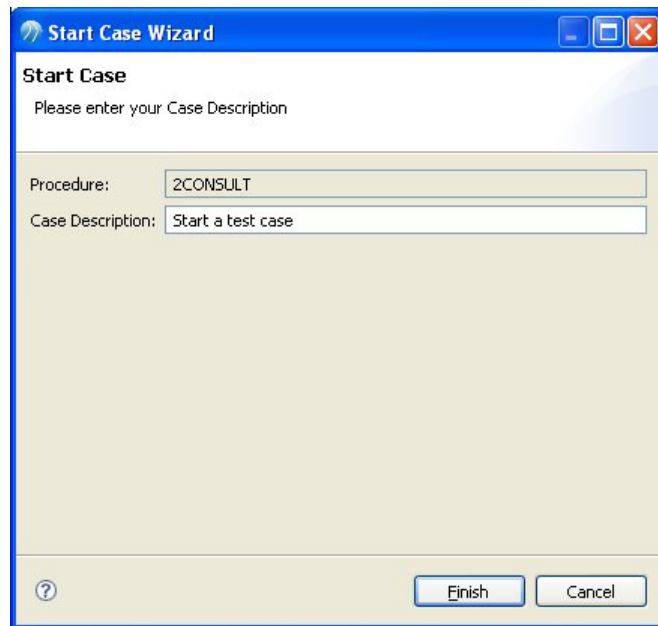
If you are unsure of the correct settings for these parameters, consult your system administrator.

1. Create an iProcess server as described in [Synchronizing your Step Index on page 223](#), making sure to configure a workspace server as described in [Enter the Workspace \(Browser\) Server Configuration Details on page 229](#).
2. Connect to the iProcess server.
3. Expand the server to show the deployed processes.
4. Right-click a process and select **Start Case**:



When you right-click a sub-process, the **Start Case** menu option is not available. The **Start Case** menu option is available for deployed process interfaces, but attempting to deploy these fails.

5. In the resulting dialog, enter a case description and click **Finish**.



6. A progress dialog is displayed, and when the case has been started, a dialog similar to the following is displayed:



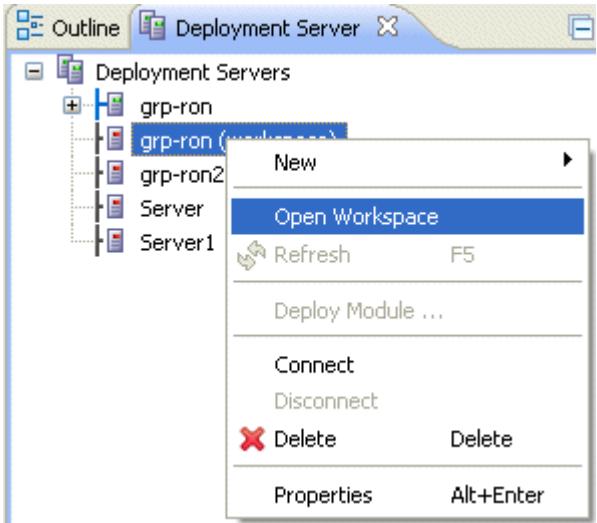
Opening the TIBCO iProcess Workspace (Browser)

If you entered the connection details for TIBCO iProcess Workspace (Browser) either for an iProcess server or a standalone workspace server, you can connect to the iProcess Workspace (Browser) as follows:



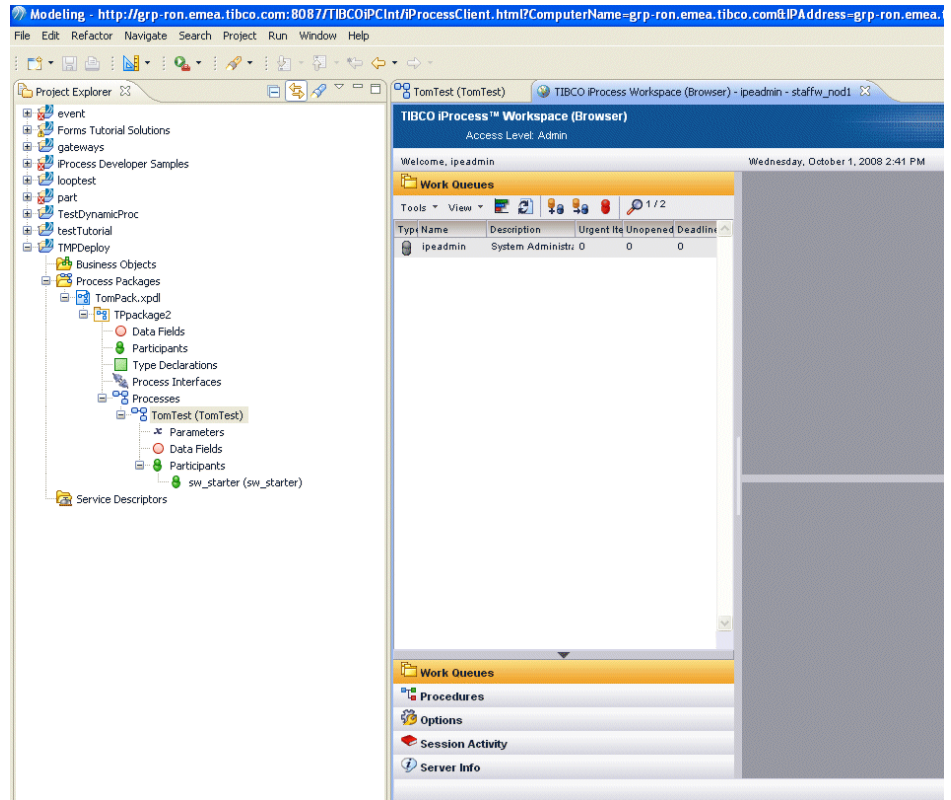
To open the iProcess Workspace (Browser) from within TIBCO Business Studio, you must have direct login enabled (see *TIBCO iProcess Workspace (Browser) Configuration and Customization*). You must also have entered the connection details for the iProcess Workspace (Browser) as part of [Synchronizing your Step Index on page 223](#) or [Creating a New Workspace Server on page 231](#).

1. Right-click the server and select **Open Workspace**.



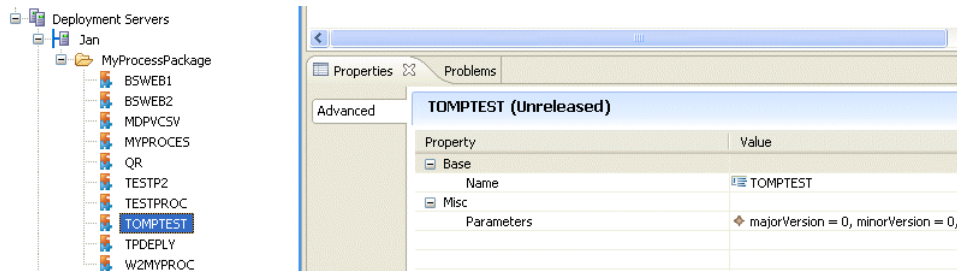
2. If when you configured the workspace server connection settings you did not elect to save the user name and password, you are prompted to enter them.

3. After connecting, the iProcess Workspace (Browser) is displayed:



Managing Deployed Modules

When a Module is first deployed to the iProcess Engine, the resulting Procedure is in an Unreleased state with major and minor versions set to 0. This is indicated in the Properties view:



If set to Released, the Procedure version remains 0.0. If redeployed at that point, the Released version is 0.0 and the newly deployed, Unreleased version becomes 0.1. To change the major version of a deployed procedure, use the Procedure Management capabilities of the iProcess Modeler (see *TIBCO iProcess Modeler Procedure Management*).

Right-clicking the Procedure shows the operations that you can perform when the Procedure is in an Unreleased state:



As shown, you can Release or Undeploy the Procedure.



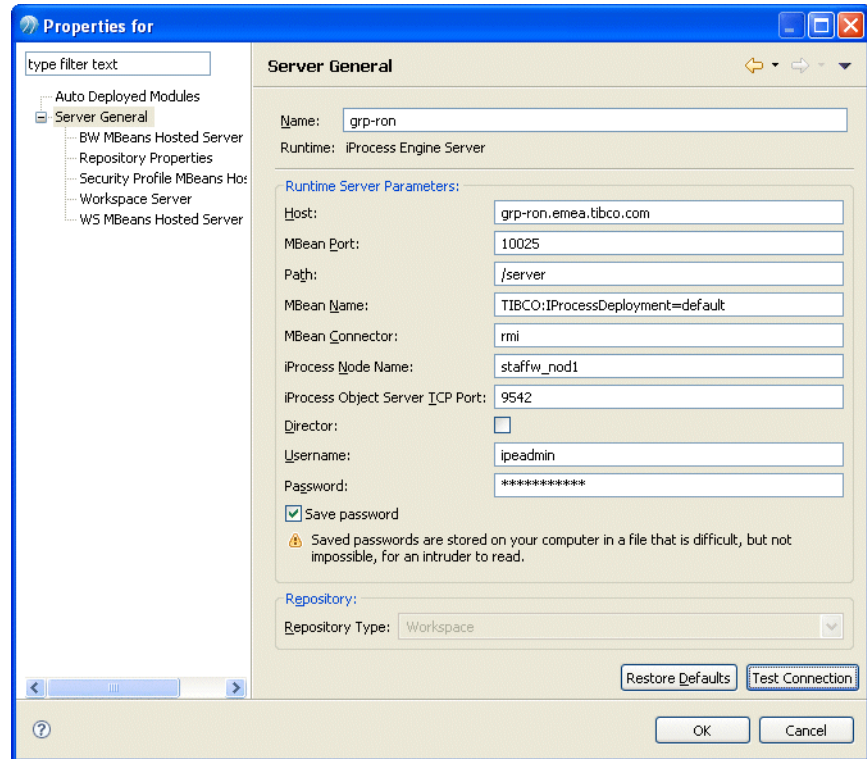
You cannot undeploy a Procedure with active Cases.

Similarly after you have released a Procedure, you can withdraw or undeploy a Procedure.

Changing Server Properties

If you need to change a Server's properties (for example, to change the password or to change the IP address of the server) do the following:

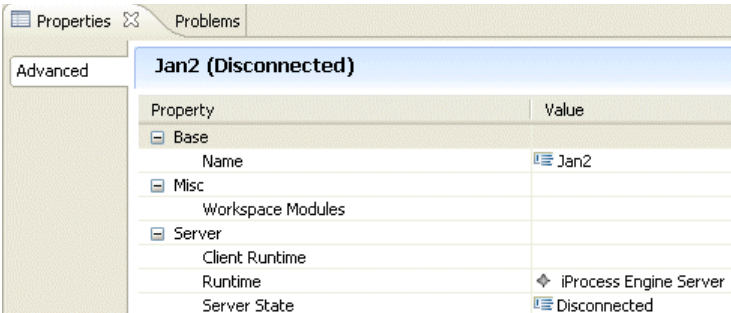
1. Select the Server, right-click and select **Properties**. A dialog similar to the following is displayed:



2. Make the necessary changes to the properties. There are also dialogs for the MBean servers and the Workspace Servers. When you make changes to the connection details, click **Test Connection** to ensure that the values you have entered are valid.
3. Click **Auto Deployed Modules** to see a list of Modules for which you specified **Deploy on save** when you created them. If there are Modules which you no longer wish to automatically deploy, click **Remove** to disable automatic deployment for the selected Module.
4. When you have finished making changes, click **OK**. The changes take effect the next time you connect to the Server. You can change the configuration while connected to the Server, and when you next Connect your changes will be applied.

Disconnecting from the Server

To disconnect from a Server, right-click the Server and select **Disconnect**. The Server State in the Properties view should change to Disconnected:



Chapter 9 **Import and Export - iProcess Modeler**

This describes how to export TIBCO Business Studio processes to the TIBCO iProcess Modeler, and how to import procedures from iProcess Modeler into TIBCO Business Studio.

- [Integrating with the TIBCO iProcess Suite on page 248](#)
- [Importing a TIBCO iProcess Modeler Package, page 250](#)

Integrating with the TIBCO iProcess Suite

The primary way of getting a TIBCO Business Studio process into iProcess is to deploy it as described in [Chapter 8 on page 219](#). However, you can also do the following:

- [Correct any Validation Errors](#)
- [Export the Process from TIBCO Business Studio](#)
- [Import the Process into TIBCO iProcess Workspace](#)

Task A Correct any Validation Errors

1. If you do not already have one, create a destination environment that includes the Modeler destination component as described in [Tutorial 1: Working with Destination Environments on page 19](#). Also add any other required destination components (for example, web services).
2. Go to the Properties view for the process click the **Destinations** tab, and select a destination environment that includes the Modeler destination component.
3. Save the Package you want to export.

The process is validated for potential import into the TIBCO iProcess Modeler and any errors are reported in the Problems view. To see this view, select **Window > Show View > Project Explorer**. For more information about correcting validation errors, see [Chapter 7 on page 213](#).

Task B Export the Process from TIBCO Business Studio

TIBCO Business Studio can export Packages/processes in XPDL format which can be read by the TIBCO iProcess Modeler.

To export a process, do the following:

1. Select the Packages you want to export.
2. Select **File > Export**. The **Export** dialog is displayed.
3. Expand **Business Process Management** and select **TIBCO iProcess Modeler XPDL** and click **Next**.
4. Change your package selection if required.

5. Select a destination for the export by either:
 - Selecting **Project** (to put the export in the **Exports\iProcess Modeler XPDL** sub-directory within your Project directory), or
 - Selecting **Path:** and clicking **Browse...** to specify a directory for the export.



TIBCO recommends that you choose the directory where your TIBCO iProcess Workspace software is located as the export destination. If you do not export into the directory that contains TIBCO iProcess Workspace and your process has BusinessWorks or web services steps, you may receive the error message `Missing URL alias manager property` when you import the XPDL into iProcess and open the EAI step.

6. Click **Finish**.

Task C Import the Process into TIBCO iProcess Workspace

Import your TIBCO Business Studio process as you would any other process (see the *TIBCO iProcess Modeler: Procedure Management Guide* for more information). See [Object Mapping on page 112](#) for information about how the TIBCO Business Studio objects are mapped into TIBCO iProcess objects.

Open the process in the TIBCO iProcess Modeler and complete its detailed implementation. Then execute and test the process.

Importing a TIBCO iProcess Modeler Package

Processes and the Packages that contain them are stored in XPDL format. This section describes how to import a TIBCO iProcess Modeler XPDL Package.

Important Notes on iProcess Import

- It is important to follow this particular sequence of importing iProcess XPDLs:
 - IO templates
 - Subprocesses (may reference IO templates and subprocesses but only those that have already been imported)
 - Main processes which reference the imported sub-processes
- The order of deployment must be the same as mentioned for imports.
- You cannot export from TIBCO Business Studio and reimport.

When importing a TIBCO iProcess Modeler XPDL Package, TIBCO Business Studio preserves as many of the implementation details from the original XPDL file as possible, however some implementation details are lost during the import.

For example:

- TIBCO iProcess scripts are supported "as is" on import and export, and when deployed.
- Stand-alone scripts are preserved on import/export and are viewable via the **Advanced Properties** tab for the process.
- TIBCO BusinessWorks steps are converted on import to service tasks of the type BW Service.
- Web Service steps are converted on import to service tasks of the type Web Service.
- When an EAI step is discovered in iProcess XPDL import that is not converted to a TIBCO Business Studio equivalent, then it will be converted to an

"iProcess EAI" implementation service task. The following EAI types will be converted to an "iProcess EAI" implementation service task:

- Custom EAI types (developed using the iProcess EAI SDK)
- EAI Java
- EAI COM
- EAI Pojo
- EAI Decisions
- EAI ORCH
- EAI ORDER
- EAI TRANSFORM

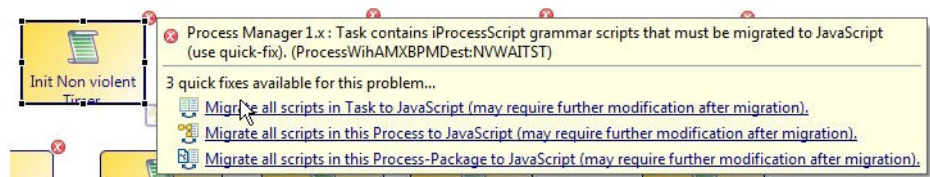
See [Service Task Properties \(EAI\)](#) on page 266.

Restrictions

The procedure described in this section is *only* for importing a TIBCO iProcess Modeler XPDL Package that was created in TIBCO iProcess Modeler. You *cannot* use the Import menu option to do the following:

- Import TIBCO Business Studio XPDL files created using previous versions of TIBCO Business Studio. To use them with the current version of TIBCO Business Studio, manually copy the Package into your workspace and migrate the processes as described in *TIBCO Business Studio Process Modeling Guide*.
- You cannot import XPDL files that were created using the TIBCO Business Studio **Export > TIBCO iProcess Modeler XPDL** option. Only iProcess Modeler XPDL files exported from TIBCO iProcess Modeler can be imported.

TIBCO iProcess grammar scripts cannot be directly edited. Instead you must first change the script grammar to JavaScript. The script text is converted to JavaScript using the helper function which performs a translation of those iProcess script constructs that can be readily translated.



TIBCO iProcess scripts are read-only, and cannot use content-assist or validation.

Procedure

1. Select **File > Import**.
2. In the **Import** dialog, select **TIBCO iProcess Modeler Analyst process** and click **Next**.
3. Enter the following:
 - a. Click **Browse** to select the **From Directory** where the XPDL file you want to import is located.
 - b. Click **Browse** to select the **Into folder** where you want the imported package to be placed.
 - c. Select **Overwrite existing resources without warning** if you want to automatically overwrite any existing Packages or processes with the same name.
 - d. Click **Finish**.
4. The Package/processes that you have imported are shown in the Project Explorer.

Chapter 10 **Reference**

This chapter describes the major parts of the TIBCO Business Studio user interface that are specific to the Solution Design capability of the Modeler Perspective. For information about parts of the TIBCO Business Studio user interface that are specific to the Business Analysis capability, see *TIBCO Business Studio Process Modeling User's Guide*.

Topics

- [Overview](#), page 254
- [Properties View](#), page 255
- [Process Properties \(Advanced Tab\)](#), page 256
- [User Task Properties](#), page 259
- [Service Task Properties \(Web Service/BusinessWorks Service\)](#), page 261
- [Service Task Properties \(EAI\)](#), page 266
- [Service Task \(Database\)](#), page 269
- [Service Task \(Java\)](#), page 271
- [Service Task \(TIBCO iProcess Conductor - Orchestrator\)](#), page 272
- [Embedded Sub-Process \(Under Transaction Control\)](#), page 277
- [Reusable Sub-Process Tasks \(Calling Sub-Processes\)](#), page 278
- [Reusable Sub-Process Tasks \(Calling Process Interfaces\)](#), page 279
- [Send and Receive Tasks](#), page 280
- [XOR Gateway \(With Outgoing Conditional Flow\)](#), page 281
- [Intermediate Events](#), page 282
- [TIBCO iProcess System Fields](#), page 283

Overview

In Eclipse, a **Perspective** includes the views and set of editors that you commonly use for a specific type of work. TIBCO has created several TIBCO Business Studio perspectives that include the views and editors you commonly use when creating business processes. A **Capability** in Eclipse is a mechanism to enable and disable specific areas of UI based on the current user's selected role.

Within the Modeling Perspective, there are two Capabilities:

- Business Analysis (see the *TIBCO Business Studio Process Modeling User's Guide*)
- Solution Design (described in this section).



You can switch between these Capabilities using the following menu.



Properties View

The **Properties View** shows you detailed information about the currently selected object and allows you to specify the characteristics of an object.

To see this view, select **Window > Show View > Properties**. To view property information, you must have an object selected.

Process Properties (Advanced Tab)

When you have selected a process in the Process Editor, the following properties are available:

Property	Description
Auto-Purge Cases on Completion	<p>Specifies whether or not cases of this process will be automatically purged when they complete.</p> <p>By default, the property is not set to false, so cases are not automatically purged. Warning: When a case is purged, all record of it is deleted, including the audit trail. This option should therefore be used with care.</p>

Case Admin Access Permissions	<p>This property allows the user to perform functions that are specific to cases of a procedure, such as viewing lists of cases, rebuilding a case, and auditing cases. For information about which users can administer a case by default, see <i>TIBCO iProcess Modeler Procedure Management</i>.</p> <p>You can specify the users who can audit cases of the procedure by specifying an user attribute expression that evaluates to a user, group, or role name. The ability to audit this procedure is limited to only the users, groups, or roles specified, or the users for which the expression(s) evaluates to true. For example:</p>
-------------------------------	---

Property	Value
<input type="checkbox"/> iProcess Process Properties	
Auto-Purge Cases On Completion	false
<input checked="" type="checkbox"/> Case Admin Access Permissions	
User Attribute Expression [1]	IPEUserUtil.GETATTRIBUTE("GRP_ProcessAdmins")== "Yes";
User Attribute Expression [2]	IPEUserUtil.GETATTRIBUTE("NAME")== "willis";

In this example, users that are part of the group **ProcessAdmin** and the user **willis** can administer cases.

For more information, see the chapter that describes TIBCO Business Studio JavaScript classes in the *TIBCO iProcess Expressions and Functions Reference Guide*.

Property	Description
Case Description On Start	<p>Specifies whether users can, cannot or must provide a case description when they start a case of a procedure. Select from the following options:</p> <ul style="list-style-type: none"> • Required The user must enter a case description in the Case Start dialog when they start a case of this procedure. • Optional The user can, but does not have to, enter a case description in the Case Start dialog when they start a case of this procedure. • Hidden The user cannot enter a case description in the Case Start dialog when they start a case of this procedure. (The Case Description field is not displayed in the Case Start dialog.) <p>You can use the Hidden option if you want to calculate the case description from within the process. For example, you might want to make the case description equal to the customer's postcode (zip code) and name to facilitate searching for the case in response to customer queries.</p> <p>By default, a case description is required.</p>

Case Start Access Permissions	<p>By default, when a process is deployed, all iProcess users on the node have the authority to start cases of that procedure. You can, however, specify an user attribute expression that evaluates to a user name or role. If a user name or role is specified, case-start access is limited to only the users, groups, or roles specified, or the users for which the expression(s) evaluate to true. If no one is designated as having access through this property, all users have access. For example:</p>
-------------------------------	---

For example:

Property	Value
iProcess Process Properties	
Auto-Purge Cases On Completion	false
Case Admin Access Permissions	
Case Description On Start	Required
Case Start Access Permissions	
User Attribute Expression [1]	IPEUserUtil.GETATTRIBUTE("GRP_ProcessStarters")=="Yes";

In this example, only users that are part of the group **ProcessStarters** can start cases.

Property	Description
Enable Case Prediction	<p>Specifies whether or not the use of case prediction is enabled for this process.</p> <p>By default, the property is set to false, so case prediction is disabled.</p> <p>For more information about case prediction, see “Using Case Prediction to Forecast Outstanding Work Items” in the <i>TIBCO iProcess Modeler - Advanced Design</i> guide.</p>
Ignore Blank Addressees	<p>Specifies whether a blank addressee for a step should be treated as an error condition.</p> <p>By default the property is not false, so a step with a blank addressee is treated as an error.</p> <p>For more information about using fields to define variable addressees (dynamic routing), see “Routing the Business Process” in the <i>TIBCO iProcess Modeler - Basic Design</i> guide.</p>
Normalise Case Data	<p>Specifies whether or not case data normalization is enabled for this process. By default this property is set to true.</p> <p>Case data normalization makes case data searching more efficient and therefore faster.</p> <p>For more information about case data normalization, see "Administering Case Data Normalization" in the <i>TIBCO iProcess Engine Administrator's</i> guide.</p>
Use Working Days	<p>Specifies whether to use the iProcess Engine’s working week or a full 7-day week, when performing date calculations such as checking for deadline expiry.</p> <p>By default, the property is set so the iProcess Engine’s working week is used.</p> <p>For example, if a step is sent to a queue on a Wednesday and has a deadline that is due to expire in three days, the deadline will expire:</p> <ul style="list-style-type: none">• on the following Monday, if the Use Working Days property is selected.• on the Saturday, if the Use Working Days property is not selected.

User Task Properties

When you have selected a user task in the Process Editor, the following properties are available:

Tab	Property	Description
General	Label	The label displayed on the process diagram for the task.
	Name	Task name used by iProcess. This is used for script references and sub-process references to process data.
	<ul style="list-style-type: none"> No Form URL User Defined URL Form 	<p>Configure the user task as follows:</p> <ul style="list-style-type: none"> No Form URL Select this option if you have not created a specific form for the user task. Upon export to iProcess a standard iProcess form is created. User Defined URL Select this option if you want to point to a specific URL (for example, if you authored a form outside of TIBCO Business Studio). Manually enter the URL. TIBCO Business Studio cannot validate the URL, so ensure it is correct. Form Use this option if you have created a form using TIBCO Business Studio. Browse to select a form from Forms special folder. Upon deployment/import to the iProcess Engine, the Task becomes a step with a Form type of Formflow Form.
Interface	Visibility	Allows you to set whether the Task is private or public. Public Tasks publish information (such as required parameters) to an external process or application. Private Tasks do not publish any information.

Tab	Property	Description
	Data	<p>Allows you to select the data fields or parameters that the Task requires, their mode (In, Out, or In/Out), and whether they are mandatory.</p> <p>By default all process data is available to a Task. When you explicitly associate process data with a Task, only that process data is associated with the Task.</p>
Scripts	Open Script	Allows you to specify a Script that when exported to the iProcess Modeler will create a script object. The script object will be called by the Initial Form Command when the work item form is opened from the user's Work Queue.
	Close Script	Allows you to specify a Script that when exported to the iProcess Modeler will create a script object. The script object will be called by the Keep Form Command when the form is returned to the user's Work Queue.
	Submit Script	Allows you to specify a Script that when exported to the iProcess Modeler will create a script object. The script object will be called by the Release Form Command when the form is released.
	Audit Initiated Script	Defines the expression for the value of an EAI Step Call-Out Initiated message in a case audit trail.
	Audit Completed Script	Defines the expression for the value of an EAI Step Call-Out Completed message in a case audit trail.
	Audit Timeout Script	Not supported for iProcess destinations.
	Audit Cancel Script	Not supported for iProcess destinations.
Advanced		See Priority and Step Permissions on page 94 .



Service Task Properties (Web Service/BusinessWorks Service)

When you have selected a Service Task in the Process Editor with either **Web Service** or **BW Service** as the **Service Type**, the following properties are available:

Tab	Property	Description
General	Service Type	<p>When you click the Select button and choose a WSDL operation, the Port Type, Operation Name, and so on are populated based on the selected WSDL operation.</p> <p>Before you can select an operation, you must have either imported a WSDL file to a Service Descriptors folder, or have generated a WSDL file (by clicking the Generate button). See Importing a WSDL File/Service Description on page 139 and Generating a New WSDL File on page 132).</p> <p>Note: Except for a WSDL file obtained from BusinessWorks, the WSDL file you select must use either the Document Literal or RPC Encoded call style. It must also use the Synchronous Request/Response Message Exchange Pattern.</p>
	Port Type	See Service Name description.
	Operation Name	Click the Select button to choose a WSDL operation from a WSDL file that you have added to the Project (see Working with WSDL Files on page 131).
	Port Name	Populated automatically when the WSDL operation is selected.
	Service Name	Populated automatically when the WSDL operation is selected.
	Transport	Either SOAP over HTTP or XML/JMS. Populated automatically when the operation is selected.
	Use local WSDL	Select this option if you have added the WSDL file to the project and want the WSDL file to be packaged with the Project during the Packaging phase.
	Use remote WSDL (Web Services)	Select this option if the WSDL file is located remotely. You must specify either a URL or a System participant as the Endpoint Name .

Tab	Property	Description
	Use LiveLink (BusinessWorks)	Select this option if the WSDL file will be obtained using a BusinessWorks live link in the runtime environment. You must specify a System participant as the Endpoint Name .
	Security Profile	Select this option to specify a logical name (system participant) for a web services security profile. At deployment, this must be mapped to an actual security profile in iProcess.
Interface (<i>all task types except Reference</i>)	Visibility	Specify the visibility of an Event or Task (whether it is private or public). Public Events or Tasks publish information (such as required parameters) to an external process or application. Private Events or Tasks to publish any information to external applications.
	Data	<p>Allows you to select the data fields or parameters that the Task requires, their mode (In, Out, or In/Out), and whether they are mandatory.</p> <p>By default all process data is available to a Task. When you explicitly associate process data with an Task, only that process data is associated with the Task, and therefore available on the Input to Service and Output From Service tabs.</p>
Input To Service		Use this section to create a mapping from an actual parameter (data field or parameter) into the formal parameters of the selected service. Create a mapping by clicking a parameter, dragging to the destination parameter, then releasing the mouse button. You can also apply XPath or XSLT scripts to a mapping by selecting the desired script grammar, and entering the script in the Script Editor (see Applying Scripts to a Mapping on page 150 . Note: For deployment to the iProcess Engine, you must specify at least one input parameter mapping.

Tab	Property	Description
Output From Service		Use this section to create a mapping from a formal parameter of a service to an actual parameter (data field or parameter). Create a mapping by clicking a parameter, dragging to the destination parameter, then releasing the mouse button. You can also apply XPath or XSLT scripts to a mapping by selecting the desired script grammar, and entering the script in the Script Editor (see Applying Scripts to a Mapping on page 150 . Note: For deployment to the iProcess Engine, you must specify at least one output parameter mapping, however you cannot specify more than one output parameter mapping.
Advanced	Immediate Release	(for web service steps, corresponds to Automatic Delayed Release in iProcess) After the called service completes, flow continues.
	Delayed Release	(for web service steps, corresponds to Manual Delayed Release in iProcess) After the service completes, the Service Task remains outstanding until a signal is received from an external source.
	Immediate Release (Asynch With Reply)	(for web service steps, corresponds to Asynchronous with Reply in iProcess, not applicable to BusinessWorks steps) Select this option so that the request and response calls are de-coupled into separate requests from the iProcess Background process. If a procedure is designed with multiple, parallel steps, they can be invoked asynchronously and allowed to run simultaneously but the Background process will be blocked until all of the responses are received. This means that no other processing can occur by that Background process. However if the response time is too slow, then the iProcess Web Services Plug-in will automatically switch to Delayed Release. For more information about invocation styles and transaction scope, refer to <i>TIBCO iProcess Modeler - Integration Techniques</i> .

Tab	Property	Description
	Return Configuration (BusinessWorks only)	<p>Allows you to use data fields and values to specify the behavior for this step in the case of a timeout period being reached.</p> <ul style="list-style-type: none">• To specify the Deadlock Timeout Field, click  to select a data field or formal parameter that should be populated if a failed JMS message could not be re-consumed, but the deadlock timeout has reported success. You can only select Fields or parameters of type Text, Integer, or Decimal Number. In the Deadlock Timeout Value field, specify the value to which the selected field should be set.• To specify the Response Timeout Field, click  to select a data field or formal parameter that should be populated when a response is received. You can only select Fields or parameters of type Text, Integer, or Decimal Number. In the Response Timeout Value field, specify the value to which the selected field should be set.

Tab	Property	Description
	Timeout Configuration (<i>BusinessWorks only</i>)	<ul style="list-style-type: none"> • In the Consume Timeout field, enter the length of time (in milliseconds) that the iProcess Engine should wait for the re-consumption of a JMS message to succeed. If the message is re-consumed, the iProcess Engine generates an error and re-tries the step. • In the Deadlock Timeout field, enter the length of time (in milliseconds) that the iProcess Engine should wait if the attempt to re-consume the message fails. If a response is received before the end of this period, processing continues as normal. If no response has been received by the end of the period, it aborts the step and sends an error message to the iProcess Engine. The background process re-tries the step. • In the Expiration Timeout field, enter the length of time (in milliseconds) that the JMS server should wait before deleting the JMS message, in the event the message fails to be consumed. You are recommended to set this to at least one second (that is, a value of 1000 milliseconds) more than the Response Timeout value. • In the Response Timeout field, enter the length of time (in milliseconds) that the iProcess Engine should wait for a JMS message to be consumed before deleting it. <p>Note that specifying 0 for any of the timeout values means an indefinite wait (effectively no timeout), not an immediate timeout.</p>

Service Task Properties (EAI)

When you have imported EAI steps that are not natively handled as TIBCO Business Studio service task implementation types, a Service Task in the Process Editor is set with **iProcessEAI** as the **Service Type**.


The following properties are available:

Tab	Property	Description
General	Service Type	This is set to iProcessEAI .
	EAI Type	Read-only. See note under Permit Edit below.
	EAI Run Type	Read-only. See note under Permit Edit below.
	EAI Definition	Multi-line description of the raw-content of the EAI run-step. Read-only. See note under Permit Edit below.
Permit Edit		<p>Editing of pre-defined EAI definition data is not something that should be done lightly by the user. Some have very complex formats that are easily broken and can therefore cause runtime issues.</p> <p>This checkbox is always unchecked when an iProcess EAI task is selected.</p> <p>No changes are permitted until this checkbox is checked.</p>
Advanced	iProcess EAI Task	<p>Select one of the following:</p> <ul style="list-style-type: none">• Immediate Release Equivalent to iProcess Delayed Release - Never. The step is never set to delayed release so it will be released immediately.• Delayed Release Equivalent to iProcess Delayed Release - Always. The step will be released by the external application.• Delayed Release Conditional The condition is evaluated to determine if the step will be delayed release. If you select this option, press the Return key, and then enter a valid condition expression in the Delayed Release Condition property.

Service Task Properties (Email)

When you have selected a Service Task in the Process Editor with **Email** as the **Service Type**, the following properties are available:





Unless otherwise stated, all of the properties listed in the following table can be specified using a data field or parameter. Click the  button to select the data field or parameter.

Tab	Property	Description
General	To	Specify the recipient of the email.
	Subject	Specify a subject line for the message.
	Body	Specify the body text that comprises the message.
E-mail > Definition	From	Select either Use Server Configuration to use the address of the server from which the email is sent or select Use Custom Configuration to select different configuration details.
	To	Specify the recipient of the email.
	Cc	Specify the recipients to whom you want to send a copy of the email. Their email address is visible to other recipients of the email.
	Bcc	Specify the recipients to whom you want to send a copy of the email. Their email address is not visible to other recipients of the email.
	Reply To	Use this parameter to specify a different email address to which recipients of a message can reply.
	Headers	Use this parameter to specify additional information in the header of the email.
	Priority	Select a priority from the drop-down list (Normal , High or Low).
	Subject	Specify a subject line for the message.

Tab	Property	Description
E-mail > Body		Enter the body text of the message. Click src and preview to alternate between viewing the source of the body text and previewing how it will look to the recipient
E-mail > Attachments	Field Contents	Specify a document to be attached to the message: <ul style="list-style-type: none"> • Field Contents: - use this option to attach the contents of a parameter or data field to the email message. • Files: use this option to browse the file system and attach a file to the email message.
E-mail > SMTP Configuration	Use Server SMTP Configuration	Select User Server SMTP Configuration to use the settings of the server where the process will run.
	User Custom SMTP Configuration	Select Use Custom SMTP Configuration settings to specify a different Host and Port from the server where the process will run.
E-mail > Error Handling	Return Status Fields	Use this Property to provide for basic error handling (for example, if a message cannot be sent) <ul style="list-style-type: none"> • Code: - select either a parameter or data field that represents an error code. • Message: - select either a parameter or data field that contains the message you want displayed when the email cannot be sent.

Service Task (Database)

When you have selected a Service Task in the Process Editor with **Database** as the **Service Type**, the following properties are available:

Tab	Property	Description
General	Operation	Stored Procedure is the only option valid for iProcess.
	Name	Displays the name of the selected stored procedure in the following format: [owner.]stored_procedure_name Click  to display dialogs that allow you to select the database and a stored procedure to run.
	SQL	Displays the SQL query to run on the database. This option is not supported for iProcess destinations. Click  to display dialogs that allow you to select the desired database and use the query builder to create a query.
Database	Operation	See previous description on the General tab.
	Name/SQL	See previous description on the General tab.
	Parameters	Allows you to define the parameter mappings between parameters in the stored procedure and Case Fields in the TIBCO iProcess Engine. These Case Fields are represented in TIBCO Business Studio as data fields.

Tab	Property	Description
Advanced	Invocation Style	<p>Select one of the following:</p> <ul style="list-style-type: none">• Immediate Release Equivalent to iProcess Delayed Release - Never. The step is never set to delayed release so it will be released immediately.• Delayed Release Equivalent to iProcess Delayed Release - Always. The step will be released by the external application.• Delayed Release Conditional The condition is evaluated to determine if the step will be delayed release. If you select this option, press the Return key, and then enter a valid condition expression in the Delayed Release Condition property.
	Database Name and Server Name	

Service Task (Java)

When you have selected a Service Task in the Process Editor with **Java** as the **Service Type**, the following properties are available:



Tab	Property	Description
General	Project	Specifies the Project that contains the Java code you want to call. You can automatically populate this field by clicking Select Class and selecting the desired class.
	Class	Displays the Class of which you want to create an instance (selected by clicking Select Class or Select Factory).
	Method	After you have selected a class, this drop-down list is populated with the available methods. Select a Method from the drop-down list.
	Class (factory)	As an alternative to clicking Select Class , you can click Select Factory to choose a factory class to create an instance of a Class.
	Method (factory)	Method factory used to create an instance of a Class. This is populated when the Factory is selected.
Input To Service		Use this section to create a mapping from an actual parameter (data field or parameter) into the parameters of the selected Method. Create a mapping by clicking a parameter, dragging to the destination parameter, then releasing the mouse button.
Output From Service		Use this section to create a mapping from a parameter of a Method to an actual parameter (data field or parameter). Create a mapping by clicking a parameter, dragging to the destination parameter, then releasing the mouse button.

Service Task (TIBCO iProcess Conductor - Orchestrator)

When you have selected a Service Task in the Process Editor with **iPC Orchestrator** as the **Service Type**, the following properties are available. For more information about configuring orchestrator tasks, see [Orchestrator Service Tasks on page 181](#).


Tab	Property	Message Type	Description
General	Message Type	Activate Sink	This defines a milestone of the plan and instructs the TIBCO iProcess Conductor to wait for the completion of ALL activities upon which this milestone depends before releasing this step. Use the Interface tab to specify the process data that is passed between the TIBCO iProcess Conductor and the TIBCO iProcess Engine.

Tab	Property	Message Type	Description
		Start Plan	<p>This instructs the TIBCO iProcess Conductor that the execution plan in the parent fulfillment process should be started. You must enter the following fields to provide the necessary information that TIBCO iProcess Conductor requires:</p> <ul style="list-style-type: none"> • Graft Step Name Enter the name of the graft step defined in the fulfillment process that instructs the TIBCO iProcess Conductor which graft step to use to orchestrate the required execution plan. Content assist is provided; press Ctrl+Space to choose from a list of defined graft steps. For more information about defining graft steps, see Graft Steps on page 106. • Execution Plan Id Override Field Select the data field or parameter from your fulfillment process that contains the unique identifier for the execution plan or execution plan template. This ID is required because the Plan ID which is generated when the execution plan is instantiated is not available at the start point, so the Execution Plan ID is used instead. This ID is used to identify the execution plan template that this execution plan should instantiate. • Callback Task Press Ctrl+Space to select from a list of receive tasks. The receive task is used to pass the execution plan ID back to the fulfillment process. This enables the execution plan ID to be passed back to the fulfillment process. See <i>TIBCO iProcess Conductor Implementation</i> for more information about the ad-hoc orchestration event step.
		Task Complete	<p>This instructs the TIBCO iProcess Conductor that the process component has executed and is complete (the case continues to execute). Use the Interface tab to specify the process data that is passed from the TIBCO iProcess Conductor to the TIBCO iProcess Engine. You can only specify parameters with a Mode of Out.</p>

Tab	Property	Message Type	Description
		Execution Plan Classification Values	<p>Specify classification values for the execution plan:</p> <ul style="list-style-type: none">• To create groups and values in TIBCO Business Studio, select the Designtime option. Click  to add classification values, and click into the table cells to change the Group Name and Value Code.• To specify a data field or parameter to represent the classification value in the runtime environment, select the Runtime option and click  to select a data field or parameter.

Service Task (TIBCO iProcess Conductor - Order)

When you have selected a Service Task in the Process Editor with **iPC Order** as the **Service Type**, the following properties are available. For more information about configuring orchestrator tasks, see [Order Service Tasks on page 182](#).

Tab	Property	Description
General	Order Identification	Specify data fields or parameters for the Order Field Primary Key Field and Order Amendment Index Field . The easiest way to do this is using the quick fix from the Problems view (see Order Service Tasks on page 182).
	Order State Change	<p>Specify the details of the change in status of the order:</p> <ul style="list-style-type: none"> • Transition Select the desired transition for the task: <ul style="list-style-type: none"> — Execution Sets the status of the order from Feasibility to Execution. — Feasibility Sets the status of the order from Draft to Feasibility. — Feasibility Failed Sets the status of the order from Feasibility to Feasibility Failed. • Plan Development Where the execution plan is being created for the order: <ul style="list-style-type: none"> — Execution The state that normally follows plan development where the execution plan is actually being executed. • Ad-Hoc Event Step Press Ctrl+Space to select from a list of receive tasks. The receive task is used for iProcess Conductor Order Management to notify of order amendments. • Event Case Specify the data fields from which the order step should get the process name and case number. <ul style="list-style-type: none"> — Select This Procedure to use the default iProcess fields SW_CASENUM and SW_PRONAME. — Select Specify Fields to specify different data fields for the case number and process name. Then, click  to select the data field you have defined for the case number and process name, respectively.

Tab	Property	Description
	Order Line State Change	<p>In the Order Line State Change field, you must specify:</p> <ul style="list-style-type: none">• A data field for Line Number Selection that specifies the location in the XML code where you want the status to change. The value that you specify is then inserted into an XPath expression to locate the value in the XML that you want to change.• The Order Line Status that you want the order step to set. You can select either Cancelled to indicate that the cancel operation performed on the order line is complete, or Complete to indicate that the order line has orchestrated and has completed.
	Exception Handling	<p>In the Exception Handling section, you can either:</p> <ul style="list-style-type: none">• Select the Fail Step on XML Exception checkbox. This box is selected by default. Leaving this selected means that the order service task causes the process to stop and logs the messages to the eaajava log files in the \SWDIR\logs\ directory, or• De-select the Fail Step on XML Exception checkbox. In the Exception Field Name field, specify the data field that you want to populate with the message from the non-fatal exception.
	Input to Order	Map the process data to parameters in the order.
	Output From Order	Map the data from the order to process fields.

Embedded Sub-Process (Under Transaction Control)

When you have selected an Embedded Sub-Process with the **Is A Transaction** checkbox selected, the following iProcess-specific properties are available on the **Advanced** tab:

Property	Description
Defer Start Of New Transaction For Post-Embedded Sub-Process Tasks	<p>If the start of the new transaction is deferred (the setting is true), the transaction created by the Embedded Sub-Process is committed. Then, a message is placed in the Mbox queue instructing the background to continue processing from the first post-Embedded Sub-Process Task. However, processing of this Task <i>deferred</i> because other messages in the Mbox queue are processed before returning to the post-Embedded Sub-Process Task.</p> <p>If the start of the new transaction is not deferred (the setting is false), the transaction created by the Embedded Sub-Process is committed, and processing of the first post-Embedded Sub-Process Task continues immediately. In this case you can also specify a time (in minutes) for which the background should wait before attempting to retry the transaction created by the Embedded Sub-Process. This allows you to control the Mbox retry mechanism directly from TIBCO Business Studio.</p>
Non-Deferred Transaction Failure Retry Delay (Minutes)	<p>If the start of the new transaction is not deferred (see the description of the previous property), specify a time (in minutes) for which the background should wait before attempting to retry the transaction created by the Embedded Sub-Process.</p>

Reusable Sub-Process Tasks (Calling Sub-Processes)

When you have selected a reusable sub-process Task that calls a Sub-Process in the Process Editor, the following iProcess-specific properties are available on the Advanced tab:

Property	Description
Prediction > Use Sub-Process Task Duration for Prediction	<p>If you enable case prediction for a sub-procedure, all the steps in the sub-procedure are displayed in the outstanding step list.</p> <p>If you do not enable case prediction for a sub-procedure, iProcess calculates the total duration of all the steps in the sub-procedure. Therefore, although the sub-procedure is displayed in the outstanding step list, the steps in the sub-procedure are not displayed.</p>
Start at Step	<p>This feature is provided so that an external process (or application) can limit itself to perform case start at step / trigger events to only those defined as Public Steps or Events. The public steps/events are not enforced by iProcess/TIBCO iProcess Objects. (This also affects sub-procedures. The list of steps displayed in the Select Step to Start Sub-procedure At: can be restricted to those steps defined as Public steps.</p>

Reusable Sub-Process Tasks (Calling Process Interfaces)

When you have selected an reusable sub-process Task that calls a process interface in the Process Editor, the following properties are available:

Property	Description
Is Graft Step	Set this property to true if you want the task to be considered a graft step.
Return Status Array Field	Select an array field that will be used to return an error return value. Refer to “Returning an Error Status” in the <i>TIBCO iProcess Modeler - Advanced Design</i> guide for more information about using arrays for this purpose.
Runtime Sub-Process Validation	<p>The settings in this category determine what happens as a result of validation in the runtime environment:</p> <ul style="list-style-type: none"> • Halt Process on Invalid Interface Sub-procedures that do not use the same process interface (in the iProcess, I/O parameter template). Select this option to stop the process if iProcess finds parameters that are not in the I/O parameter template being used by the dynamic sub-procedure call. • Halt Process on Invalid Interface Version Sub-procedures that use different versions of the same process interface (I/O parameter template). Select this option to stop the process if iProcess finds some parameters that are not valid for the version of the template being used for this call. • Sub-procedure names are invalid Select this option to stop the process if iProcess cannot find one of the sub-procedures it needs to call.
Start Step Array Field	Specify the name of a text array data field that contains Task Names where each Sub-Process should be started.

Send and Receive Tasks

When you have selected a Send or Receive Task in the Process Editor, the following properties are available:

Tab	Property	Description
General	Service Type	<p>By default this is Unspecified, but optionally, you can specify for documentation purposes that a Send or Receive Task send a message or receive a message using a web service. You select a web service operation for a Send or Receive Task in the same way you would for a Service Task (Web Service) - see Service Task Properties (Web Service/BusinessWorks Service) on page 261. However, note the following:</p> <ul style="list-style-type: none">• Send Tasks cannot be deployed.• If a Receive Task specifies a web service implementation, it cannot be deployed or exported.

XOR Gateway (With Outgoing Conditional Flow)

When iProcess predicts the duration of a procedure, it needs to know what path to follow through the procedure. In a TIBCO Business Studio process, you can define an expected path the case will follow on an XOR gateway with an outgoing conditional Sequence Flow. This is defined on the **Advanced** tab, with the **Condition Prediction Configuration** property. Select one of the following values:

- **Evaluate Expression** Case prediction will always evaluate the condition.
- **Always True** Case prediction will always default the condition to true. If you have a condition that usually evaluates as true, you should select the **Always True** checkbox.
- **Always False** Case prediction will always default the condition to false. If you have a condition that usually evaluates as false, you should select the **Always False** checkbox.

Intermediate Events

When you have selected an Intermediate event (of type Timer or None) in the Process Editor, the following properties are available:



The exact properties displayed depend on whether the event is located in the Sequence Flow, or on a Task boundary

Property	Description
Conditional Deadline	Specify the conditional deadline for Condition 1 (OR) and Condition 2 (OR) . If either condition is true, the deadline is set. Also, if neither conditions are present, the deadline is not set.
Don't Include In Future Work Items List	If set to false , the event is included in prediction. If set to true , the event is excluded from prediction.
Use Deadline for Task Duration	If set to true , if a deadline is set then the event duration will be the same as the deadline. This means you do not have to manually set the duration to be the same as the deadline.

TIBCO iProcess System Fields

The list below shows the name of the iProcess system field, whether the field is read only or read-write, the data type, and length.



For more information on TIBCO iProcess system fields, see the TIBCO iProcess documentation.

System Field Name	Read only or read/write	Data type	Length
SW_CASEDESC	read/write	Text	24
SW_CASENUM	read only	Integer Number	15
SW_CASEREF	read only	Text	20
SW_CP_INCPERIOD	read/write	Integer Number	4
SW_CP_INCREMENT	read/write	Integer Number	4
SW_CP_NUMINC	read/write	Integer Number	3
SW_CP_PERIODTYP	read/write	Text	1
SW_CP_VALUE	read/write	Integer Number	3
SW_DATE	read only	Date	n/a
SW_GEN_IDX	read/write	Integer Number	6
SW_HOSTNAME	read only	Text	24
SW_IP_INCPERIOD	read/write	Integer Number	4
SW_IP_INCREMENT	read/write	Integer Number	4
SW_IP_NUMINC	read/write	Integer Number	3
SW_IP_PERIODTYP	read/write	Integer Number	1
SW_IP_VALUE	read/write	Integer Number	3
SW_PRODESC	read only	Text	24
SW_PRONAME	read only	Text	8

System Field Name	Read only or read/write	Data type	Length
SW_QRETRYCOUNT	read only	Integer Number	15
SW_STEPDESC	read only	Text	24
SW_STEPNAME	read only	Text	8
SW_TIME	read only	Time	n/a
SW_MAINCASE	read only	Integer Number	10
SW_MAINHOST	read only	Text	24
SW_MAINPROC	read only	Text	8
SW_PARENTCASE	read only	Integer Number	10
SW_PARENTHOST	read only	Text	24
SW_PARENTPROC	read only	Text	8
SW_PARENTREF	read only	Text	64
SW_NODENAME	read only	Text	24

Index

A

abstract WSDL file [79](#)
 access control [112](#)
 audit trail
 customizing [87](#)

B

Business Process Management (BPM) [7](#)
 BusinessWorks
 task [145](#)

C

changes from the previous release [x](#)
 concrete WSDL file [79](#)
 Conditional Flow script [204](#)
 customer support [xvi](#)

D

database [13](#)
 database call [160](#)
 deadline expressions [97](#)
 delayed release [87](#)
 deployment [235](#)
 managing Modules [244](#)
 tutorial [54](#)
 dynamic sub-procedures [103](#)

E

E-Mail Task [36](#)

G

Gateways [116](#)
 graft steps [106](#)
 groups and roles [110](#)

I

importing a Package [250](#)
 iProcess functions [42](#)
 iProcess script file functions [194](#)

J

Java code [14](#)
 calling [172](#), [172](#)
 Java deployment [175](#)
 JavaScript preferences [207](#)

M

mapper [149](#)
 mapping date and time parameters [152](#)
 memo field [113](#)
 Model Driven Architecture (MDA) [3](#)
 Module deployment [235](#)

P

- Package import [250](#)
- Parameter
 - adding [120](#)
 - mapping [174](#)
- Plain Old Java Object (POJO)
 - configuring [177](#)
- Process
 - correcting errors [217](#)
 - testing [8](#)
- Properties View [255](#)
- public events [111](#)
- public steps [111](#)

Q

- Quick fixes [217](#)

R

- Receive Tasks [280](#)
- registry search
 - creating [128](#)
- return codes [170](#)

S

- sample Processes [9](#)
- script [93](#)
 - associating with Conditional Flow [204](#)
 - in parameter mapping [150](#)
 - on Conditional Flow [204](#)
- Script Task [14](#)
 - creating [186](#)
- Send Task [280](#)

server

- connecting to [232](#)
- creating [223](#)
- disconnecting [246](#)
- properties [245](#)

Service Oriented Architecture (SOA) [7](#)

Service Task [10](#)

- BusinessWorks [261, 266](#)
- Database [269](#)
- Email [267](#)
- Java [271](#)
- tutorial [31](#)
- Web Service [261, 266](#)

support, contacting [xvi](#)

T

- technical support [xvi](#)

TIBCO Business Studio

- package import [250](#)
- sample Processes [9](#)
- tutorials [31](#)

TIBCO iProcess Modeler

- package import [250](#)

TIBCO_HOME [xiii](#)

Timer Event scripts [205](#)

Transaction Control Steps [112](#)

U

Uniform Description, Discovery, and Integration
(UDDI) Registry

- adding [128](#)
- searching [128](#)

User Task [9](#)

- properties [259](#)
- tutorial [24, 39](#)

V

- validation error [217](#)
- validation errors
 - turning off [218](#)

W

- Web Service Definition Language (WSDL) file
 - [139](#)
 - adding [131](#)
 - copying [134](#)
 - importing [139](#)
 - Task [145](#)
 - validating [144](#)
- web services
 - calling [31](#), [31](#)
- web services task [145](#)
- Withdraw [98](#)
- withdraw links [98](#)
- WSDL file [79](#)