TIBCO Business Studio™

Concepts

Software Release 3.6.0
September 2013

Two-Second Advantage®

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 Enterprise Message Service, and TIBCO ActiveMatrix are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

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.

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 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 © 2004-2013 TIBCO Software Inc. ALL RIGHTS RESERVED.

TIBCO Software Inc. Confidential Information

Contents

Preface

This guide provides an overview of TIBCO Business Studio, its components, the concepts that underpin it and how you can use it. No previous software experience is necessary, but a familiarity with Business Process Modeling Notation (BPMN) is useful (see http://www.bpmn.org).

This guide does not cover implementation, export, or deployment. For more information about how to develop processes for a specific runtime environment, see the appropriate implementation guide.

Topics

Related Documentation

This section lists documentation resources you may find useful.

Third-Party Documentation

The Eclipse help contains useful information on the Workbench and the Eclipse user interface.

TIBCO Business Studio supports the following standards:

- Business Process Modeling Notation (BPMN)

  http://www.bpmn.org

- XML Process Definition Language (XPDL)

  http://www.wfmc.org/standards/xpdl.htm

Typographical Conventions

The following typographical conventions are used in this manual.

Table 1 General Typographical Conventions

Convention	Use
TIBCO_HOME *STUDIO_HOME*	Many TIBCO products must be installed within the same home directory. This directory is referenced in documentation as *TIBCO_HOME*. The default value of *TIBCO_HOME* 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 <*TIBCO_HOME*>. This directory is referenced in documentation as *STUDIO_HOME*. The default value of *STUDIO_HOME* 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: • 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]
italic font	Italic font is used in the following ways: • To indicate a document title. For example: See *TIBCO ActiveMatrix BusinessWorks Concepts*. • To introduce new terms For example: A portal page may contain several portlets. *Portlets* 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 *PathName*

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.

Connecting with TIBCO Resources

How to Join TIBCOmmunity

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

How to Access TIBCO Documentation

You can access TIBCO documentation here:

http://docs.tibco.com

How to Contact TIBCO Support

For comments or problems with this manual or the software it addresses, 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:

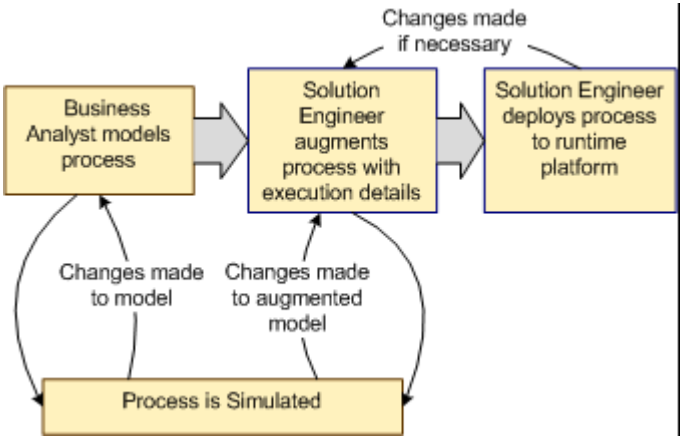
  https://support.tibco.com

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

Introduction

TIBCO Business Studio is for business analysts and solution designers, including those responsible for the analysis, design, simulation, implementation, and deployment of business processes.

The following diagram shows how TIBCO Business Studio is intended to be used:



Topics

Welcome Page

If you are new to using this product, there are several resources available to you from the Welcome page that is displayed when you first start TIBCO Business Studio. The Welcome page is also available from the **Help** menu.

These resources include:

- Overview of the features available to you.
- Sample processes that you can install into your workspace.
- Tutorials that explain how to create a simple process. These tutorials will help you become familiar with the Process Editor.
- Links to the TIBCO Business Studio discussion forum and to the TIBCO Business Studio Developer Network.

Process Modeling Methodology

Modeling a process can be achieved in several different ways, however the general approach described here reflects best practices.

Task A Define the "As is" Process

1. Interview business end users about their current practices.
2. Capture the process flow (either on paper first or directly in modeling tool).
3. Capture the process relevant data (either on paper first or directly in modeling tool).

Set the appropriate *destination environment* on the process (this specifies the runtime environment where the process will be executed). This can be done either when first creating the process, or at any time prior to implementation. Setting the destination environment when the process is first created avoids error messages and warnings associated with modeling constructs that cannot be executed in the runtime environment.

4. Attach the process-relevant data at key points (for example, at decision points or certain activities).
5. Simulate and analyze to ensure that the "As is" process is an accurate representation of the current process.

Task B Define the "To be" Process (Optional)

This is an iterative exercise in which you:

1. Propose optimizations (process changes and new automation of existing processes).
2. Simulate to validate changes or to quantify estimated savings.

Task C Define the Business Object Model and Organization Model (Optional)

Define a *business object model* that defines key business terms specific to your corporate environment (for example, in an insurance environment, a claim, claimant, and so on). This can be used as an analysis tool.

Task D Hand Over for Implementation (Optional)

If you have been using Studio for Analysts to create the process, you must switch to Studio for Designers to complete the implementation.

If the process is to be executed in a BPM environment, you should do the following:

1. If it has not already been set, set the appropriate *destination environment* (this specifies the target environment where the process will be executed).
2. Check the Problems view for any warnings or errors in the process.
3. Hand the process off to the solution engineer for implementation. The solution engineer will underpin the process with the necessary details (such as calls to web services and so on) that will enable the process to execute in the specified destination environment.

Task E Deployment

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.

For more information about deployment, see the destination-specific implementation guide.

Capabilities and Perspectives

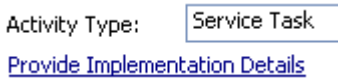
A *capability* in Eclipse is a mechanism to enable and disable specific areas of the user interface (UI).

A *perspective* in Eclipse 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. This document describes the features of the Modeling perspective.

Studio for Analysts provides controls in a simplified user interface (ribbon format). You cannot switch between perspectives or capabilities when using Studio for Analysts.

When you install the Solution Design features (the default installation), you can also see parts of the UI relevant to process implementation by clicking the provided links in the Properties view. For example:



Source Control

One problem that occurs when dealing with processes across their normal life cycle (from creation, testing, rollout to maintenance), is how to know that a given process created by the analyst, elaborated by the solution engineer, and signed off by the process owner, is *exactly* the one that is in use in a specific environment (for example, a development, user-acceptance or production environment).

Particularly in large and complex projects where data is shared or modified by several people, a source control system becomes necessary. Most enterprises have one or more products for Source Configuration Management (SCM). This may be a commercial product such as Perforce, Rational's Clearcase or an open source solution such as:

- Concurrent Versions System (CVS) (http://www.nongnu.org/cvs/) or
- Subversion (http://subclipse.tigris.org/).

TIBCO does not provide its own SCM product, preferring to integrate with the enterprise's choice for SCM. The Eclipse feature for integrating with such an SCM product is known as the Team Synchronization. Since Eclipse provides CVS by default, the following section describes how to use Subversion; you should contact your SCM vendor for commercial plug-ins.

SCM is one part of Application Lifecycle Management (ALM) dealing only with the preservation of revisions of software at different times, not the editorial and approval processes that drive those different revisions.

Standards Support

TIBCO Business Studio supports the industry standards Business Process Modeling Notation (BPMN) and XML Process Definition Language (XPDL).

Business Process Modeling Notation (BPMN)

BPMN is a graphical notation, developed by the Business Process Management Initiative (BPMI) and now part of the Object Management Group (OMG), for representing the steps and flow of business procedures. The TIBCO Business Studio Process Editor supports BPMN 1.2.

For more information, see http://www.bpmn.org.

XML Process Definition Language (XPDL)

XPDL is used to represent the underlying structure of a business process to TIBCO Business Studio. Packages are stored in XPDL 2.1 format. Normally, you do not use XPDL directly, but indirectly by creating a process package then editing a process within it using the Process Editor.

For more information, see http://www.wfmc.org/xpdl.html.

Process Modeling Concepts

This section covers the concepts involved in process modeling in TIBCO Business Studio. See the *TIBCO Business Studio Modeling User Guide* for more information.

Projects, Packages and Processes

In TIBCO Business Studio there is a hierarchy consisting of project, package, and process:

Projects

TIBCO Business Studio supports the full project life cycle, bringing together all artifacts in a single place. The *project* is the container for these artifacts. As such, projects help to facilitate sharing and organization of resources.

For example, team members may have different responsibilities but need to use the same resources that are made available through the TIBCO Business Studio project.

You must create a project to use TIBCO Business Studio. Each project has a corresponding directory in the file system (specified when you create the project). Projects can also refer to other projects (see Project References on page 17).

Project Lifecycle

When creating a project, you can assign it a version using standard Eclipse format:

major.minor.micro.qualifier

The specified version will be the default for project artifacts such as process packages and organization models, and can be used to indicate revisions to the project. The exact use of project versions should be coordinated with the solution designer working on the project, however the following are general guidelines:

major Incremented when the new version is incompatible with the previous version. Note that all references within a project to an organization model must be to the same **major** version of the model.

minor Indicates a compatible revision.

micro Indicates an internal change.

qualifier Used to identify unique builds may use time format or other convention.

Consult with your solution designer to establish a common practice.

Packages

A *package* is a mandatory container for processes and their infrastructure (such as *participants* and *data fields*). The package and any processes stored in it are saved in XPDL format.

For example, in the insurance environment, separate packages could contain the processes used by the Claims Department, the Policy Origination/Maintenance Team and the IT Department. Processes can be shared between packages and projects so libraries of process components can be created and reused..

If the *Data Fields* folder is empty, it will be hidden by default. This is because the preferred usage is to define Data Fields at the Process level.

Processes

There are two types of process in TIBCO Business Studio, the business process and the pageflow process.

- A *business process*  models actual and future processes in your organization that usually involve more than one person. Business processes are short or long-lived.
- A *pageflow process*  is a short-lived process (always executed in a single sitting) designed to implement a user interface dialog. It is always executed by one person (the person that initiates the process instance).

The Process Editor provides tools on a palette that use Business Process Modeling Notation (BPMN). By creating your process this way, you can fully prepare it for implementation by a specialist in your organization.

Some objects such as *business assets* can be shared at the project level. Others such as *data fields* and *participants* can be created at either the package level (where they can be shared amongst processes in that package), or at the individual process level (where they can only be used by that process).

If the Participants folder is empty, it will be hidden by default. This is because the preferred usage is to define Participants at the Package level.

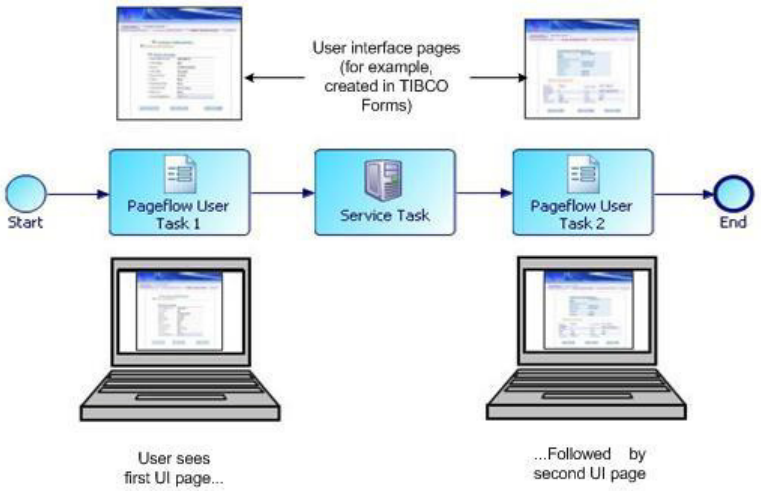
Pageflow Process Modeling

A *pageflow process* is a short-lived process designed to present user interface pages to the user in sequence. They are always executed by one person (the person that initiates the process instance).

All tasks that are available in a business process are available within a pageflow process with the exception of a *business process user task*. Pageflow processes have a special variant of a business process user task that does not have participants, and does not generate work items. These are referred to as *pageflow user tasks*.

A pageflow process is stored under the **Processes** branch of the Project Explorer alongside business processes.

For example:



In this example, the user is presented with the user interface page (in this case a form created using TIBCO Business Studio Forms) associated with pageflow user task one. When the form is submitted, the service task runs. When the service task is completed, and the user interface page associated with user task 2 is displayed. The user is not aware of the service task, and sees one form followed directly by the next one.

A user task in a pageflow process differs from a user task in a business process in several key respects:

- Pageflow user tasks do not have participants assigned to them (this is because the user who initiates the process instance completes all the tasks in the pageflow process).
- Pageflow processes cannot contain lanes or pools.
- Pageflow user tasks do not create work items. The user interface pages are presented to the user without them needing to access their work queue.

- Pageflow user tasks do not restrict the type of technology used to create the user interface page that is displayed. For example, you could use TIBCO Business Studio Forms or a different technology. This allows the same process to be deployed to several runtime environments that utilize different user interface display technologies.

There are also special considerations to observe when using pageflows, specifically:

- Pageflow processes are not persistent (if the user cancels out of a pageflow process, data entered to that point is lost).
- Pageflow processes are not audited.
- Pageflow processes are not transactional (for example, there is no provision to roll back changes if a service task fails). If transactional control is required, chaining might be a better choice than a pageflow process (see Creating a New Embedded Sub-Process on page 184).

Process Interface

A *process interface* specifies the events and their parameters that must be present in processes created using that interface. At runtime, any of the processes that implement the interface may be chosen based on data available at that time.

The use of a process interface allows the dynamic selection of sub-processes at runtime, promoting separation of the design-time and runtime environments.

You can use a process interface with a business process or a pageflow process.

A process interface consists of start events, intermediate events, and their associated formal parameters and errors.

Start Events

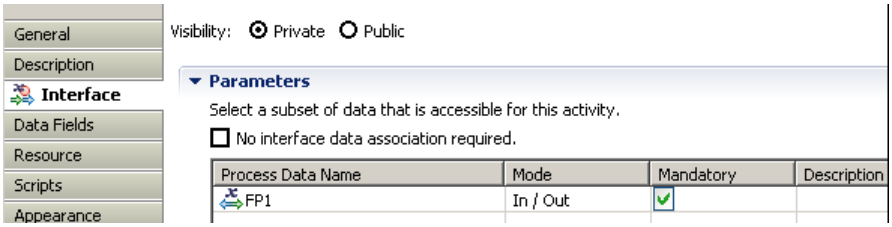
A process interface must have one or more start events (one is automatically created by default) that specify how a process instance is initiated.

When more than one start event is specified, a process implementing the interface can be invoked by any of the start events specified in the interface. As per BPMN, multiple separate start events are considered exclusive (each starts a new process instance).

The start event for a process interface can be of type None or Message:

- A maximum of one start event of type None can be specified. Use this type of start event for invoking a sub-process using a reusable sub-process task or when a sub-process is meant to be manually invoked. See Dynamic Sub-Processes on page 31.

- Use message start events for the invocation of the process using a message based interface (for example, a web service operation).
- A start event for a process interface can be associated with mandatory formal parameters using the **Interface** tab. For example:



If a process instance is invoked using this start event, the parameter **FP1** must be specified.

Both input and output (including combined in/out) parameters can be associated with the event. Specifying output parameters means that these are the parameters should be returned when the process is invoked using this start event.

Intermediate Events

The intermediate events for a process interface can be of type None ~~or Message~~:

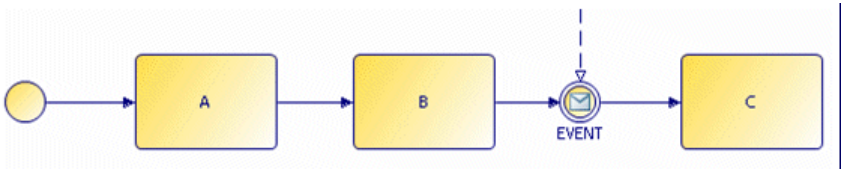
- **None** Use this type of intermediate event for manual event triggering.
- **Message** Use this type of intermediate event for triggering using a message based interface (for example, a web service operation).

An intermediate event can be associated with input or input/output formal parameters that should be present when triggering the event, as well as with errors that will be translated into catch error end events in implementing processes.

Output formal parameters cannot be associated with catch message intermediate events.

Event Persistence

Because a process interface does not specify the intended location of the event within the flow of the process implementation, events are considered "persistent". As such, they can be triggered at any point in the process flow that is upstream of the event, or when the flow is paused waiting for the event to be triggered. For example:



The trigger for the intermediate event can be received while Task A or B is being processed, or when the flow is paused at the intermediate event waiting for the event to be triggered. In either case, Task C is not processed without an event trigger. If the trigger is received during processing of Task A or B, the trigger arrival is persisted, and the event is triggered immediately when it is reached in the flow.

Errors

TIBCO Business Studio allows you to specify errors that may be thrown by a process that implements the process interface. This is useful where:

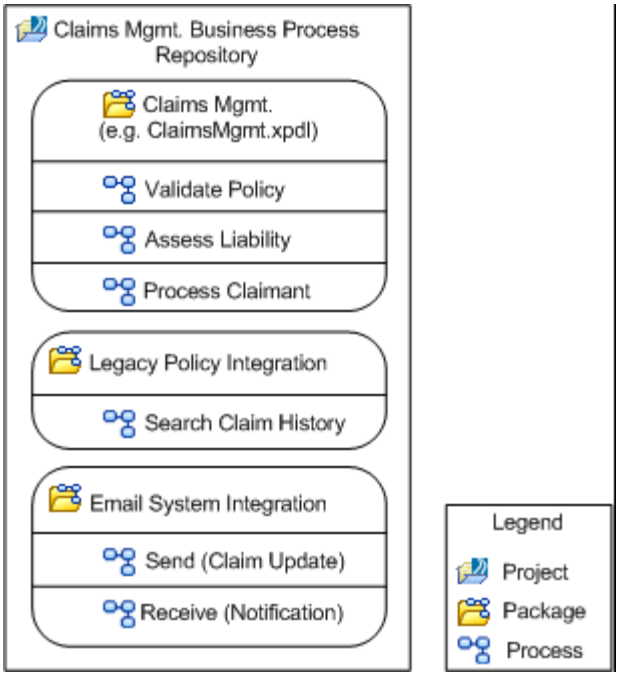
- The process interface defines a messaging or web-service interface. In this case, the error events are translated into WSDL operation fault messages.
- The process interface is used for dynamic sub-process invocation. In this case, the error events are translated into end error events in process implementations. Independent sub-process tasks that reference process interfaces can then identify the thrown error events directly from the process interface.

Process Components

Process components represent reusable building blocks that encapsulate the management of a particular item in a business process. The process components form a reusable library that you can call upon in different contexts.

For example, you might have an item to "verify the caller’s address/contact details" in the business process for taking out an insurance policy. This could be implemented as a sub-process and this particular process component could be used in the context of renewing an insurance policy.

The following example shows a project and the associated packages and processes used in an insurance environment.



In this example, the Validate Policy process might call a sub-process in another package (for example, the Search Claim History process). This sub-process is in the same project in this example, but it could be located in a different project.

Process Migration

Process migration provides the ability to migrate a long running process instance from one version to another version of the same process. In other words, migrated process instances will continue execution using the new process definition.

Project References

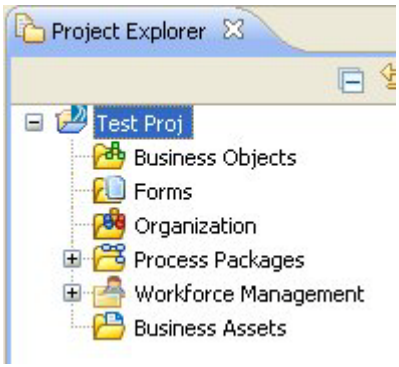
Projects can refer to other projects, and you can add project references explicitly or automatically.

Allowing projects to refer to other projects means you can do the following:

- Invoke a sub-process from another project (see Reusable Sub-processes on page 29).
- Use a process interface from another project (see Process Interface on page 13).
- Create a data field or formal parameter of the type external reference (see Data Fields and Parameters on page 68).
- Select a WSDL from a different project.

Assets

When you create the project, you can decide which types of assets to include, and also designate a special folder for each asset type. For example, the default analysis project has special folders for business objects, forms, organization models, process packages, and business assets:



For more information about special folders, see Special Folders on page 21.

Assets include XPDL package files, WSDL files, documents, business object models, and so on that relate to the project, and they are usually stored in special folders under the project.

Because TIBCO Business Studio is extensible, there may be other types of assets displayed. For more information about special folders, see Special Folders on page 21.

Business Process Assets

Business process assets include the XPDL package file and all the associated processes, data fields, parameters, and so on. The default special folder for business processes is called **Process Packages**.

Forms Assets

Forms assets include all the files and parameters necessary to display the forms that you have created in your project. These include XML, HTML, and properties files, as well as the parameters used by the forms. The default special folder for these assets is called **Forms**.

Organization Model Assets

Organization Model assets include the organization model and its components, such as organization units, positions, groups, and so on.

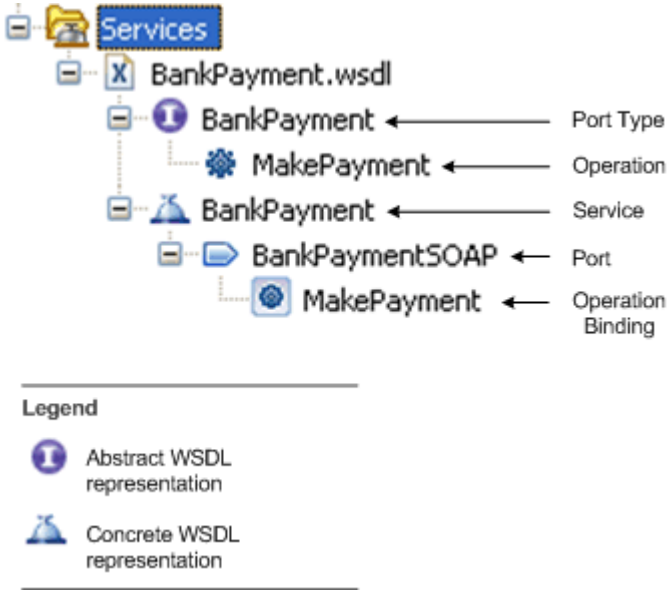
Service Assets

Service assets include the WDSL files for any web services that you import into your project. The default special folder for these assets is called **Service Descriptors**. You can specify either an abstract or concrete WSDL file:

An Analysis project does not have a service descriptors special folder by default. If you need to create a special folder, see Special Folders on page 21.

- 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:



Business Object Model Assets

Business object model assets include the files for any business object models that you create.

Business Assets

There are two categories of project-related business assets in TIBCO Business Studio:

- **Quality Process** Business cases, project plans, and so on.
- **Ad-hoc Assets** Supporting documents, spreadsheets, and so on that are not part of the quality process.

Special Folders

Special Folders are folders in the Project Explorer that are reserved for storing specific types of assets. For example, the **Process Packages** folder is the default special folder for storing processes and the packages that contain them.

When you create a project, you have the option of creating a special folder for each type of asset that you include in your project. By doing this, you can utilize special features of the Project Explorer. For example, enabling a special folder for business processes allows you to view the participants, data fields and so on. If you do not use a special packages folder, you only see the XPDL file for the package in the Project Explorer. Another example of unique special folder behavior is the **Services** folder, which allows you to expand the operations of contained WSDL files:

Special Folders and Asset Types

The following table shows the asset types and the default special folder for each.

Default Special Folder	Asset Type
Business Assets	Prince2 templates, quality templates
Business Objects	Business object and concepts models
Forms	Forms generated by *TIBCO Business Studio Forms*
Organization	Organization models
Process Packages	Business process, packages
Business Objects	Business object model
Generated Services	WSDL file automatically generated for a business process
Organization Models	Organization Models

Default Special Folder	Asset Type
Service Descriptors	Imported WSDL files for services

Process Fragments

Rather than reusing an entire process, you can use the **Fragments** view to quickly create new processes.

TIBCO Business Studio provides predefined fragments in the folder **BPMN Process Fragments**. For example, **Basic Fragments** contains simple fragments such as an embedded sub-process, split conditional, and task sequence.

You can also create your own process fragments. For example, there may be process patterns that you frequently use. By storing these patterns or "fragments" you can easily use them to construct new processes.

Custom Fragments

The Fragments that you create are stored in the category you specify and can be used to create new processes in the same way as pre-defined fragments .

You can drag fragments and drop them into other categories.

Exports from TIBCO Business Studio

You can export documentation and artifacts from TIBCO Business Studio.

Documentation

From within TIBCO Business Studio you can export information about your entire project, or your Organization Model, Business Object Model, or Process and the Package that contains it, to an HTML file. You can also use the command line to create documentation.

Work Data Models

A Work Data Model is an artifact that you can export from a TIBCO Business Studio project. It contains XML data definitions that a custom client application may need to enable it to handle XML data generated from user activities defined in the project processes (for work items, pageflow pages or business service pages).

See ’Work Data Models’ in the *TIBCO ActiveMatrix BPM - BPM Developer’s Guide* for more information.

A Work Data Model Export contains:

- the generated schema for the project's BOM file, as well as schemas from any dependent projects.
- work type definition file (wt.xml).
- work model definition file (wm.xml)
- a file containing the data interface definition for all user activities used in page flows (pfActivityTypes.xml)

Activities

An **activity** represents work that a company or organization performs using business processes. An activity can be atomic (it is not broken down into a finer level of detail) or non-atomic. Atomic activities are represented in the Process Editor by tasks. For more information about how BPMN defines activities and tasks, see http://www.bpmn.org.

When creating the TIBCO Business Studio Process, each time a different person, group, role, or system does something, an activity is added to the Process.

Activities may be triggered by events such as the receipt of an email, phone call or workflow item, and may involve making a judgement on the presented facts and performing an action (such as entering data to a computer system, phoning someone in the same or a different organization, and so on).

A task in a process diagram represents an atomic activity (one that cannot be further broken down). A task of an unspecified type looks like this in the Process Editor:

If the activities can be broken down into finer steps, they should be represented as Sub-Processes. The Activity Type is set in the **Properties** view.

User Tasks

User tasks are those that require human interaction with a software application.

User tasks can be further configured for inbound and outbound parameters. Forms can be generated from the task's input and output parameters, representing the information you want to present to and capture from the user.

You can also generate a pageflow process from a task.

TIBCO Business Studio Forms enables you to design, view, and test the forms you need to collect user input in a business process. You can create sophisticated forms without programming, and associate them with user tasks in order to provide richer user experiences for business process participants.

Manual Tasks

Manual tasks are those that are completed by a person without using software.

Service Tasks

Service tasks can ideally complete without human interaction (for example, an automatic email notification or a web service).

Script Tasks

Script tasks contain a set of instructions written in a scripting language (usually added to the step by the solution engineer) that will be executed in the runtime environment when the process is deployed and executed.

Using the business analysis capability, you can add text to a script task to describe the desired behavior of the script. However, by switching to solution design capability or by clicking **Provide Implementation Details** in the Properties view for the task, you can enter JavaScript to a script task. For more information, see the relevant implementation guide.

Script tasks and service tasks can be used to achieve the same results, however TIBCO recommends that only small and simple tasks are implemented using script tasks because properly-written services are easier to maintain and test.

Send Tasks

Send tasks are used to send messages to a system or person outside of the process (often using a web service):

They can be paired with a receive task or message event to form a request response operation.

Receive Tasks

Receive tasks are used to wait for a message from a system or person outside of the process (often using a web service). This type of task can be used to start a process as long as it has no incoming sequence flow and there are no start events in the process:

On the **Interface** tab, you can add parameters to a receive task; however, you cannot add data fields because data fields are used internally in a process, and parameters are in this case, inputs from an external process (for more information, see Data Fields and Parameters on page 68).

Receive tasks can be paired with a send task or message event to form a request response operation.You must also ensure that incoming messages are received by the correct process instance. For more information, see Correlation Data on page 71.

Reference Tasks

Reference tasks refer to another task and prevent you from having to duplicate the same task several times in a process.

You can create a reference to another task using either the **Reference Task** gadget on the reference task, or in the Properties view for the reference task.

Activity Markers

You can select BPMN Activity Markers on the Properties of the activity. The currently selected Activity Marker is indicated by the following symbols on the activity:

- **Multiple Instance Loop**  Indicates a task or sub-process that is replicated a fixed number of times based on the evaluation of an expression. The ordering can be either parallel or sequential.
- **Standard Loop**  Indicates a task or sub-process that may have more than one instance, depending on the conditions of the loop. A standard loop consists of a Boolean expression that is evaluated before or after each cycle of the loop. If the expression evaluates to True, the loop continues.

  The conditions of the loop are set on the **Loops** tab (for more information, see Loops on page 51).
- **Ad-hoc**  Indicates an embedded sub-process that contains activities that have no pre-defined sequence. This also means that the number of times the activities are repeated is completely determined by the performers of the activities and cannot be defined beforehand.

Presentation Channels

Presentation channels control how tasks will be displayed to the user after deployment.

When you use Presentation Channels, you can choose between Workspace Google Web Toolkit, Openspace Email and Openspace Google Web Toolkit which are provided by default. You can also choose to add channel types, or create your own presentation channel. See the TIBCO Business Studio *BPM Implementation Guide* for more information.

Sub-Processes

Some activities can contain further steps, or sub-processes.

There are two types of sub-process that are described in this section, embedded and reusable..

A sub-process that is embedded can be re-factored into a reusable sub-process and vice versa. For more information see Refactoring Sub-Processes on page 31.

Reusable Sub-processes

When identifying the process components you require for your business process, it is important to analyze the entire business. One way to do this is by thinking in terms of the products, services and resources (by considering the specifications for each).

To facilitate the re-use of process components, an activity (or several activities) can call another process as a reusable sub-process. The reusable sub-process could be a process that you have already created, or you can refactor activities in your current process into a reusable sub-process (see Refactoring Sub-Processes on page 31).

Activities that contain a reusable sub-process look like this in the Process Editor:

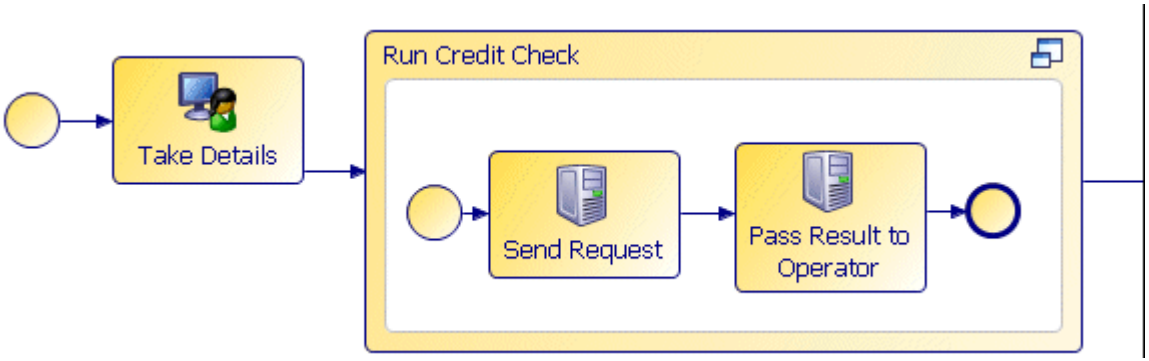
Click the plus sign (+) in the activity to view the sub-process.

An activity of this type defines a call-out to another process:

- The called process exists as a separate process from the parent process, and because of this it can be started from other processes.
- The called process does not have access to data fields and parameters of the calling process and package. for this reason, data mapping must be performed to and from the called process.

Embedded Sub-processes

An embedded sub-process is one that is fully contained within the parent process; it does not exist as a separate process:

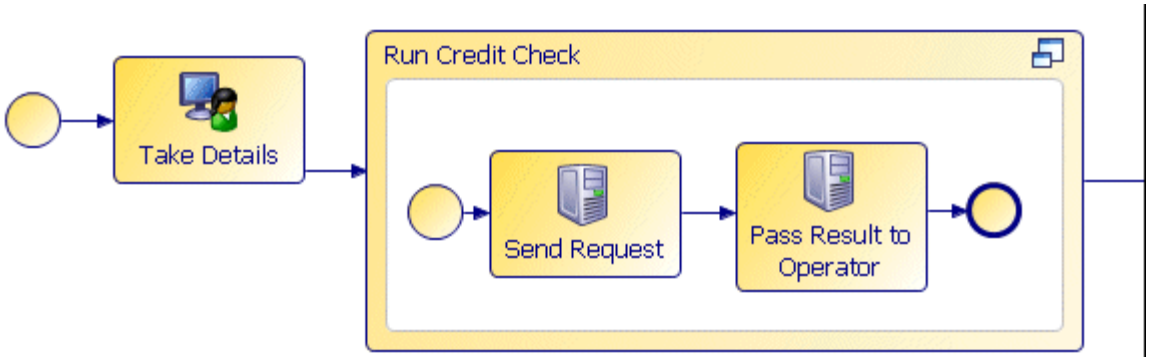


In this example, Run Credit Check is an embedded sub-process. This implies that running a credit check is an activity that is not needed by other processes. If you subsequently decide that you want to be able to run a credit check from within other processes, you can expose the embedded sub-process as a reusable sub-process by refactoring it. See Refactoring Sub-Processes on page 31.

An embedded sub-process has the following characteristics:

- It is fully contained within the parent process, and is executed within the parent process.
- Activities within the embedded sub-process have access to the same data fields and parameters as the parent process and package.
- No data mapping is required.
- It cannot contain lanes and pools.

In this example, Run Credit Check is an embedded sub-process. This implies that running a credit check is an activity that is not needed by other processes. If you subsequently decide that you want to be able to run a credit check from within other processes, you can expose the embedded sub-process as a reusable sub-process by refactoring it. See Refactoring Sub-Processes on page 31.



To create an embedded sub-process, refactor one or more objects in your process as described in Refactoring Sub-Processes on page 31.

If you want to use the chained execution resource pattern, you can do so by selecting the **Chained Execution** check box in the Properties view for the embedded sub-process. For more information, see Chained Execution on page 39.

Refactoring Sub-Processes

Refactoring a sub-process allows you to do the following:

- Create a new embedded sub-process from selected objects.
- Create a new reusable sub-process from selected objects and replace the selected objects with a call to the newly created sub-process.
- Convert an existing embedded sub-process (and its contents) into a reusable sub-process and replace it with a call to the newly created sub-process.

Dynamic Sub-Processes

Dynamic sub-processes are used when a process (which can be either a business process or a pageflow process) calls one of several sub-processes at runtime, but it is not known at design time which of these sub-processes will be called. The exact sub-process to be called on any given occasion is chosen at runtime, depending on the process data. For example, a corporate HR Department's recruitment process might need to call different sub-processes for determining a candidate's eligibility for employment depending on the country where the candidate is recruited.

In order for the main process to be able to accommodate any of the sub-processes that might be called, the sub-processes must all take and return a common set of data, which is known at design-time. This common data is specified by a process interface on which all the sub-processes are based. A Reusable Sub-Process task in the main process then specifies a call to the process interface instead of naming an individual sub-process.

Transactions

A sub-process (either embedded or independent) can be specified as a transaction using the **Is a transaction** check box. This means that the behavior of the sub-process is governed by a transaction protocol (defined in the runtime environment). This is indicated by a double-line around the activity:

Typical outcomes of a transaction sub-process that you should cater for are success and cancel. For more information, refer to the BPMN specification (see http://www.bpmn.org).

Participants

Participants are used to identify who or what performs an activity.

For example, in a hiring process, a person (Human Participant) interviews the candidate and an email system (system participant) sends out an automatic follow-up reminder. There are the following options for creating participants:

- Use the basic types of participant that are provided by TIBCO Business Studio.
- Create your own organization model, and use the positions from the organization model for your participants.

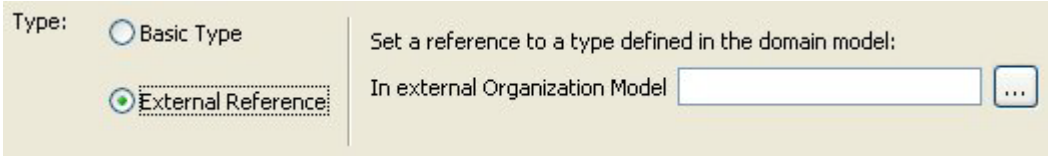
You can also refactor the participants in your process into an organization model.

There are several basic types of participant:

- **Role** - identifies the role responsible for performing an activity. For example, in a financial institution there may be roles such as Reconciler, Manager, and so on.
- **Organizational Unit** - identifies the department or unit within an organization that performs an activity. For example, Legal, Marketing and so on.
- **Human** - identifies a specific person or user that performs an activity.
- **System** - identifies an activity that is performed by a system.
- **Organization Model Query -** allows you to enter a query using a script or expression. This is evaluated when a referencing task is executed at run-time, so the actual participant is resolved and the activity dispatched and offered to the participant. A query could resolve to a participant in the package/process or to an entity in the organizational model.

Participants from the Organization Model

You can also create participants by creating an external reference to types defined in a model of your own organization.



You can create an external reference to the following parts of the organization model:

- Positions
- Groups
- Organization Units

- Types from the default meta model

You can also create an organization model from the participants you have defined in your process as described in Participants from the Organization Model on page 33.

Resource Patterns and Work Distribution

This section describes patterns that are available to model how you want work to be distributed to resources. Resources are the people who carry out the work, and are represented in TIBCO Business Studio by participants. How these patterns are interpreted depends on your runtime environment, which may not support all the patterns.

The Workflow Patterns initiative (a joint effort of Eindhoven University of Technology and Queensland University of Technology) provides a conceptual basis for process technology. Their research identifies numerous patterns that can be supported by a workflow language or a business process modeling language. Many of these patterns are supported in TIBCO Business Studio. For more information about the Workflow Patterns initiative, see:

http://www.workflowpatterns.com/index.php

To support the modeling of workflow patterns, the following sections are available on the **Resource** tab in the Properties view:

- Resources (user and manual tasks)
- Distribution Strategy (message events and user tasks)
- Piling (user and manual tasks)
- Separation of Duties (user and manual tasks)
- Retain Familiar (user and manual tasks)

Resource patterns do not apply to tasks within a pageflow process because pageflow processes do not generate work items.

The section **Calendar Reference** allows you to define an alternative to the system calendar.

Enter a Calendar alias in text (content assist will provide aliases that have been used previously in the workspace) or a reference to a process data field that will provide the Calendar alias at runtime.

The information you enter is subsequently used by the Calendar gadget in Openspace. See the *TIBCO Openspace User Guide* and the *TIBCO ActiveMatrix BPM - BPM Developer's Guide* for more infomation on using alternative calendars.

This information can refer to a base or an overlay calendar.

In addition to the patterns on the **Resource** tab, you can also use Chaining, which is configured on an embedded sub-process (see Chained Execution on page 39).

All items in a chained group need to be in the same embedded sub-process, at the same level as the embedded sub-process. You cannot use chaining with nested sub-processes.

You can also access the Retain Familiar and Separation of Duties resource patterns by selecting tasks, right-clicking, and selecting **Resource Patterns**:

Resources

The participant in the Resources section is the same as the participant specified on the **General** tab, and specifies who or what completes the task (see Sub-Processes on page 29).

You can specify the **Initial Priority**, which indicates the relative urgency with which the item should be completed. By default, the priority is 200 (normal level). You can edit this to be one of the values 400, 300, 200 and 100, with 400 being the one which would be processed first.

Distribution Strategy

Set the distribution strategy for the task to offer or allocate the work to a user:

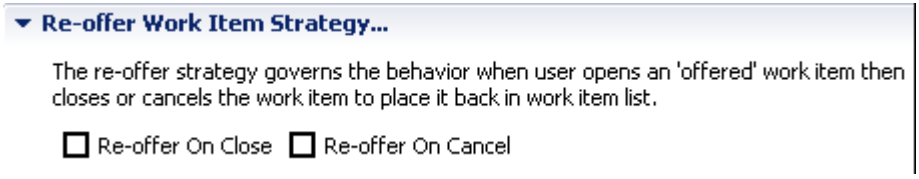
The exact method with which work items are offered and allocated may differ, depending on the runtime environment.

- **Offer To All** Select this option to specify that you want *all* users that match the participant definition to have the opportunity to accept or decline the work item. For example, if there is a claims handler organizational entity (such as a group), the work item is offered to all users in that group. Once a user opens the work item, it is allocated to them and removed from the work lists of other users in that group.

- **Offer To One** Select this option to specify that you want only *one* user that matches the participant definition to have the opportunity to accept or decline the work item. If the user declines the work item, it is offered to another user that matches the participant definition.
- **Allocate To One** Select this option to allocate automatically the work item to a user that matches the participant definition.

Re-offer Work Item Strategy

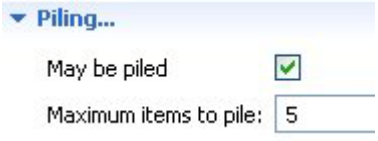
This allows a user task to be configured to re-offer the work item to any valid user (as defined in Distribution Strategy) when the user closes or cancels the work item.



Piling

Specifying that a user task may be piled means that multiple instances of that user task in a user's work queue will be presented to the user in sequence (in preference to other work items).

To specify piling on a work item, select the **May be piled** checkbox and specify the maximum number of items to pile. For example:



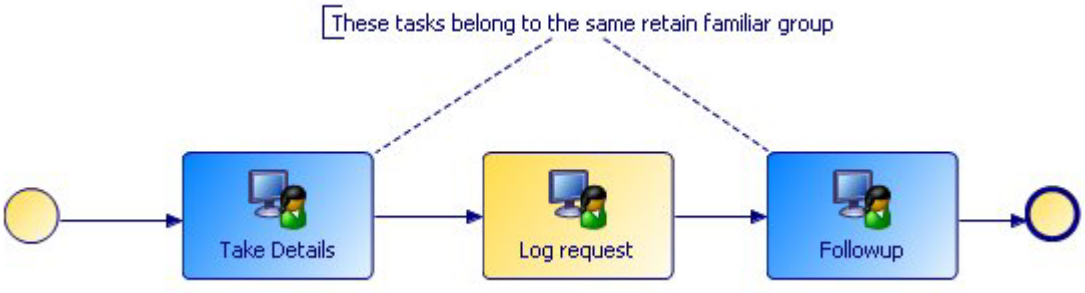
Separation of Duties

This pattern stipulates that you want specific tasks executed by different resources. For example, the resource that prepares a contract is different from the one who witnesses it. There are several ways to specify this pattern:

- On the right-click menu in the Process Editor
- On the **Resources** tab in the Properties view for the task
- On the **Task Groups** tab in the Properties view for the process

Retain Familiar

This pattern stipulates that you want a specific task to be executed by the same resource that executed a previous task in the same process instance. For example, the resource that handles the initial customer contact is the same one that handles the follow-up call. Consider the following process:



In this example, all three tasks are assigned to the same organization unit. The **Take Details** task is allocated to a resource from the organization unit. Because the **Followup** task is in the same Retain Familiar group as the **Take Details** task, they will be allocated to the same resource as well.

This pattern specifies that you would ideally like tasks to be executed by the same resource. However, this is not always possible in the runtime environment. For example, it may be that by the time the later tasks in a retain familiar group are reached, the original participant is unavailable or not in the participant set. In such cases, the work item is delivered as normal according to the distribution strategy.

Where there are parallel threads it is possible for the second task to become active before the other has completed in which case it would be impossible to retain familiar. This situation is disallowed through validation within Business Studio such that the two activities can be guaranteed to be sequential.

If the pattern is broken for any reason the 'familiar' resource becomes the most recent resource used. For example, resource A performs the first step in a process, but is not available when the second task is allocated, so the task is allocated to resource B. When the third task is allocated, it will be allocated to resource B, who has now become the 'familiar' resource.

There are several ways to specify this pattern:

- On the right-click menu in the Process Editor
- On the **Resources** tab in the Properties view for the task
- On the **Task Groups** tab in the Properties view for the process

Chained Execution

This resource pattern specifies the intention that the workflow engine should automatically start the next work item in a case once the previous one has completed. For more information, see the workflow patterns web site:

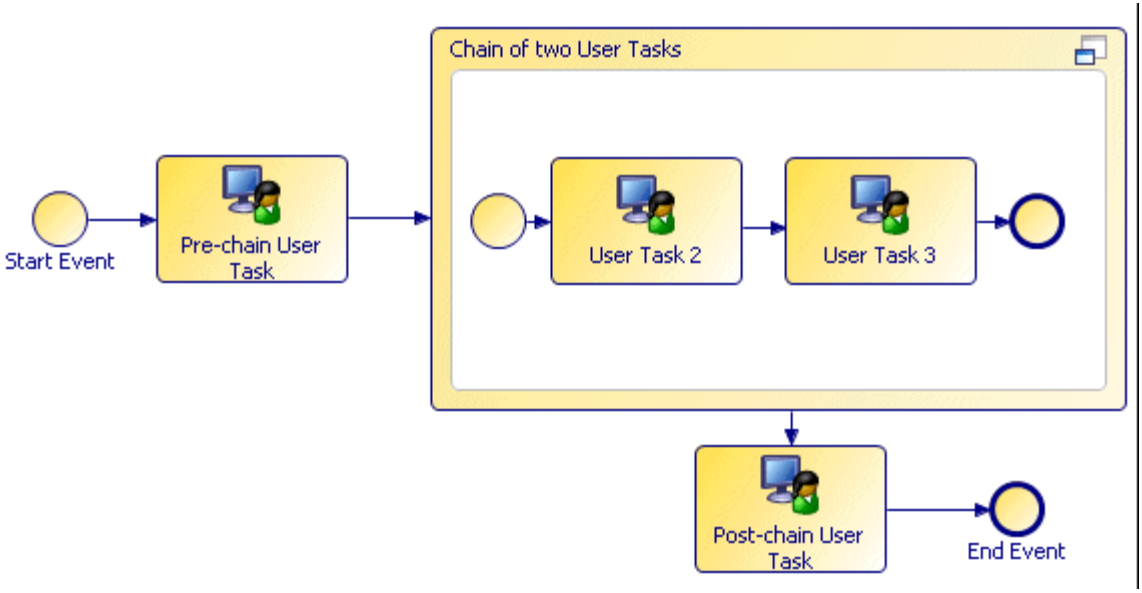
http://workflowpatterns.com/patterns/resource/autostart/wrp39.php

This has the effect of a resource being allocated sequential work items within a process instance, and when a work item is completed, the next task is immediately initiated. This keeps the resource constantly processing a given process instance.

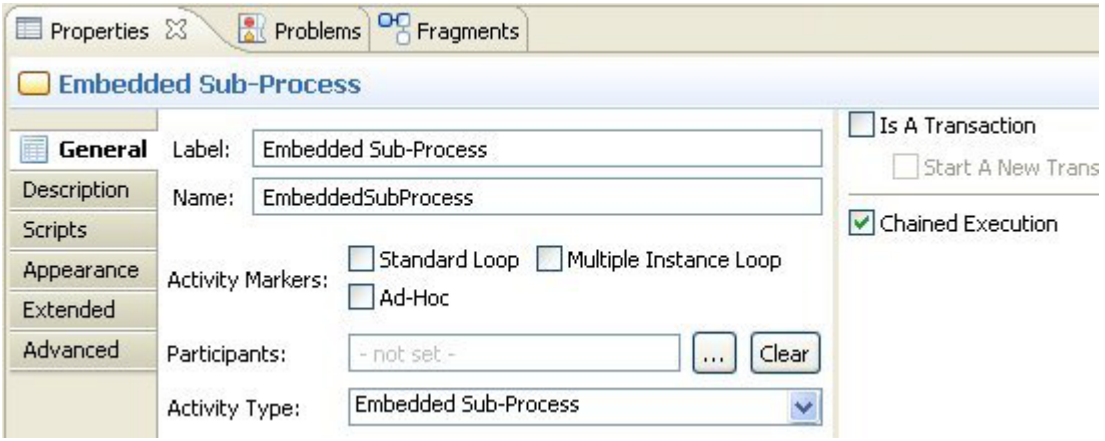
To provide chained execution, configure an embedded sub-process as follows:

You can refactor existing tasks into an embedded sub-process as described in Embedded Sub-processes on page 30.

The following example shows a very simple chain of two user tasks in an embedded sub-process. Note that you can include other types of task, such as script tasks, in between the chained user tasks. However your system administrator must take into account the time likely to be taken up by these other tasks when setting the properties in the "Chaining" section of the **WPProperties.properties** file. See the section "Configuring Chained Work Patterns in TIBCO ActiveMatrix BPM" in the *TIBCO ActiveMatrix BPM - BPM Administration* guide for details of this file.

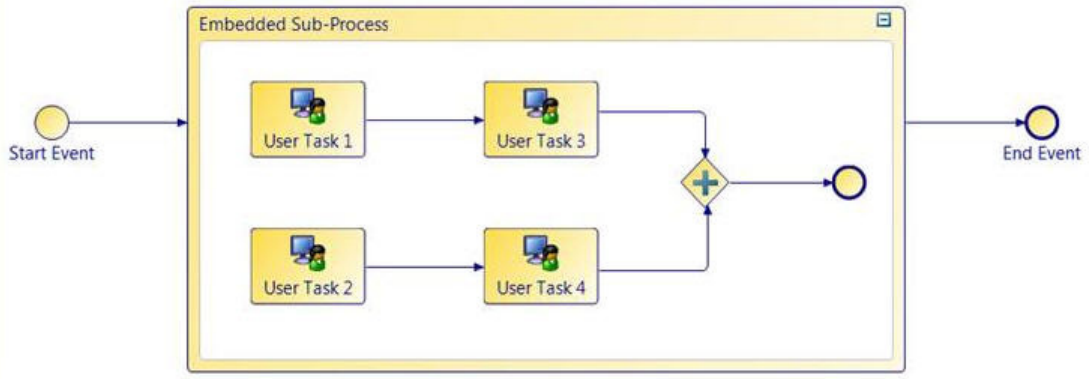


In the Properties view for the embedded sub-process, select **Chained Execution** in the Properties view:



Multiple Parallel Paths in a Chaining Group

When multiple parallel paths exist in a chaining group, items will be chained in the order they are scheduled, and not the order of process flow. For example:



In the example above, User Task 1 or User Task 2 would be performed first. Both would appear in the relevant user's work list. If User Task 1 was opened first then on completion User Task 2 would be performed (it would be the only user task in the chained group available at that point). It is likely that whilst User Task 2 is completed that User Task 3 would be scheduled. This would be performed and then User Task 4 would be performed last. The user tasks would therefore be performed in scheduled order and not according to the connections between the user tasks.

Making Processes Easier to Follow

There are a number of ways you can make complex processes easier to follow:

- A gadget is a user interface aid that allows you to easily create sequence flows or other links between objects.
- Swimlanes allow you to organize your process.
- Data objects, text annotations and groups, all of which do not affect the sequence or message flow of the process, can be used to make a process easier to follow.

Gadgets

A gadget is a user interface aid that allows you to easily create sequence flows or other links between objects.

The advantage of using gadgets is that it is quick to do, and can be used to perform many, but not all, of the tasks traditionally performed from the palette.

See the *TIBCO Business Studio Modeling Guide* for more information on the different gadgets available.

Swimlanes

Swimlanes consist of Lanes and Flows.

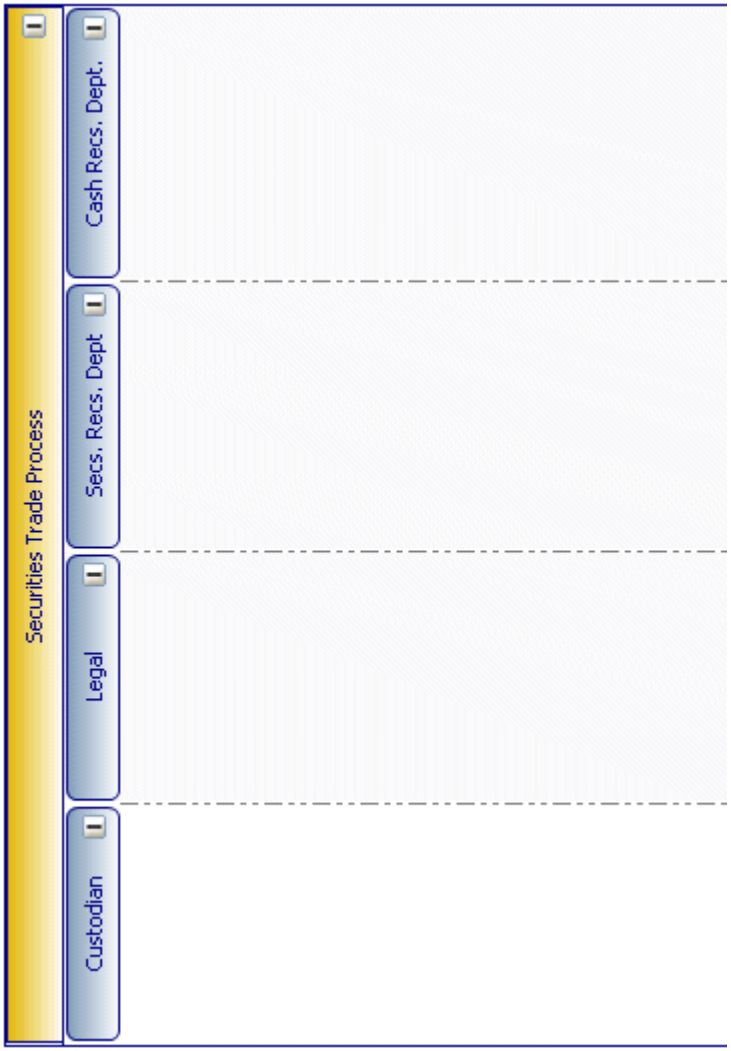
You cannot create or display lanes or pools in a pageflow process.

Lanes

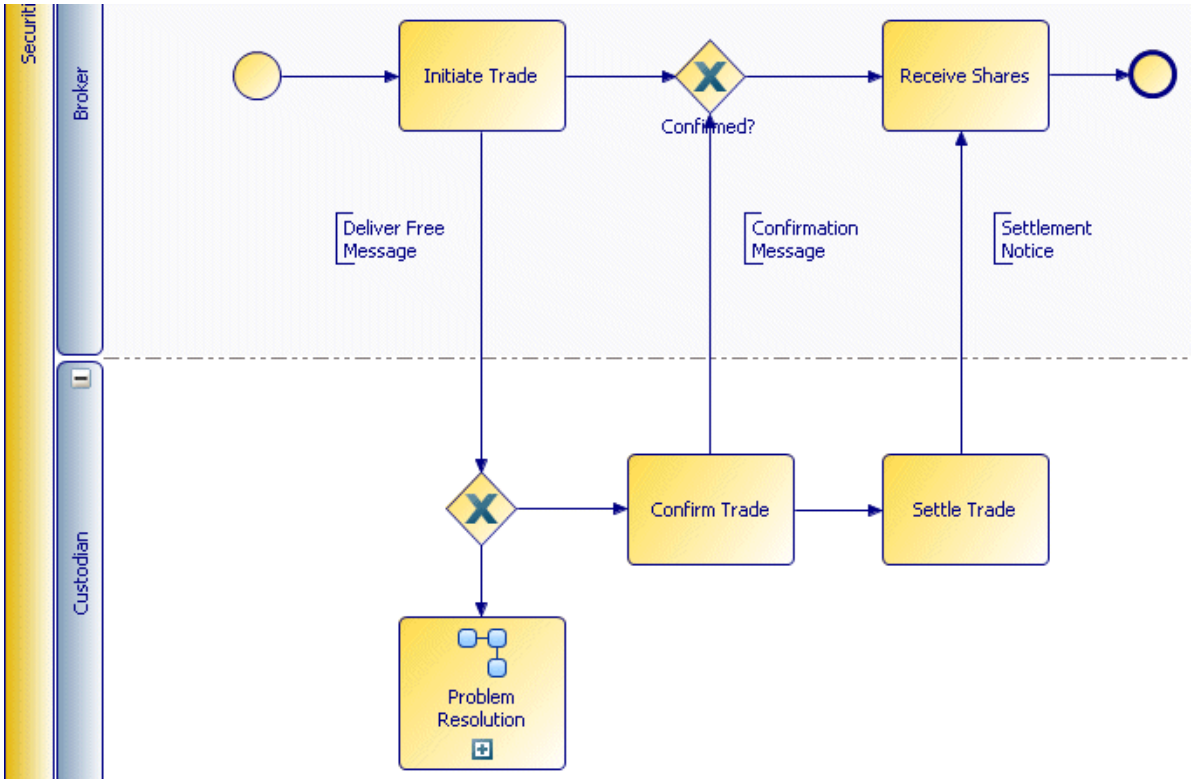
A lane is a subdivision of a pool that is used as a container for partitioning processes. How you use lanes is specific to your business. A lane can be **Normal** or **Closed**. The contents of a normal lane are visible. The contents of a closed lane are not visible and are used for "black-box" processes where you do not know the details of the contained processes.

If you delete a lane in **TIBCO Business Studio**, all the objects in that lane are also deleted. If you do this inadvertently, press Ctrl+Z to restore the contents of the lane.

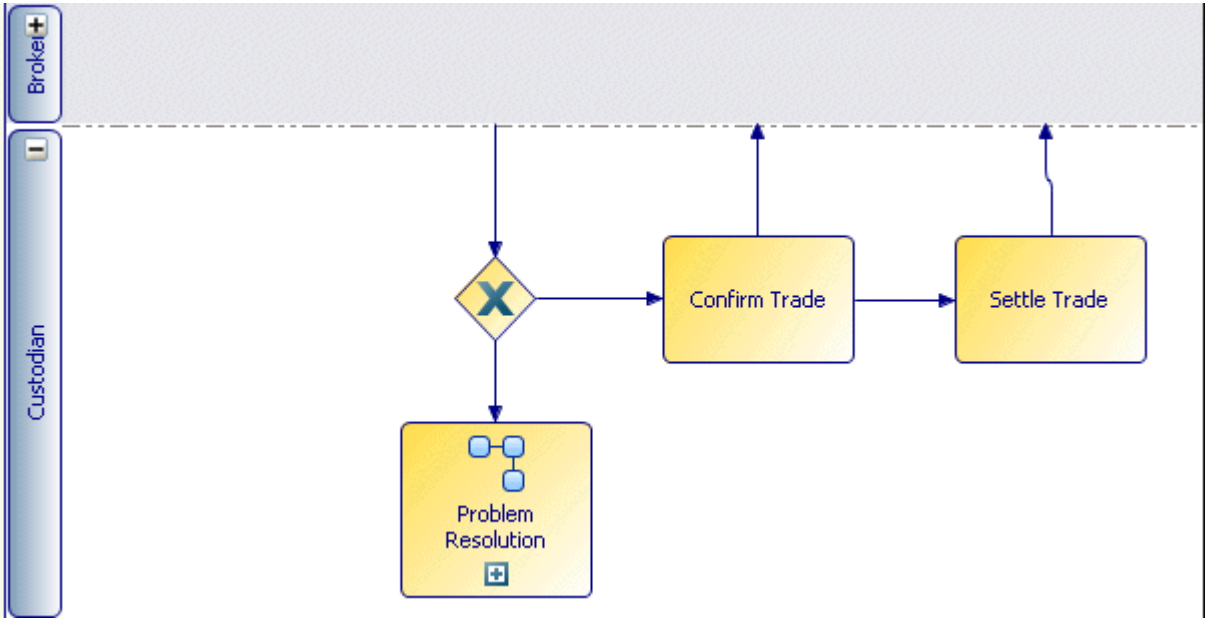
Suppose the back office of a financial institution has a process in which the Securities Reconciliations Department, the Cash Reconciliations Department, the Legal Department and a custodian are all involved. When defining this process, they can be represented by lanes within the pool:



Sequence flow can connect objects in different lanes:



The previous example shows a securities trade between a Broker and a Custodian. The same trade could be represented as follows:



This shows the Custodian lane in a Normal state and the Broker lane in a Closed state. The trade might be represented this way because it is documenting the Custodian's part in the trade and the Custodian has no knowledge of the internal processes of the Broker.

Pools

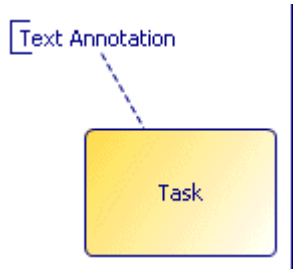
A pool is used as a container for partitioning processes in ways that make sense for your business.

A pool is commonly used to document a process in a self-contained system. Typically the business analyst focusses on just one system or pool, but sometimes needs to show interactions with outside systems. For example, a customer places order and does not know how it is fulfilled in the closed supplier pool, but at some point in the process the customer pool receives a fulfillment response (message).

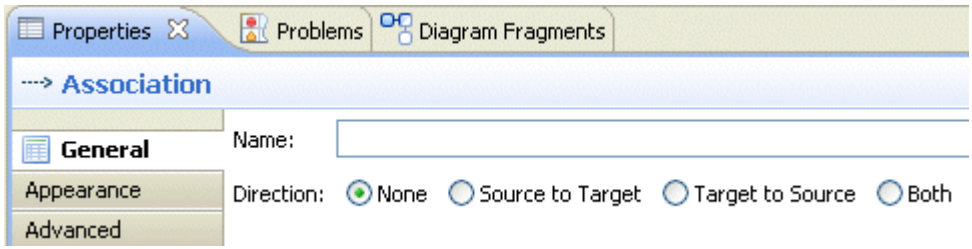
Pools are used in conjunction with lanes and are also related to message flow, which can also be used to show message flow between objects in different pools (see Message Flows on page 49).

Associations

An association is a connection from a data object or text annotation to a flow object (for example, an activity) used to make a process more readable:



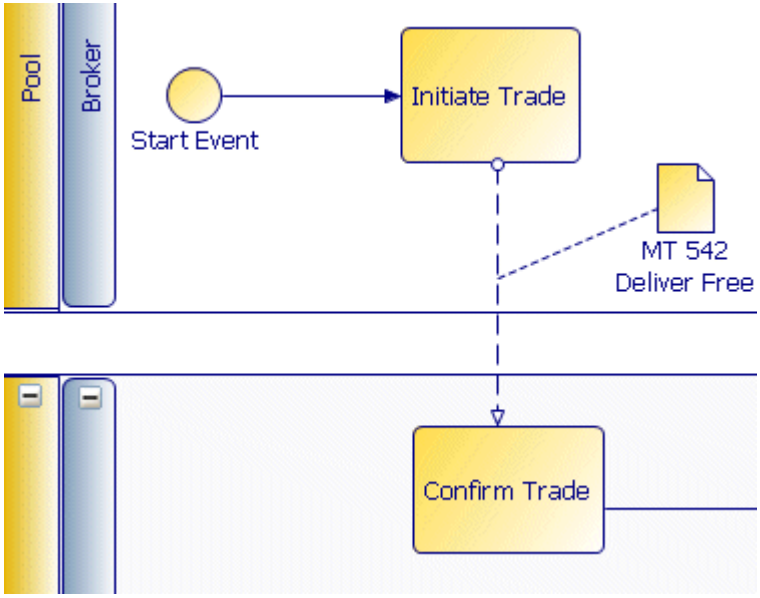
You can change the direction of the association in the Properties view:



Data Objects

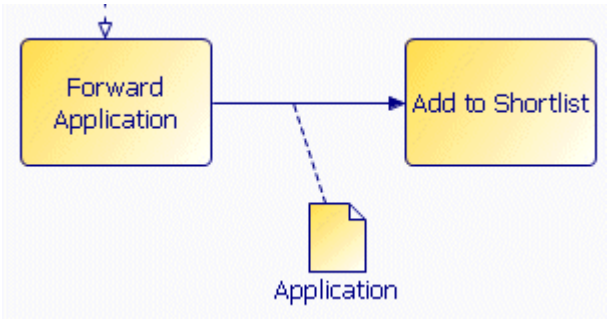
A data object is used for informational purposes to show how a document or other data relates to the process. It may be associated with a sequence flow or message flow, but it does not affect either flow.

Data objects are usually associated with flow objects with an association:



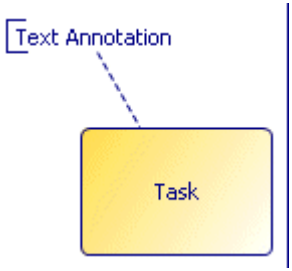
The data object in the preceding example should not be confused with the actual message being sent between the two pools.

A data object can also be associated with a sequence flow or other flow object:



Text Annotations

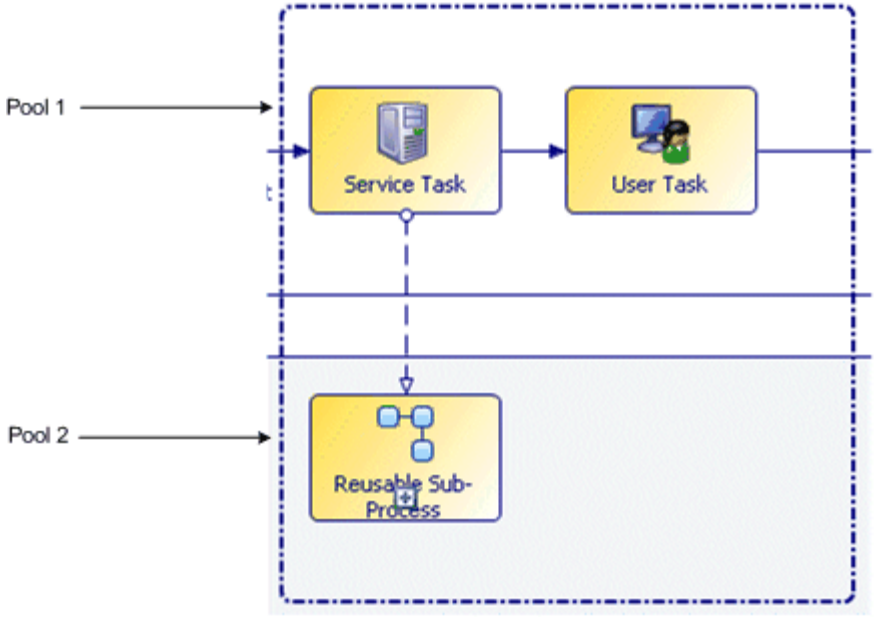
Text annotations serve to explain or clarify the process. They can be connected to flow objects (for example, tasks) using an association:



Unlike any text that you enter on the **Description** tab in the Properties view for an object, text annotations are displayed directly on the process.

Groups

Groups are used to indicate a relationship between elements of a process by enclosing them in a dashed line. A group can span lanes and pools. For example:



Groups are not preserved upon export because they have no meaning in the runtime environment.

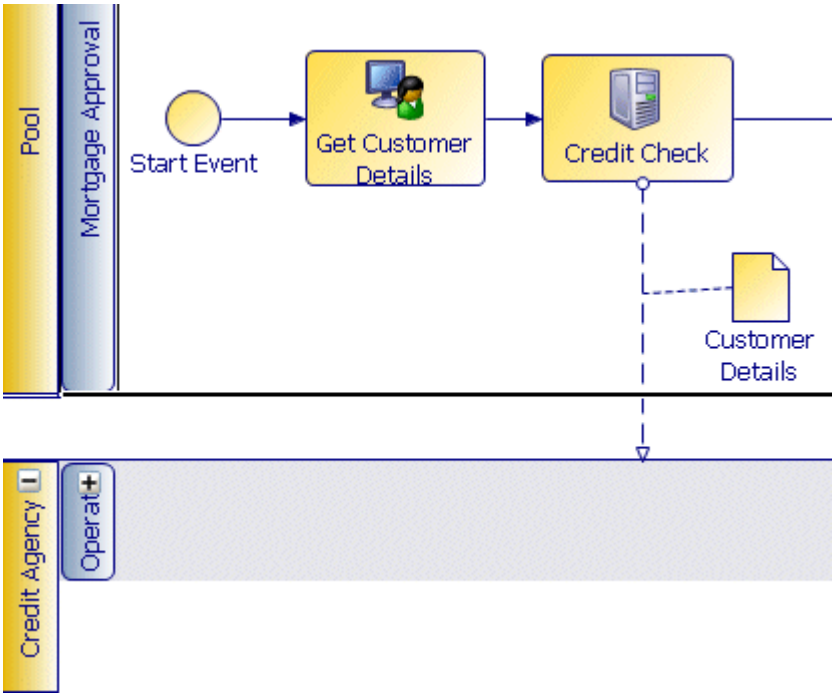
Because the group is a separate layer that is overlaid on the process, it is not deleted when a lane or pool is deleted - even when the group is fully enclosed within the lane or pool.

Flows

Flows in a process can be either mesage flows or sequence flows.

Message Flows

A message flow indicates the flow of messages between objects in separate pools or between pools.

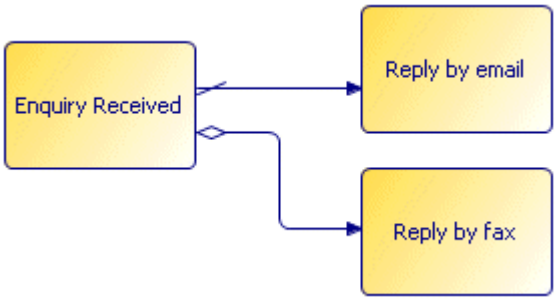


This process shows a mortgage approval process where the customer details are sent to the credit agency for approval using a message. In this case the pool and lane that represent the credit agency process are closed because we either do not care about or are not familiar with their internal processes.

Sequence Flows

Sequence flows indicate the order in which activities will be performed. You can set up sequence flows between flow objects (activities, events, and gateways). When you create a sequence flow, you can highlight it, right-click and select one of the following types of sequence flow:

- Default Flow - Shows the default flow from a gateway or activity that will be taken if no conditional flow has its condition met.
- Conditional Flow - Shows a flow that is only followed if the associated condition is met.
- Uncontrolled Flow - Indicates a flow that does not have a condition associated with it (the default). This path will be taken in all cases.



In this example, an enquiry is responded to either by email (the default), or depending on a condition, by fax.

Loops

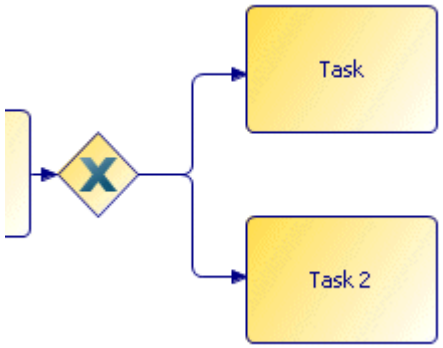
TIBCO Business Studio supports standard and multi-instance loops as defined in BPMN.

- **Standard loop** A standard loop consists of a Boolean expression that is evaluated either **before** (loop while condition is true) or **after** (loop until condition is true) each cycle of the loop.
- **Multi-instance loop** A multi-instance loop has an expression that evaluates to an integer, and is evaluated only once (before the activity is performed). The resulting integer specifies the number of times the activity should be performed.

You can select the type of loop applied to a task either in the Properties view for that task (on the **General** tab), or by right-clicking the task and selecting from the **Activity Markers** menu. The **Loops** tab in the Properties view for the task is where you specify the details of the loop.

Gateways

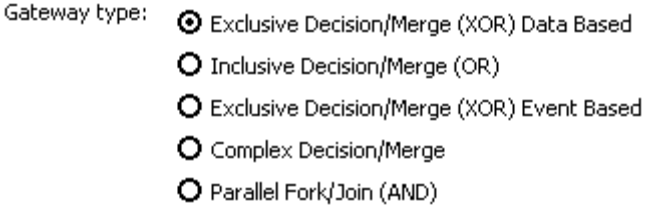
Gateways are a control mechanism for the sequence flow in the process. They are represented by a diamond:



Although the gateway resembles a decision box in a flow chart, gateways provide a variety of behaviors besides conditional decisions.

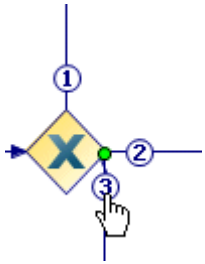
By default, gateways do not have names. This means that you must specify a name for all gateways in your process if you want them to be valid migration points.

As shown on the Properties view of a gateway, these are the different types of gateway that you can create.

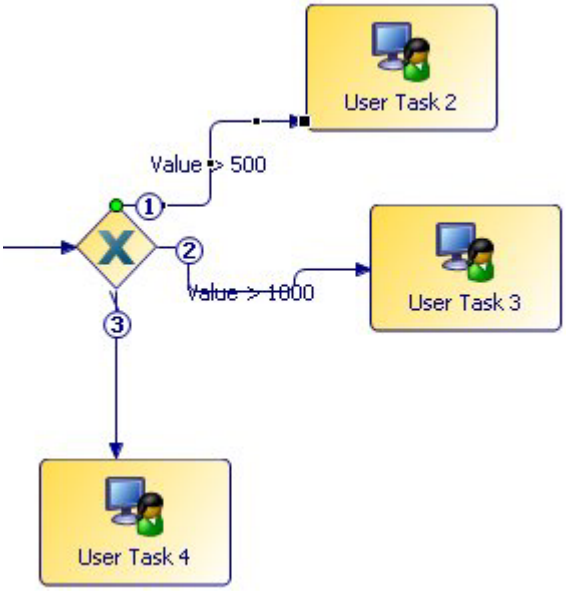


Order of Flow Evaluation

When a gateway has multiple sequence flow output, you can specify the order in which the outgoing sequence flow is processed. You can view the current order by highlighting one of the sequence flows and placing the pointer over the outline numbers that appear. For example:

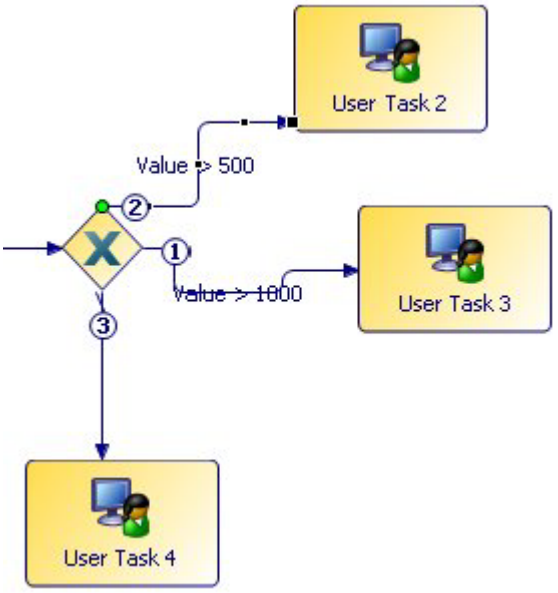


This is especially significant for evaluating the conditions on sequence flows attached to exclusive gateways. For example:



In this case, the **> 500** sequence flow is processed first. Assuming that the JavaScript conditions are set up as their labels imply, if the value is greater than 500, **User Task 2** will be executed. The **> 1000** sequence flow and **User Task 3** will never be reached.

To change the order of evaluation, drag the numbers that appear on the sequence flows. For example, dragging the 2 and dropping it onto the 1 changes the order of sequence flow evaluation as follows:



Regardless of the selected order of evaluation, conditional sequence flows are always evaluated *before* default sequence flows.

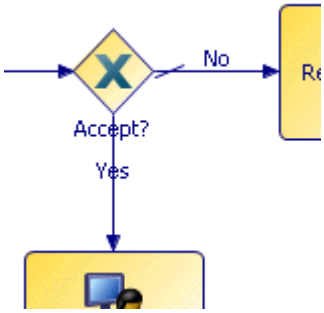
Exclusive (XOR)

In an exclusive gateway, there are several paths through which the process can continue, but only one is actually chosen when the process is run. There are two types of exclusive gateway:

- **Exclusive (Data)** The sequence flow is chosen based on an expression using data from the process. This type of gateway is indicated in the process as follows:

- **Exclusive (Event)** The sequence flow is chosen based upon an external event (for example, a JMS message). This type of gateway is indicated in the process as follows:

The following shows a typical XOR (data) gateway:



There is one uncontrolled input sequence flow to the gateway, and conditional and default output sequence flows.

An XOR (data) gateway displays an **X** as a visual cue to the gateway type. However, because the display of the **X** is not required by BPMN, you can disable the display of the **X** in the Property view for the gateway by deselecting **Show "X" Marker**.

Inclusive (OR)

An inclusive gateway looks like this:

In an inclusive gateway used for branching, each output Sequence Flow is independently evaluated according to an expression. This means that anywhere from zero to the maximum output sequence flows can be taken. In practice, you should either provide a default sequence flow or ensure that at least one sequence flow evaluates to True.

When used to merge flow, any upstream sequence flows are synchronized, but the gateway does not wait for all sequence flows.

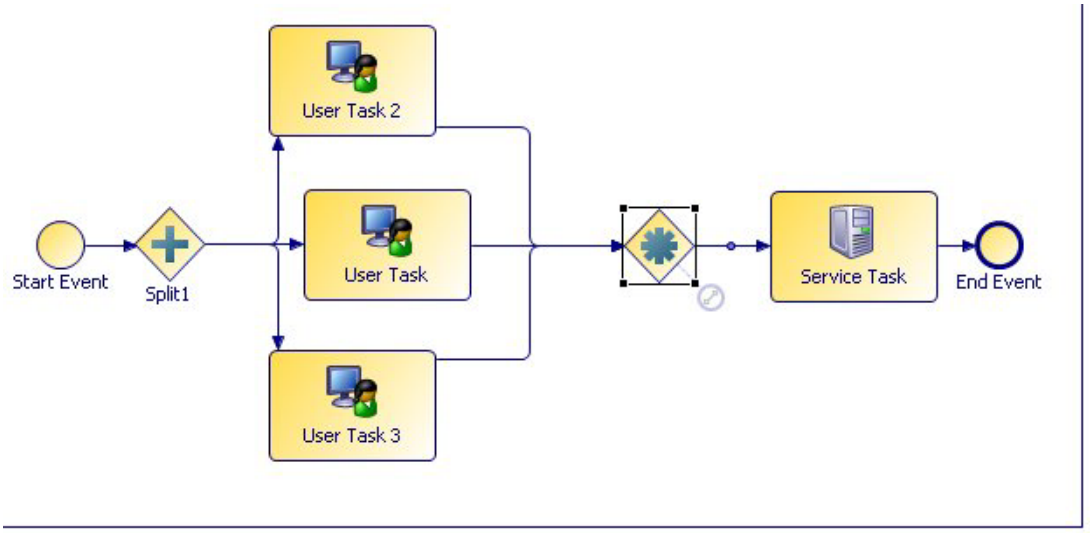
Complex

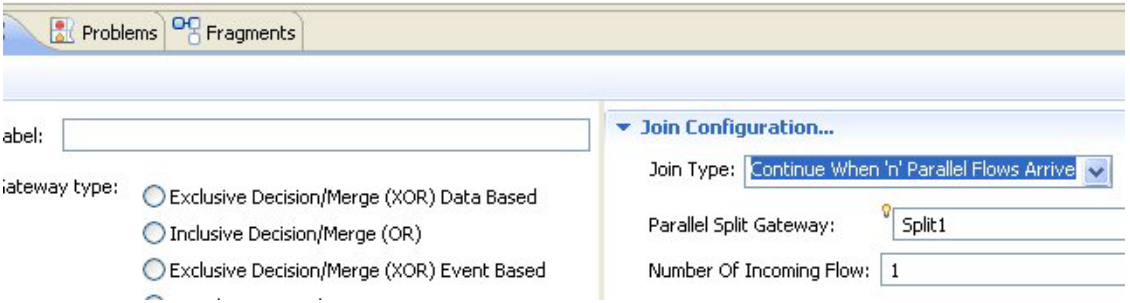
A complex gateway is used to fork or merge depending on how an expression evaluates. When used as a decision, the expression determines which of the outgoing sequence flow are chosen for the process to continue.

When used to merge flow, the expression determines which of the incoming sequence flows is required for the process to continue. This type of gateway is indicated in the process as follows:

Join Configuration

Although TIBCO Business Studio does not provide for entering an expression on a complex gateway, there is a **Join Configuration** section in the Properties view. This allows you to specify how many incoming sequence flows are received before flow continues. For example:





There are three sequence flows going into the complex gateway. On the properties of the gateway, the **Continue When 'n' Parallel Flows Arrive** join type is selected. The parallel gate way is specified indicating that this complex gateway is handling an earlier parallel split (named **Split1**), and that flow should continue when only one of the sequence flows reaches the complex gateway.

Parallel (AND)

A parallel gateway is used to fork or merge several parallel paths (synchronization). When several sequence flows enter a parallel gateway, the process flow waits until all arrive at the gateway before continuing. This type of gateway is indicated in the process as follows:

Events

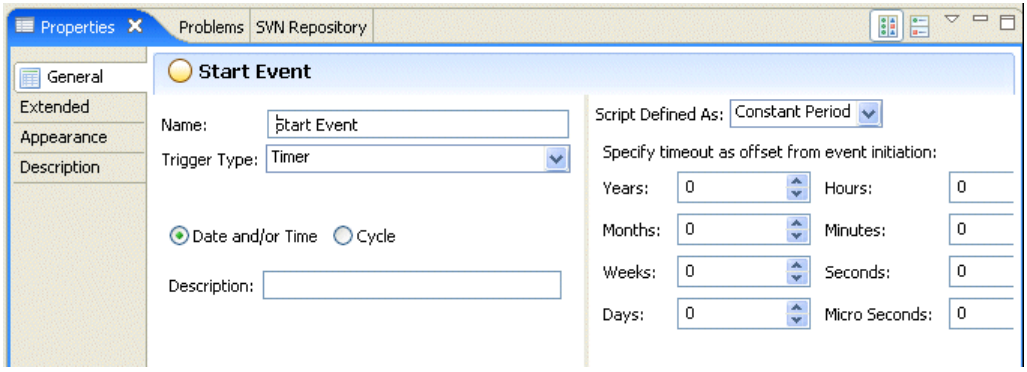
An Event in a process is something that happens that affects the sequence or timing of activities in a process, for example the receipt of a message. There are three main types of Event: Start, Intermediate and End.

- Start events are used to indicate the start of a process or to control how a process is started (or triggered).
- Intermediate events can throw or catch events with a specified trigger type (signal, message etc) after a process instance has started. In-flow Catch signals halt the flow until event is triggered. Task boundary signals affect the task they are attached to if the event is triggered whislt the task is in progress.
- End events are optional, and indicate the end of a flow or branch.

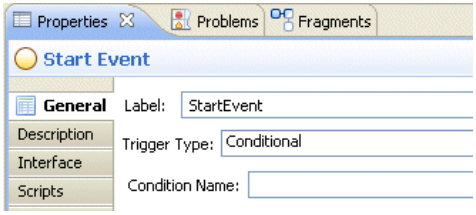
Start Events

Start events can be used to indicate the start of a process (they are optional). Different types of start events can control how a process is started (or triggered). All start events are "catch" events:

- **None** There is no specific trigger to start the Process.
- **Message** The process is started upon receipt of a message from an external source. This can be implemented using a web service using the Solution Design capability (see Events on page 58).
- **Timer** The process is started at a specific date/time or at a regular interval (time cycle):



- **Conditional** The Process is started by the evaluation of a condition:



- **Multiple** There can be several possible triggers for the start of the process, which are specified in the Properties view of the start event.
- **Signal** The start of the process is triggered by the receipt of a signal.

Intermediate Events

Intermediate events can used as follows:

- Intermediate events can be used "in-flow" that is, between two other activities.
- Most catch type intermediate events can be attached to task boundary to catch a triggered event only whilst that task is in-progress.
- You can define an event handler by starting a flow with an intermediate event. See Event Handlers on page 67 (trigger type support is destination specific).

An exception to this is described in "Using Message Event Handlers in Business Processes" in *TIBCO Business Studio Modeling Guide* .

You can use the following types of Intermediate events:

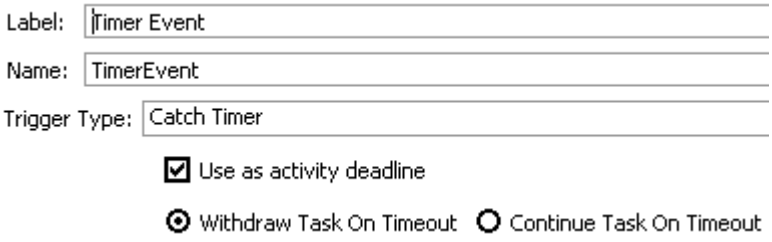
BPMN imposes some restrictions on the placement of intermediate events. For example:

- Intermediate events of type **None** and **Link** cannot be placed on the boundary of a task.
- Intermediate events of type **Cancel** and **Multiple** cannot be placed in sequence flow.

TIBCO Business Studio allows you to place any type of intermediate event on the boundary of a task or in sequence flow; however any invalid constructions are reported in the Problems view.

- **None** Indicates an unspecified change in the process.
- **Throw/Catch Message** Either stops the flow of the process pending the receipt of a message (catch), or sends a message and resumes flow (throw). This can be implemented using a web service using the Solution Design capability (see "Implementing Message Events" in *TIBCO Business Studio Modeling Guide*).
- **Timer** The event is triggered at a specific date/time or at a regular interval (time cycle). When placed on the boundary of a task, a timer event defines a deadline for the task. In the Properties view for the event, **Use as activity deadline** is preselected (this is BPM-destination specific). If more than one timer event is attached to a task, only one of them can be selected as the timer deadline. If there is a cancelling timer then it must be selected as the deadline.

There are two options you can select from to decide how the task is treated if the event times out (**Withdraw Task on Timeout** and **Continue Task on Timeout**).



The timer deadline uses the appropriate system calendar (specific to a destination environment) to calculate when the deadline will expire. For more information about the system calendar, see the destination-specific implementation guide.

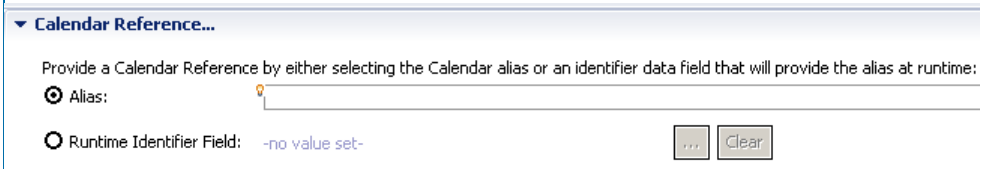
(BPM-destination specific)

— At runtime the timeout period is calculated using the **calcDeadline** API operation described in the "BusinessDeadlineService" chapter in the *TIBCO ActiveMatrix BPM Developer's Guide*. Note that if you specify a date without a time element (no hours or smaller units) then the period is assumed to be in working days.

— The **calendarLookAhead** property in the **dac.properties** file specifies how far ahead the algorithm should look when calculating the timeout. If there is not enough working time available to complete the task in the period defined by **calendarLookAhead**, an error is returned. The property defaults to a value of one month, but you should ensure that it is set to a large enough value to give correct results for your calculations.

See the section "Configuring TIBCO ActiveMatrix Calendar Properties" in *TIBCO ActiveMatrix BPM Administration* for more details.

— You can choose to enter a reference to an alternative calendar to the system calendar from the **Resource** tab. Under Calendar Reference, enter a Calendar alias in text (content assist will provide previously used aliases in the workspace) or a reference to a process data field that will provide the Calendar alias at runtime.



— The information you enter is subsequently used by the Calendar gadget in Openspace. See the *TIBCO Openspace User Guide* and the *TIBCO ActiveMatrix BPM - BPM Developer's Guide* for more information on using alternative calendars.

— This information can refer either to a base or to an overlay calendar. However, note that an overlay calendar must be applied to a base calendar, and if it cannot be applied to any other base calendar, the default System calendar will be used. This may not always give the desired results, particularly if the server holding the System calendar is in a different country or timezone. See "Working with Calendars" in the *TIBCO Active Matrix BPM - BPM Developer's Guide* for information on how calendars are applied.

- **Conditional** The event is triggered based on the evaluation of a condition.

- **Throw/Catch Link** Indicates a connection from one or more throw link intermediate events to a catch link intermediate event in the same parent process. This can be thought of as a "go to" or "off page connector" that you can use to break up a process for better legibility.
- **Throw/Catch Signal** Broadcasts or catches a signal. A throw signal event is assigned a default signal name (**signal*n***).
- **Throw/Catch Multiple** Indicates that there can be several possible triggers for the event.
- **Catch Error** Attached to a task boundary to end a sub-process with an error. Either catches the specified error, or catches any error if no specific error is specified.
- **Throw/Catch Compensation** Used to process compensating activities for previously executed tasks:
  - If located in sequence flow of the process, this event throws a call for compensation.
  - If attached to the boundary of an activity, this event catches a named compensation call.
- **Catch Cancel** This type of intermediate event is used on the boundary of a transaction sub-process. It is triggered if a cancel end event is reached within the transaction sub-process or if a transaction protocol “cancel” message is received while the transaction is being performed.

End Events

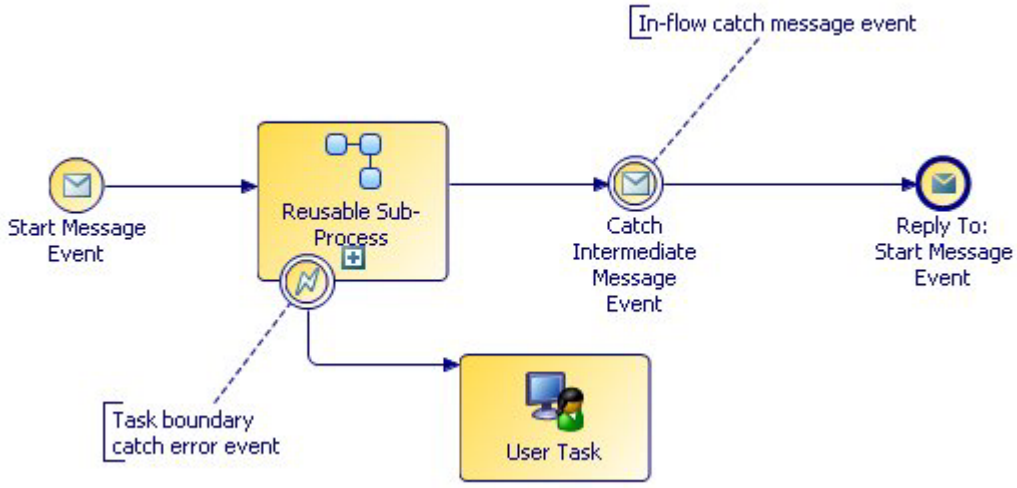
An end event indicates when the process has completed. They are optional, however if a process contains a start event, it must contain an end event. End events have different types that indicate different results upon completion of the process. All end events are "throw" events:

- **None** There is no specific end result to the process.
- **Message** Indicates that a message is sent at the end of the process. The message can either be a response to the start message, or a different message. This can be implemented using a web service using the Solution Design capability (see "Implementing Message Events" in *TIBCO Business Studio Modeling Guide*).
- **Signal** Indicates that a signal is broadcast at the end of the process. A signal end event broadcasts a default signal name (**signal*n***).
- **Multiple** Indicates that there is more than one result that will occur when the process ends.
- **Error** Ends all activities in the process immediately without compensation or events, and appears in the Event log as a failed process instance.
- **Compensation** Indicates that a compensation is necessary. For more information, see the BPMN specification at http://www.bpmn.org.
- **Cancel** Used within a transaction sub-process to trigger a cancel intermediate event attached to the sub-process boundary.
- **Terminate** Ends all activities in the process immediately without compensation or events.

Throw and Catch Events

An event that is located in a sequence flow can "throw" an event that can be "caught" by a catch event. Additionally, throw events are distinguished from catch events on a diagram by having their symbol colored in:

Catch intermediate events can be placed in flow, or on a task border. An in-flow catch event halts the flow until the event is triggered. A catch event attached to a task border usually cancels the task when the event is triggered.



Triggers and Results

An event *trigger* defines the cause for the event (for example, an error elsewhere in the Process). An end event *result* indicates the consequence of a sequence flow ending (for example, the sending of a message).

Triggers and results can either be "thrown" by events, or "caught" by other events as described.

Palette Summary

The following table summarizes the BPMN events that are available to you on the palette in TIBCO Business Studio:

Table 2 Summary of Event Types

Event Type	Start	Intermediate *catch*	Intermediate *throw*	End	Description
None					No specific trigger for the event (for example to start the process as a sub-process).
Message					Message to do one of the following: start the process, wait to resume the process, resume the process, signify the end of the process.
Timer					Event is triggered at a specific date/date or regular time interval (time cycle).
Error					Ends a sub-process with an error. On a task boundary, either catches the specified error or any error if no error code is specified.
Cancel					Ends a transactional sub-process. On a task boundary, catches the cancel event thrown from within the sub-process.
Compensation					Either throws or catches a call for compensation. Used to process compensating activities for previously executed tasks.
Conditional					Triggered based on the evaluation of conditions.
Link					Creates a "go to" or "off page connector" to break up a process for better legibility.
Signal					Broadcasts or catches signals.
Multiple					Indicates that one of several possible triggers are to be thrown or caught.
Terminate					Ends the process and all activities within without compensation or error handling.

Event Handlers

Event Handlers are supported in Business processes and Pageflow processes. You can use event handlers to execute a flow that is separate from the main flow of the process (for instance to update process data used by the main flow).

Event handlers can be triggered **zero** or more times during the life of a process instance.

The flow from an event handler cannot be joined to the main flow (the flow from a start activity) or other event handler flows.

While an event handler allows you to do something multiple times during a process, it does not have to happen for the process to complete.

An event handler is a catch intermediate event with **no** incoming flow. Event handlers with no specific trigger type will normally be triggered through a destination-specific API or utility. See the *BPM Developer's Guide* for more information.

Data Fields and Parameters

Data fields identify the inputs and outputs of an activity. Parameters are input to or output from a source external to the process.

For example:

- Data fields: an activity called "Process Student Course Request" could require a form with the list of courses the student wants to take as input. The availability is checked and a form that lists the courses they are enrolled in is output.
- Parameters: parameters are passed from a process to a sub-process.

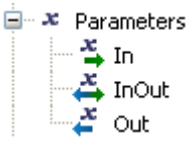
Parameters can only be created at the process level. Data fields can be created:

- at the package level, where they can be shared amongst processes defined in that package.
- at the process level, where they can only be used by the parent process.
- at the activity level, where they can only be used by the parent activity. (Activity-level data fields are only supported for certain activity types in certain destination environments. See the appropriate implementation guide for more information.).

You can change a process-level data field to a parameter by right-clicking it, and selecting **Convert Data Field to Parameter**. The default mode for converted parameters is **In/Out**, meaning that they can be specified as either input or output.

Data fields and parameters share the same basic types (Integer, Text, and so on), however they have different properties:

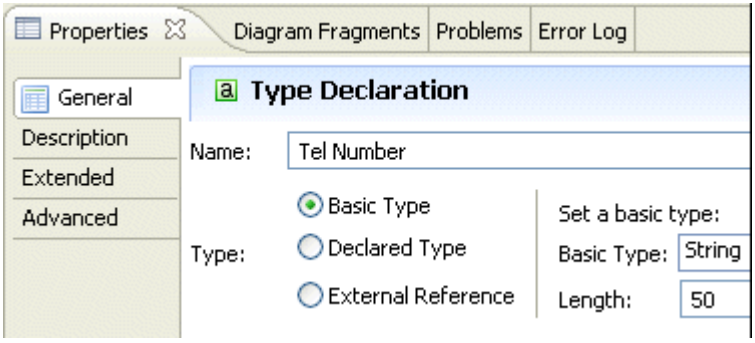
- Parameters can be specified as mandatory (they must be present at runtime).
- Both data fields and parameters can be specified as read only. For data fields, this means that they can only be assigned by their initial value setting. For formal parameters, it means that once input to the process they cannot be re-assigned (for example, in scripts or user tasks).
- Data fields and parameters can be an array of basic types (for example an array of Text).
- Parameters can also be specified as input, output or both by selecting the Mode (**In**, **Out**, or **In/Out**). The mode is indicated by the icon next to the parameter:



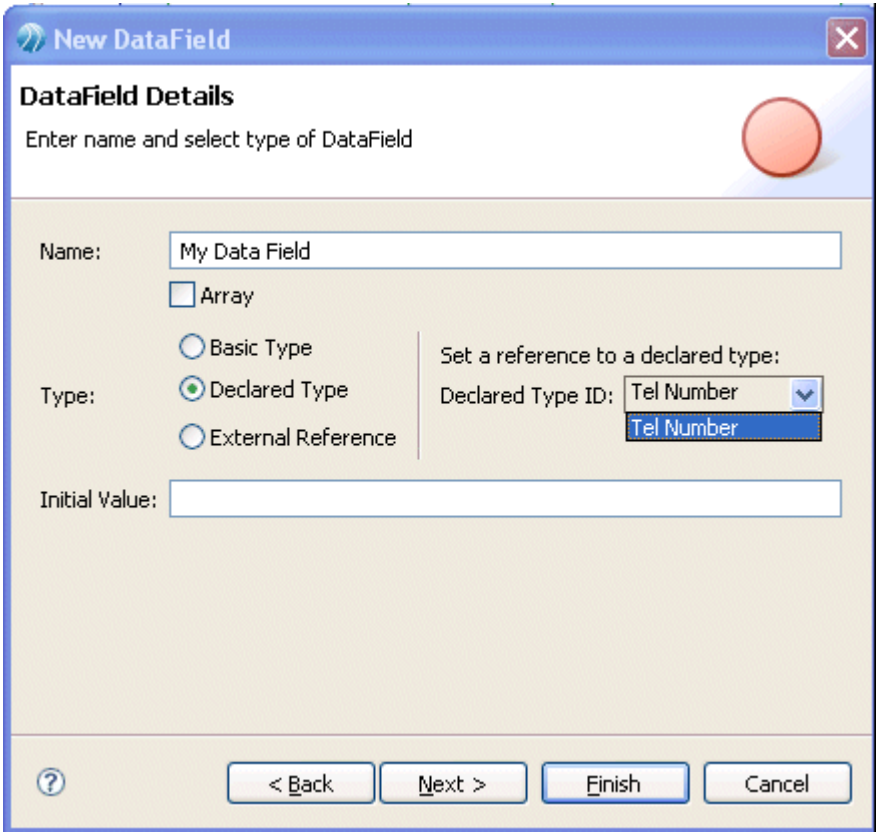
Declared Types

Declared types are used if you want to re-use a definition either when creating a data field or parameter.

For example, you could create a declared type that is text that represents a telephone number:

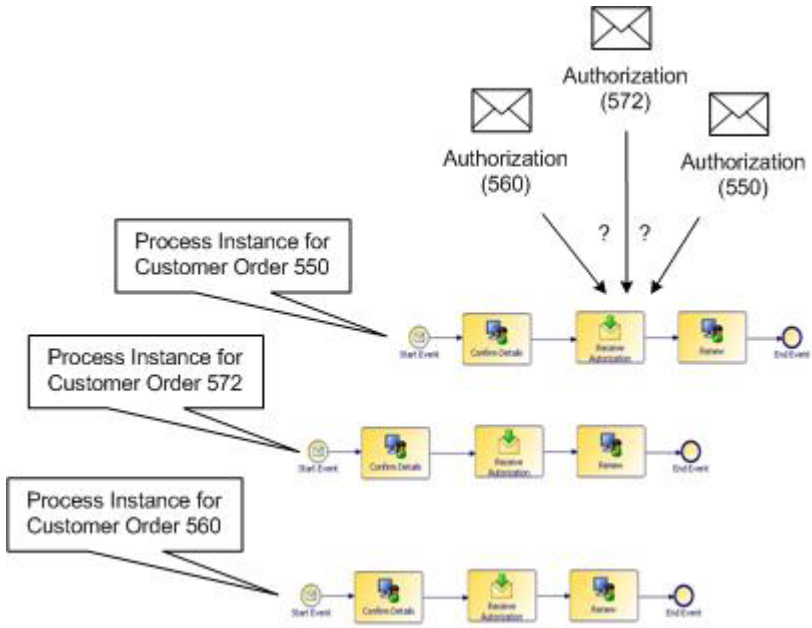


This declared type is then available for use in defining data fields or parameters. For example:



Correlation Data

Correlation data must be used to ensure that each incoming message is received by the process instance to which it applies. The incoming data to a receive task or catch message event is compared to the correlation data in existing process instances to determine whether it applies to that process instance. You must initialize the correlation data with a value, either on the start message event, or subsequently (for example in a script task).



In a process, a catch message event or a receive task waits for an incoming message to arrive. In the runtime environment there may be many instances of the process (each with different data), and many incoming messages (each with different data).

To illustrate how using correlation data works, consider the following example:

- The process begins with a start message event (the process is started upon receipt of a one-way message from an external web service). The business analyst decides whether elements within the incoming data from the web service can be used to

uniquely identify instances of this process. If the incoming data can be used to identify the process (later on in the process when there are incoming messages), the business analyst creates a correlation data field (for example, **OrderRef**), and associates it with the start event on the **Interface** tab.

- After the user task, the receive task (**Receive Authorization**) waits for the correct incoming authorization. To do this, the business analyst associates the **OrderRef** correlation data field with the **Receive Authorization** task. When an incoming message is received, the incoming data is compared to the correlation data to make sure that it applies to that process instance.

After creating correlation data fields, and associating them with the relevant events or tasks, the correlation data must be mapped with the incoming data. This is the job of the solution designer. For more information, see the appropriate implementation guide.

Destinations and Validation

When you specify a *destination environment* for a Process, you are specifying the intended runtime environment in which the Process will execute. According to the destination environment that you specify, TIBCO Business Studio performs validation on the Process

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..

In previous versions of TIBCO Business Studio, TIBCO recommended disabling in-memory validation by deselecting **Project** > **Build Automatically** as a way of achieving performance gains. Because of performance enhancements with TIBCO Business Studio, this is no longer necessary and is not desirable as it causes problems resolving references.

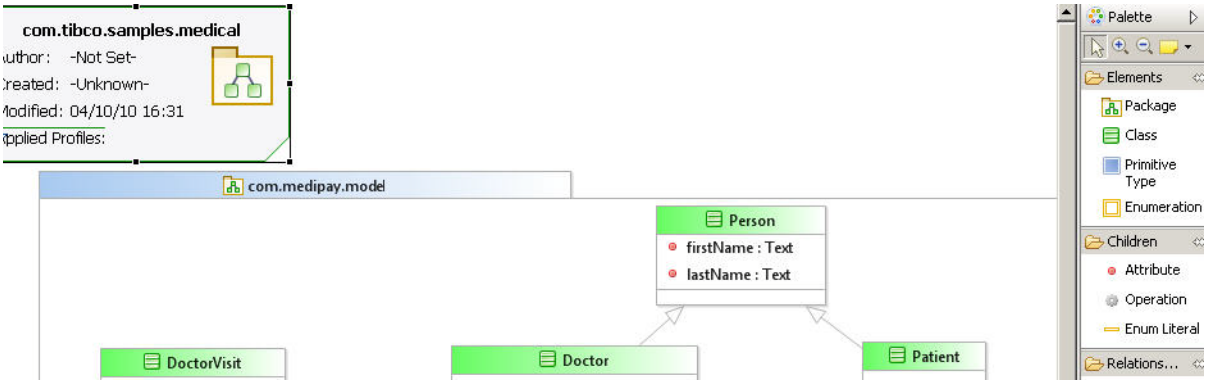
You can create your own destination environments, and the specific "destination components" that make up these destination environments can be customized in the Preferences. To view the current configuration of destination environments, select **Window** > **Preferences**, and select **Destination Environments**. For more information about customizing destination environments, see the destination-specific implementation guide.

BOM Concepts

The Business Object Modeler provided by TIBCO Business Studio allows you to define a vocabulary of core business objects (for example, an Order) and the relationships between the different objects (for example, an Order has several Item Line attributes).

This is useful in the analysis phase of a project where you can use the Business Object Modeler to visualize the data structure that underpins a Process. The data being modeled can be local to the processes, global data, or a combination of both.

A business object model is defined using the Business Object Model Editor, which is a class diagram editor in Unified Modeling Language (UML). An example of the Business Object Model Editor is shown below.



See the *TIBCO Business Studio Modeling User Guide* for more information.

Topics

Business Object Models

A business object model is a set of business terms and relationships specific to your corporate environment (for example, in a financial environment, broker, counterparty, and so on). TIBCO Business Studio provides an Eclipse editor called the Business Object Model Editor to help you construct your business object model. In object-oriented terms, when you create a business object model, you are creating a class diagram using UML.

The advantage of creating or importing a business object model in TIBCO Business Studio is that you can use it:

- for analysis purposes,
- for documentation purposes,
- to incorporate business data in your business processes.

  You can create a Data Field in a Process that corresponds to a Class that you have defined in the business object model. If you create a data field in a process and set the field type by an external reference to a Class in the business object model, the field will have sub-fields that correspond to the attributes defined for that Class. See the chapter "Working with Process Data" in the *TIBCO Business Studio Modeling User's Guide* for more about working with data fields.

UML

Unified Modeling Language (UML) is an Object Management Group (OMG) specification that helps you specify, visualize, and document models of software systems or business systems, including their structure and design.

The Business Object Modeler uses terms and notation similar to UML, so an understanding of UML can be useful.

The latest major version of UML (and the one employed within the Business Object Modeler) is Version 2, commonly referred to as UML2. For more information see http://www.uml.org/.

UML Profiles and Stereotypes

A UML profile enables you to specify stereotypes. Stereotypes provide a method of extending your business object model. You may have a particular domain within your business that you want to include in your business object model. You can have a UML profile that enables you to create new model elements, derived from your existing model elements but that have specific properties for your business domain. The UML profile enables you to tailor the language in your business object model to your specific business domain.

If you are using your UML model in a database for example, you may want to specify which database table a Class should be stored in. You can have a UML profile that contains the database tablenames as stereotypes. Once you have created your UML profile you can apply it to your business object model and then apply those Stereotypes to your Classes as required.

Stereotypes can be applied to all business objects in a business object model, including packages, classes, attributes, primitive types, operations, generalizations and associations.

A stereotype is displayed as a name enclosed by guillemets and placed above the name of another business object, as shown below:

You can apply existing UML profiles that have been created elsewhere to your business object model.

You can apply as many profiles as you like to your business object model.

Refer to the following Eclipse documentation for a description of how to create UML2 profiles and define stereotypes.

http://www.eclipse.org/modeling/mdt/uml2/docs/articles/Introduction_to_UML2_Profiles/article.html

Concept Models

In some previous versions of TIBCO Business Studio, the Business Object Modeler was called the Concept Model Editor. This version of TIBCO Business Studio enables you to create business object models, as well as concept models.

A business object model is more generic than a concept model. A concept model is a specialization of a business object model. It is specialized using a UML Profile that extends the meta-data to provide additional functionality over core UML.

In this version of TIBCO Business Studio you can:

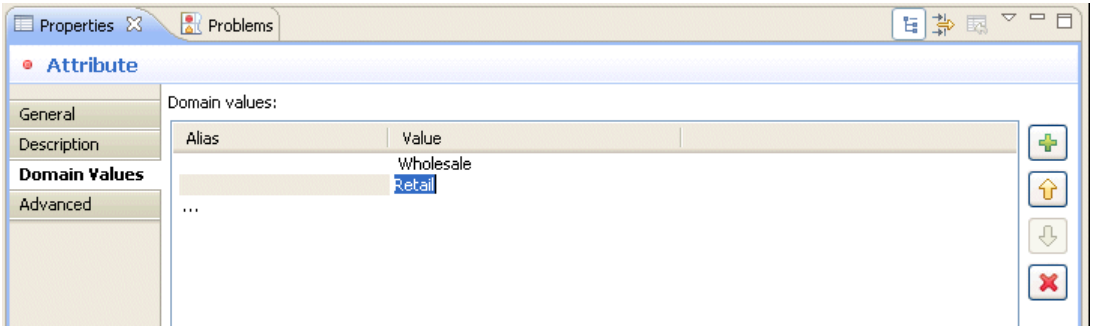
- use concept models that were created in previous versions of TIBCO Business Studio,
- create new concept models.

You may want to create a concept model instead of a business object model if it suits your business requirements.

A concept model is different from a business object model because in a concept model:

- Attributes for Domain Values are available.

  You can specify the permissible Domain Values for an Attribute (the values that they are likely to have in your model). For example:



  In this example, the Customer Class has a Type Attribute which can have the Domain Values "Wholesale" or "Retail".

- There are Concepts instead of Classes. Concepts are different from Classes because they can have domain values, whereas Classes cannot.
- Operations are not available because a concept model is intended for modeling data only. It is not intended to model how data behaves.
- Aggregation and composition are not available.
- You cannot apply stereotypes.
- You cannot apply additional UML Profiles to a concept model.

Diagram Nodes

Business object model Diagram Nodes consist of Packages, Classes, Association Class, Primitive Types, Attributes and Operations.

When creating diagram nodes, note that:

- When you create a business object model in the Business Object Modeler, the Business Object Modeler automatically creates a package of the same name in which to place your diagram nodes. Therefore, there is no need to create a package in which to place all your diagram nodes, unless you want to create extra packages in which to group diagram nodes. For example, you might wish to create a package to contain all your Primitive Types.
- The Business Object Modeler uses the following Java programming conventions.
  - Business object names should not use an Invalid Java Identifier.
  - Business object names must not be a Java keyword.
  - Business objects should follow the Java Naming Convention.

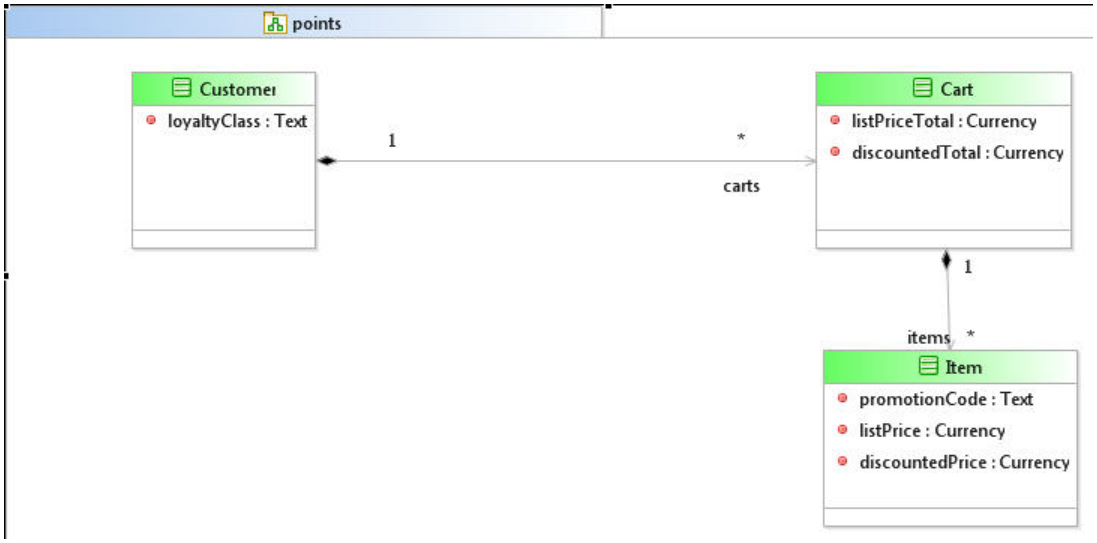
  See http://java.sun.com/docs/codeconv/ for more information about these. These Java Programming Conventions are options under Preferences, so you can choose to disable them if you require. See Setting Diagram Preferences on page 70 for more information.

Warning messages are displayed on the Problems tab if these conventions are not followed.

Warning messages indicating breaches of the Java Naming Convention are displayed only if you are working with the Solution Design capability selected. If you deselect the Solution Design capability, the Business Object Modeler carries out a full revalidation of your model against the Business Analysis capability, and these warnings will no longer be displayed. Similarly if you re-select the Solution Design capability, your model is again revalidated, and any warnings related to the naming conventions will be displayed.

Packages

A Package is a way of grouping related model elements. For example:



A Package can also contain other Packages.

Note that the business object model itself is a package, and any packages that you add are sub-packages contained within it.

Note also that you can open a new editor to edit only the contents of a selected package, for example if the main editor window is too crowded. See Opening a Diagram Editor for a Package on page 46.

Classes

A class is a description of a set of properties that, when grouped together, create a meaningful unit. Classes can be organized in a hierarchical structure.

Example: *Department*, *employee*, *purchase order*, and *inventory item* are all classes.

Primitive Types

A Primitive Type is a data type. Defining a Primitive Type enables you to define your own data types and then specify how data of that type is interpreted. You can specify what values the data can have or any constraints on the data, for example. It enables you to refine your data to your specific business domain. You could create a Primitive Type

called Patient ID and specify that Patient IDs should always be a maximum of 7 characters long, and consist of 2 letters followed by 5 numbers, for example. See Setting Restrictions on Primitive Types and Attributes on page 84 for more information on the restrictions you can set.

All the Primitive Types that you define must be based on the following standard business object model Primitive Types that are available in the business object model Editor. Some types have subtypes. The types are described in the following table.

Primitive Type	Description
Attachment	A binary file.
Boolean	A value of True or False.
Date	A date in the format dd/mm/yy.
DateTime	A date and time in the format dd/mm/yy hh:mm.
Decimal	Any number, positive or negative.
Decimal: Floating point	A subtype of Decimal, where the number of digits and of decimal places are not defined.
Decimal: Fixed point	A subtype of Decimal. A floating point number where the maximum number of digits and the number of decimal places are defined.
Duration	A string denoting a duration.
ID	A string denoting a unique ID.
Integer	Any whole number, positive or negative.
Integer: Signed Integer	A subtype of Integer, where the maximum number of digits is undefined.
Integer: Fixed length	A subtype of Integer, where the maximum number of digits is defined.
Object	
Object: xsd:Any	A subtype specific for representing the XML Schema type "any".
Text	Any characters can be entered up to the length you specify.
Time	A time in the format hh:mm (24 hour clock).

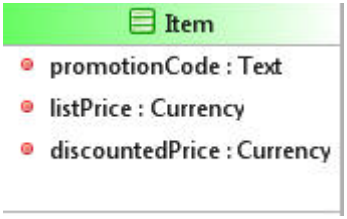
Primitive Type	Description
URI	A Uniform Resource Identifier.

When you create a Primitive Type, it defaults to Text. the type is displayed above the name of the Primitive Type in italics as shown below.

You can change the standard type on which the Primitive Type is based by clicking on the ... button in the **Superclass** field, on the General tab in the Properties View. Additional information required by the Primitive Type is specified on the Advanced tab; see Setting Restrictions on Primitive Types and Attributes on page 84.

Attributes

Attributes describe the information stored or maintained about a Class. For example:



The Item Class has the Attributes promotionCode, listPrice, and discountedPrice.

Attributes, like Primitive Types, must be based on one of the standard business object model Primitive Types that are available in the business object model Editor.

Multiplicity

The **Multiplicity** field on the General tab of the Properties view for an Attribute indicates whether TIBCO Business Studio needs to allow for multiple copies of an Attribute. For example, a purchase order Class could allow for only one po_ number Attribute but multiple line_items Attributes.

The lightbulb icon by the field indicates that content assistance is available. Press **Ctrl+space** to display a list of the possible values for the **Multiplicity** field.

The following table describes the available multiplicity indicators:

Indicator	Meaning
0..1	Zero or 1
1	One
1..*	One or more
0..*	Zero or more
n	Where *n* is greater than 1
0..*n*	Zero to *n* where *n* is greater than 1
1..*n*	One to *n* where *n* is greater than 1

Setting Restrictions on Primitive Types and Attributes

You can specify restrictions for Primitive Types and Attributes. To do this:

1. In the **Properties View** for the Primitive Type or Attribute, select the **Advanced** tab.
2. Expand the **Restrictions**.

The restrictions you can specify depend on the type (including subtype where appropriate) of your Attribute or Primitive Type. The following table describes the restrictions you can set for each type.

Type	Property	Description
Boolean	Default Value:	A default value of either True or False.
Date	Default Value:	Enables you to specify a value that should be automatically supplied for the type.
DateTime	Default Value:	Enables a value to be automatically supplied.
Decimal	Decimal Places:	Enables you to specify how many numbers there should be after the decimal point. Does not apply to the Floating Point subtype.

Type	Property	Description
	Default Value:	Enables you to specify a value that should be automatically supplied for the type.
	Lower Limit:	Enables you to specify the smallest number the type should allow.
	Lower Limit inclusive:	Enables you to specify a lower limit within a range of numbers.
	Number Length:	Enables you to specify the maximum length of the number. Does not apply to the Floating Point subtype.
	Sub Type:	Either Floating Point (the default) or Fixed Point.
	Upper Limit:	Enables you to specify the highest number the type should allow.
	Upper Limit Inclusive:	Enables you to specify a higher limit within a range of numbers.
Integer	Default Value:	Enables you to specify a value that should be automatically supplied for the type.
	Lower Limit:	Enables you to specify the smallest number the type should allow.
	Number Length:	Enables you to specify the maximum length of the number. Does not apply to the Signed Integer subtype.
	Sub Type:	Either Signed Integer (the default) or Fixed Length.
	Upper Limit:	Enables you to specify the highest number the type should allow.
Object	Sub Type:	xsd:Any

Type	Property	Description
Text	Default Value:	Enables you to specify a value that should be automatically supplied for the type.
	Maximum Length:	The maximum length of the value allowed.
	Pattern:	Enables you to create a pattern.
Time	Default Value:	Enables you to specify a value that should be automatically supplied for the type.
Attachment **Duration** **ID** **URI**	None	There are no restrictions for these types.

Operations

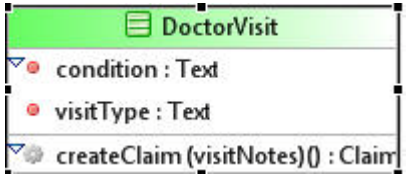
An Operation allows you to specify the function of a Class. Defining an Operation enables you to request a Class to perform that function. If you have a Class called **policy** for example, you could create an Operation called **createClaim** to create a claim against this policy.

Operations are not supported in BPM. If you add an operation in Business Object Modeler when BPM is your specified destination, a warning message is generated. Operations may still be useful for analysis purposes, but they do not map to anything in BPM at runtime.

You can specify arguments and argument types for Operations. Arguments often add extra data that the operation needs. For example, the **createClaim** operation could specify the Doctor's visit notes as a parameter.

An Operation can also return a value after it has finished. You can show the value it returns and the value's type. For example, the **createClaim** operation could return the **Claim** once it has completed.

Each Class can have a number of Operations. Operations that are defined for a Class are displayed in the Operations Compartment of the Class, below a line that separates them from the attributes, as shown below.



The example above shows the **createClaim** operation that has a parameter of **VisitNotes** and whose return value is the **Claim**.

Enumerations

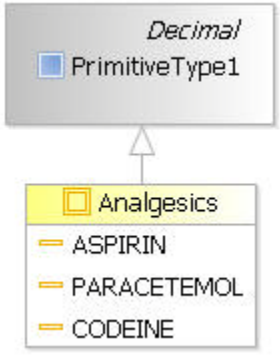
An Enumeration is a data type that can contain a list of values. An Enumeration contains a set of named identifiers, called Enumeration Literals, that represent the values of the enumeration.

An enumeration can be a generalization of Primitive Type Object if it is not a String enumeration.

- See Primitive Types on page 81.
- See Generalizations on page 90.

You must set up the generalization, and you must have created the Primitive Type object to generalize from.

In the example below, a Primitive Type of the Decimal data type has been created. There is also an Enumeration called **Analgesics**, containing enumeration literals. You create a generalization from the Enumeration to the Primitive type. In this example, this would create an Analgesics enumeration with decimal literals.



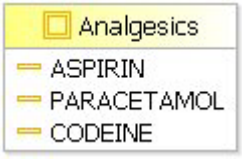
Generalizations of the following data types are **not** supported: Attachment, Boolean, Duration, ID, Object, URI.

An Enumeration extending another Enumeration can contain only a subset of the enumeration literals defined in the parent Enumeration(s): for example, an Enumeration "Shape" (RED,BLUE) extending Enumeration "Color" (RED,BLUE, SILVER,GREEN). Shape is treated as a restriction in Business Data Services rather than an extension. Therefore, if at runtime you pass a value "SILVER" for the Shape enumeration, a failure occurs.

Enumeration Literals

Enumerations contain Enumeration Literals in a similar way to the way that Classes contain Attributes. Each Enumeration Literal is one of the list of values that make up the Enumeration.

For example, an Enumeration called **Analgesics** might contain literals listing the different analgesics that might be prescribed.



Enumeration literals require values to be set. The format of the value is determined by the data type of which the enumeration is a generalization. Ranges of values are **not** supported.

In the example in Enumerations on page 87, **Analgesics** is a generalization of the Decimal data type.

So the Enumeration literal **CODEINE** requires a value in a decimal format:

Relationships

In TIBCO Business Studio, relationships are used to show the relationships between objects and consist of Generalizations, Aggregation, Composition, and Aggregation.

The terms described in this section are identical in meaning to the same terms described in UML.

Not all these types of relationship are supported for all destination environments. TIBCO Business Studio allows you to create all four types of relationship listed here, and they may all be useful for analysis purposes. However, an error will be generated if you try to deploy a business object model containing an Association or an Aggregation to BPM.

Relationships in the business object model can be grouped into two types:

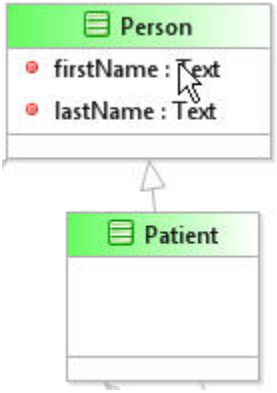
- Generalizations
- Aggregation, Composition, and Aggregation. Composition and Aggregation are both more specific types of the general relationship type Association. However note that Composition is the only one of these supported in BPM.

  These types can be further classified with regards to:

  — End ownership (Composition and Aggregation only).

  — Navigability. These relationships may be unidirectional or bidirectional.

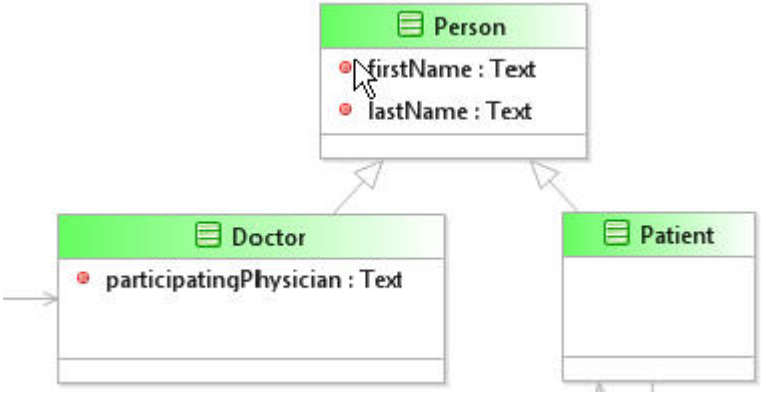
All relationships, except Generalizations, can have various labels when they are created in a business object model. A label name for the relationship is not displayed by default but can be added. They also display the names of the Classes and the multiplicity allowed for the Classes. This is shown below:



You can choose to hide these labels by clicking  or you can delete the ones that you do not require from the actual Model. If you delete them from the Model, you are not actually deleting the labels, you are hiding them. You can display them again by clicking  .

Generalizations

The generalization relationship indicates that one of the two related Classes is a more general form of the other (an "is a" relationship). For example, a doctor "is a" person, and a patient "is a" person. As such a doctor inherits the attributes and relationships of the general Class (person). In the Business Object Modeler, a Generalization is represented by a hollow triangle connected to the general form. For example:



In object-oriented terminology, this means that Doctor and Patient are Derived Classes of the Super Class Person. This relationship is also known as inheritance. In other words, the Doctor and Patient Classes inherit the attributes of the general Class Person.

Associations

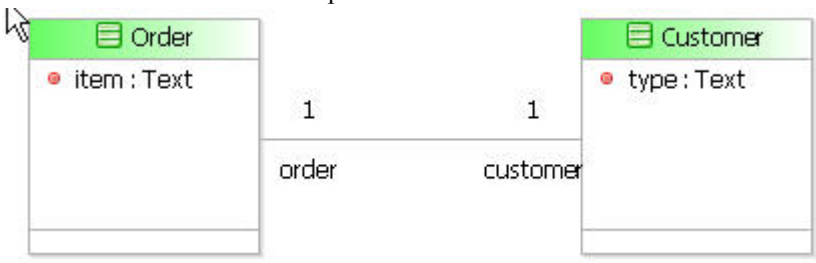
Note that Composition and Aggregation are also types of Association.

The type of connection specifically referred to in the Business Object Modeler palette as an Association is not supported in BPM.

An Association shows a relationship between two Classes. The relationship indicates that the Classes need to share data and how one Class can access another. For example, an Association between a **Customer** Class and an **Order** Class shows that a Customer has one or more orders. If you have an order, you can locate the customer who placed that order.

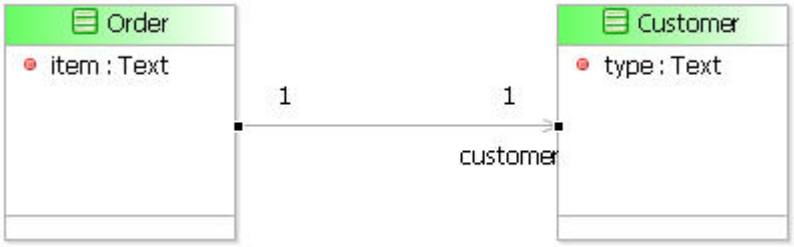
You can create Associations between two classes in the same package, or between classes in different packages.

Associations between Classes affect the Attributes of those Classes. If there is a bi-directional Association between two Classes then each of those Classes will acquire the other Class as an Attribute. An example of a bi-directional Association is shown below:



In this example, the **Customer** Class acquires an attribute called **Order** and the **Order** Class acquires an Attribute called **Customer**. In other words, from an Order you can find out about a Customer and from a Customer you can find out about an Order.

An Association can also be from one Class to another as follows:



In a one way Association, only the source Class acquires the attribute of the target Class. In the example above therefore, the **Order** Class inherits the **Customer** Class as an Attribute but the **Customer** Class does not inherit the **Order** Class as an Attribute. In other words, from an order you can find out about a customer but you cannot find out about an order from a Customer.

Composition

Composition is a specific type of Association used when a Class is a collection or container of other Classes, but the relationship is such that if the Class that functions as the container is destroyed, the Classes representing the contents are destroyed as well.

In the Business Object Modeler, Composition is represented as a filled diamond shape connected to the containing Class.

Compositions can be bi-directional, or can be navigable only in one direction. When you create a composition in Business Object Modeler, by default its navigability is from the source class to the target class.

Bi-directional compositions are not supported in BPM.

Aggregation

Note that Aggregations are not supported in BPM.

Aggregation is a more specific type of Association. It is used when a Class is a collection or container of other Classes, but the relationship is such that if the Class that functions as the container is destroyed, the Classes representing the contents are not.

In the Business Object Modeler, Aggregation is represented as a clear diamond shape connected to the containing Class.

Aggregations can be bi-directional, or can be navigable in only one direction. When you create an aggregation in Business Object Modeler, by default its navigability is from the source class to the target class.

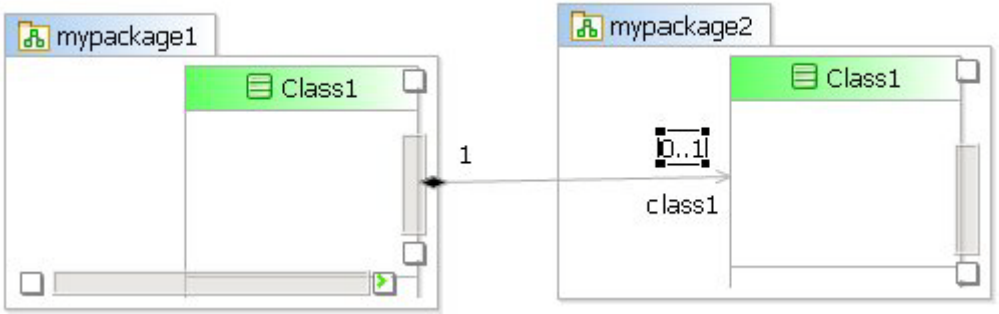
Multiplicity

Connections such as Association, Aggregation, and Composition have multiplicity at each end of the Connection, indicating how many of each Class participates in the relationship. The following table describes the available multiplicity indicators:

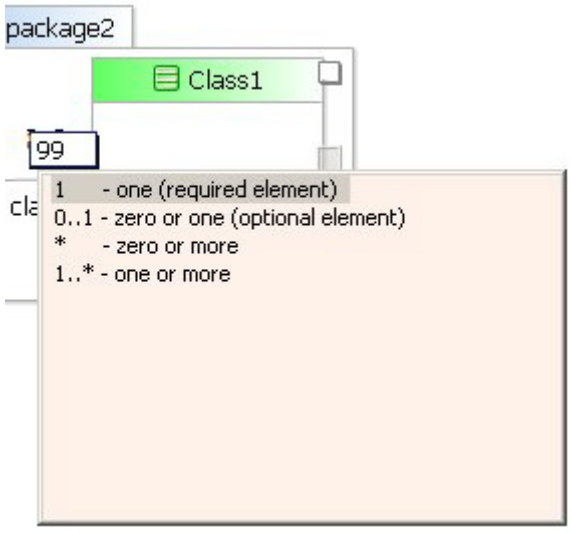
Indicator	Meaning
0..1	Zero or 1
1	One
1..*	One or more
0..*	Zero or more
n	Where *n* is greater than 1
0..*n*	Zero to *n* where *n* is greater than 1
1..*n*	One to *n* where *n* is greater than 1

To specify the multiplicity for a connection, you can:

- Select the connection, and specify the multiplicity in the **Source Role Multiplicity** and **Target Role Multiplicity** fields on the **Advanced** tab; or
- Select the multiplicity value displayed in the editor (as shown in the following diagram) and specify the multiplicity in the **Multiplicity** field on the **General** tab; or



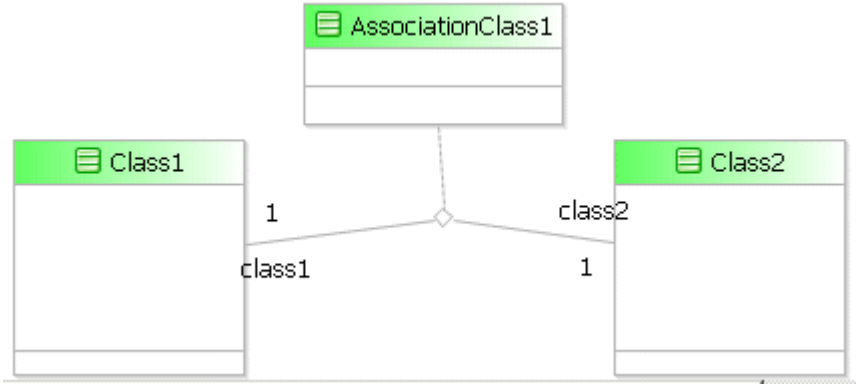
- Select the multiplicity value in the editor and enter the value directly, as shown in the following illustration.



You can select an indicator from the content assistance dialog, or type it in. Where an indicator includes *n*, that represents any integer, and you must type it in.

Association Class

An Association Class is a particular type of class that specifies information about the relationship between two other classes. It is a special type of connection that can exist between classes. An association class has no meaning without the classes with which it is associated.



An Association Class is not selected from the Business Object Modeler palette like other diagram nodes and connections.

Creating an Association Class

1. Create the two classes between which the connection exists. These are shown as **Class1** and **Class2** in the illustration above.
2. Create an Association, Aggregation or Composition connection, as appropriate, between those two classes.
3. Right-click on the connection, and select **Refactor To AssociationClass**. The existing connection is replaced by an Association Class as shown in the illustration.

Organization Model Concepts

The Organization Modeler provided as a part of TIBCO Business Studio allows you to define the organizational structure of your enterprise and the relationships between the different components (for example, organization units and positions) within your organization.

Organization models are useful both:

- In the analysis phase of a project, when you are defining the business processes; and
- If you are using BPM as your deployment destination, at runtime.

See the *TIBCO Business Studio Modeling User Guide* for more information.

Topics

The Organization Model as an Analysis Tool

In the analysis phase of a project you can use Organization Modeler to visualize the organizational structure that underpins a Process.

The structure of an organization is a fundamental aspect of how the organization works.

- It shows how people are organized to achieve the objectives of the business.
- It models the relationships between enterprise systems relevant to the organization.
- It models the relationships between the different departments that describe the day to day operation of the organization.
- It determines how the work that arises from the business processes modeled in TIBCO Business Studio is allocated between different departments and positions within your organization.
- It identifies both concrete resources - people and buildings, for instance - and abstract resources such as roles.

The structure of an organization is a key aspect in the operation of information systems like human resources, payroll and accounting and business process/workflow systems. These systems require a consistent view of the organization to operate efficiently.

However, maintaining a consistent view of the organization is difficult for two reasons:

- Modern enterprises are often spread across different locations and have relationships with extended enterprises. People work in cross-functional teams which may be spread across different locations and enterprises. This makes it hard to identify resources when allocating work within the organization.
- Organizations no longer consist of one global scheme. Organizations are split geographically, by product or by markets and these co-exist within one corporate entity.

Organization Modeler allows you to maintain a model of your enterprise's organization structure in TIBCO Business Studio. It consists of elements that represent the organization's entities, their attributes and the relationships between them.

Organization Modeler does not produce an organization chart; it does not identify named individuals. But it enables you to model your organization abstractly. Managers need to be able to develop robust models of their organization so that this information can be shared by people and systems.

Another way to produce an organization model in TIBCO Business Studio is by refactoring a process to produce an organization. For more details on this, see the *TIBCO Business Studio Modeling User's Guide*.

Deploying the Organization Model

To be used at runtime an organization model must be deployed to a BPM server. In BPM an organization model is seen as part of an application. The application consists of a business process and any supporting material, which can include an organization model. One organization model can be used by multiple applications; your business might have different applications for different business functions, but all of them would need to reference a model of the same organization. See the implementation guide for your destination environment for details of how to deploy applications.

When you deploy an organization model, any Resources that you have defined are not deployed (with the exception of the Human Resource Type, which must exist and is always deployed). All other parts of the organization model as defined in TIBCO Business Studio are deployed**.** See Resources on page 19 for further details.

The Organization Model at Runtime

At runtime in BPM, how an end user's position is defined in the organization model can be used to determine what type of work is presented to them. Customized role-based clients can offer work to users depending on the Position they hold, the Capabilities or Privileges attributed to them, or both. For example, a user with an 'LDAP Administration' privilege could be offered all and only LDAP work.

Elements of Organization Models

Using the Organization Modeler enables you to create a robust model of your organization within TIBCO Business Studio. It enables you to create the following elements that will make up your Organization Model:

- **Organizations** represent both the organization you are modeling and any other enterprises that your organization may have a relationship with. For example, there may be a company your organization has outsourced part of its operation to.
- **Organization Units** represent sub-divisions of an Organization. They are collections of positions which are associated together because they fulfil a business purpose within the organization. For example, an organization unit can be a department, project or location.
- **Positions** represent a set of responsibilities for a job. Positions are created within Organization Units. For example, an Administrator in the Finance Department has different responsibilities from an Administrator in the Human Resources Department.
- **Groups** represent job types within your organization; for example, Chef, Salesman, Doctor and Pharmacist. This is useful for example, if you want to allocate work to a group of people with a specific set of skills. Groups are equivalent to Roles in the

Process Modeler - see the *TIBCO Business Studio Process Modeling User's Guide* for details.

- **Capabilities** can be applied to Positions and to Groups. They represent the skills within your organization; for example, ability to speak Spanish or customer care training.
- **Privileges** represent the authority that Groups, Positions and Organization Units can have within your organization. They can have Qualifiers which indicate the level of the privilege. For example, an Approval privilege might have one qualifier to sign off expenses up to $500, or a higher level of qualifier for approval of budgets up to $10,000.
- **Locations** represent the physical locations that are used by your organization.
- **Resources** are used to specify items such as people, equipment or buildings.
- **Organization queries** are specified either as strings of text, which is not checked or validated within Organization Modeler; or in Resource Query Language, which is validated.
- **System actions** are actions that users perform at runtime but that need to be authorized. They are not defined within Organization Modeler, but you can associate Privileges with a list of available system actions in order to specify the level of authorization that a user needs to carry out an action.
- **Types,** in the schema, represent typical elements within your Organization. This enables you to use the schema as a template for the different organizational components and ideas that your Organization contains.

Benefits of Organization Models

Creating an Organization Model enables you to provide the following benefits and capabilities:

- take advantage of the capabilities of BPM to distribute work at runtime to those users who are best suited to carry out a particular task, as determined by their position within the organization model or by privileges and capabilities defined within the organization model.
- when you are defining a process, use expressions to define the participant who will carry out a task in terms of Organization Modeler entities.
- model the way people are organized together within the organization. This is useful as it enables you to see how people work together and how work is allocated.
- model both hierarchical relationships within the organization and also the associations that exist between cross-functional teams.

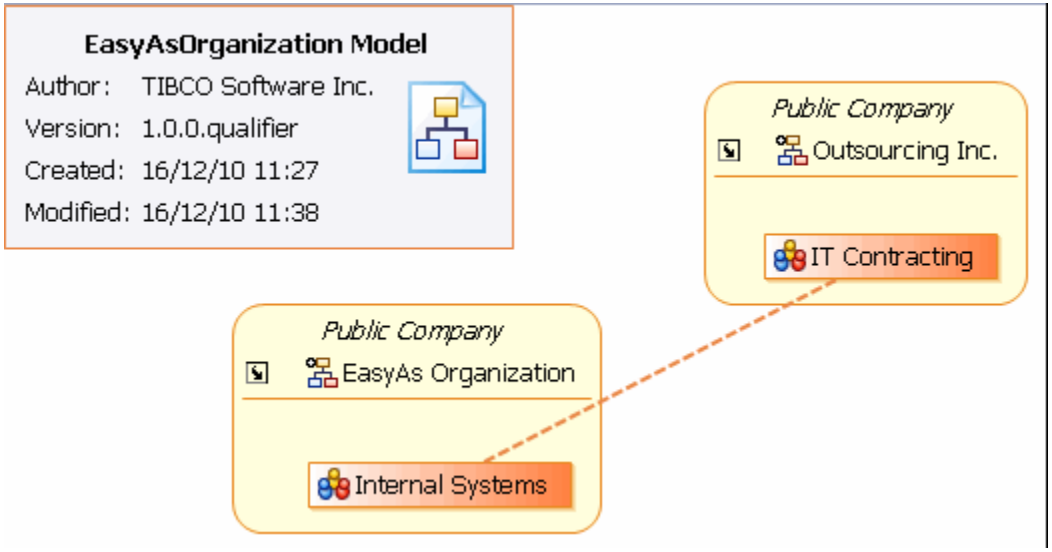
- model virtual or temporary project teams and positions. This is useful for example, as you can create project teams irrespective of their physical locations or job status.
- specify locations so that you can view how your organization is distributed across its various locations.
- specify privileges so that you can view the chains of authority in your organization. Authority is not always hierarchical; it is often given for different purposes. There may be multiple chains of authority within an organization.
- specify the skills (capabilities) you have within your organization so that you can view what skills are available and what skills your organization may need to acquire.
- specify types for certain elements within the organization. Some extended enterprises refer to organizational concepts differently, for example, region vs. district. By specifying types for particular elements, you create a generic schema for your Organization Model that enables it to be exchanged with other systems. Using types also gives you the ability to add custom attributes and custom elements to your organization schema.
- specify attributes for types. You can specify an attribute of telephone number or email address for a resource type, for example, and then values for that attribute can be assigned to each resource of that type.

About the Organization Modeler Diagram Editors

Two graphical editors are provided for producing organization diagrams:

- An **Organization Model Editor** for the root Organization Model
- An **Organization Editor** for the diagrams of each Organization included in the Organization Model

The Organization Model Editor shows a high-level view of the organization or organizations that you have created. The following example shows an Organization Model diagram which includes two organizations, one being your own organization and the second a representation of an external organization with which your organization has dealings, in this case an outsourcing company.

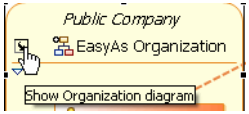


Note that an Association has been created between two organization units, one in each organization, that have a business relationship with each other.

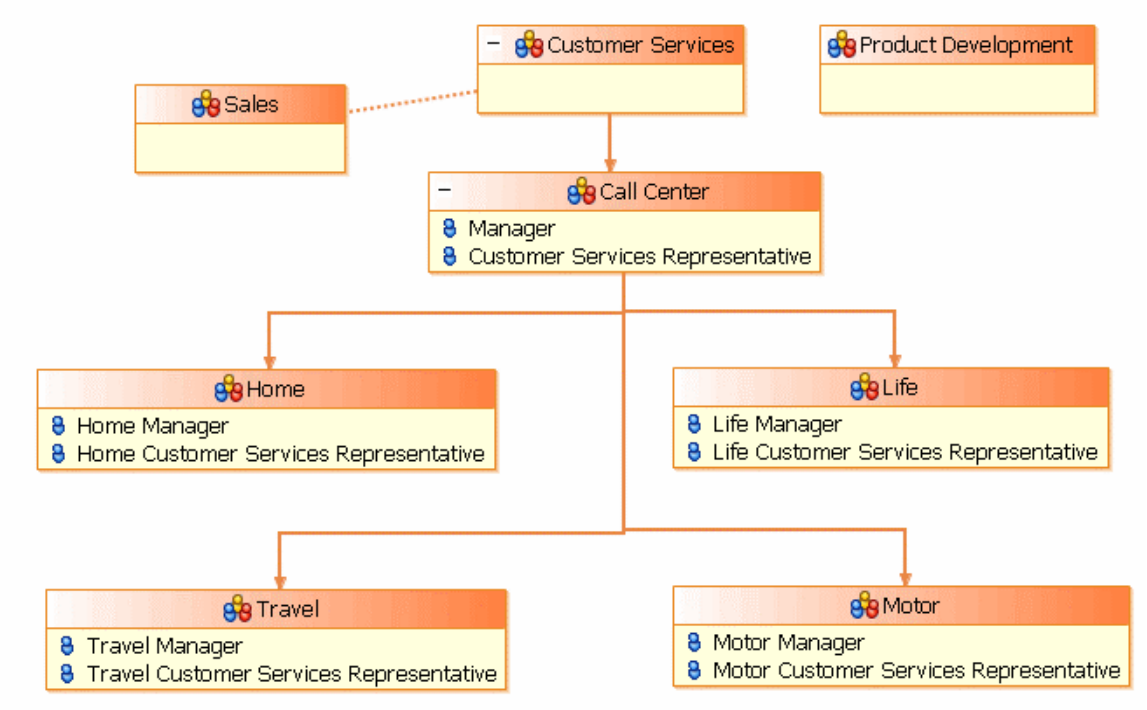
When you create an Organization Model Diagram, a new Organization is automatically created within the Organization Model. The Organization Editor for this default Organization opens automatically after you click **Finish** on the **Create New Organization Model** wizard. To view the parent Organization Model Editor you can navigate to it using the following methods:

- Double-click on an existing Organization Model**.om** file in the Project Explorer,
- Right-click on an existing Organization Model**.om** file in the Project Explorer, and select **Open** or **Open With > Organization Model Diagram Editing**,

- From the Organization Editor displaying any Organization in that Organization Model, click on the shortcut arrow that is displayed on the badge.



The Organization Editor shows a more detailed view of one Organization, including the Organization Units and Positions that it includes, and the relationships between them.

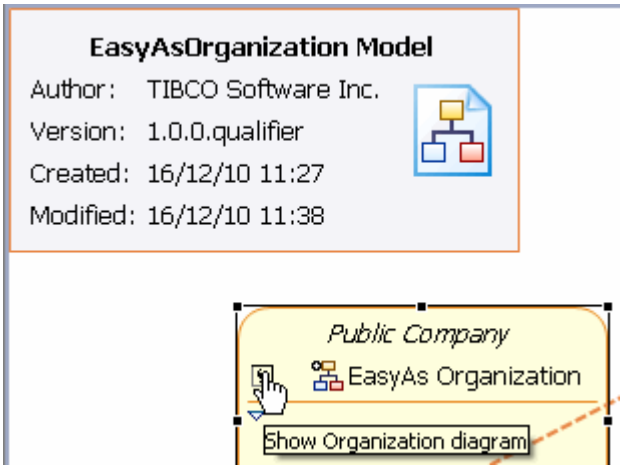


The small icon showing a letter **i** on the title badge of the Organization in this illustration indicates an information message. In this case it is displayed because the Organization has no Type assigned to it.

The Organization Editor opens:

- Automatically for the default Organization that is created when you create a new Organization Model Diagram,
- When you double-click on the representation of an Organization in the Organization Model Editor,
- When you double-click on an Organization in the Project Explorer,

- From the Organization Model Editor, when you click on the shortcut arrow that is displayed on the representation of each Organization within the model.



Organization

An Organization represents the top layer of your Organization Model. Organization Units are contained within Organizations.

You can create many Organizations in the Organization Model. This is useful because you may have an enterprise that has relationships with other enterprises. Part of the operation may have been outsourced to another company, for example. In this situation, you can create an Organization in your Organization Model for each enterprise that your Organization has a relationship with.

An Organization does not necessarily have to represent an organization or enterprise, however. It can represent a department or project. It may make sense for your business model to create a project as an Organization if the project is large enough and it consists of several Organization Units, for example.

An Organization can contain many Organization Units.

There are various properties you can assign to an Organization. You could assign an effective start date and end date. This is useful if the out-source company only has the contract for a set period of time, for example. See Organization Properties on page 81 for more information.

Organization Unit

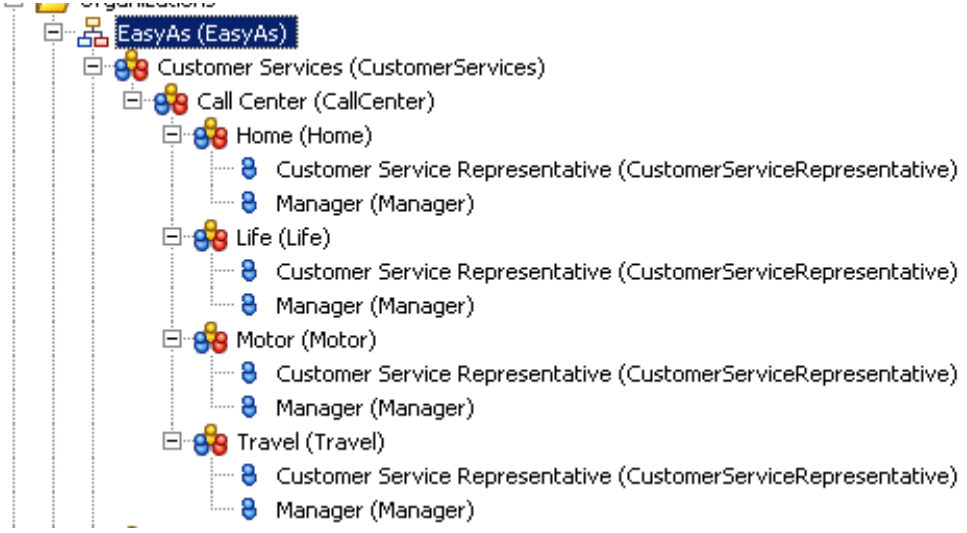
An Organization Unit represents resources that are associated together because they fulfil a business need within the organization. For example, an Organization Unit can be a department, project or location. Different Organization Unit Types are provided in the default Schema, and if it suits your organizational structure you can use them for different levels of organization or other different sorts of unit that might be present.

An Organization Unit is made up of Positions. A Position represents a set of responsibilities for a job of work to be performed in an Organization Unit. An Organization Unit can have many Positions.

Organization Units can be linked by relationships. These relationships can link Organization Units within the same Organization, or in different Organizations in the same Organization Model. (They do not link one Organization to another.)

Relationships can be hierarchical or otherwise. In the Organization Modeler diagram editor they are represented by Hierarchy or Association connections - see Hierarchy and Association on page 107.

An example of a Customer Services Organization Unit that contains Sub Organization Units and Positions is shown below.



You can specify privileges and system actions for an Organization Unit. An example of a privilege might be that the Accounts Department can authorize expenses up to $500.

For each Organization Unit, there are various properties you can assign. For example, you can specify the location of the Organization Unit and how long it should exist for. See Organization Unit Properties on page 82 for more information.

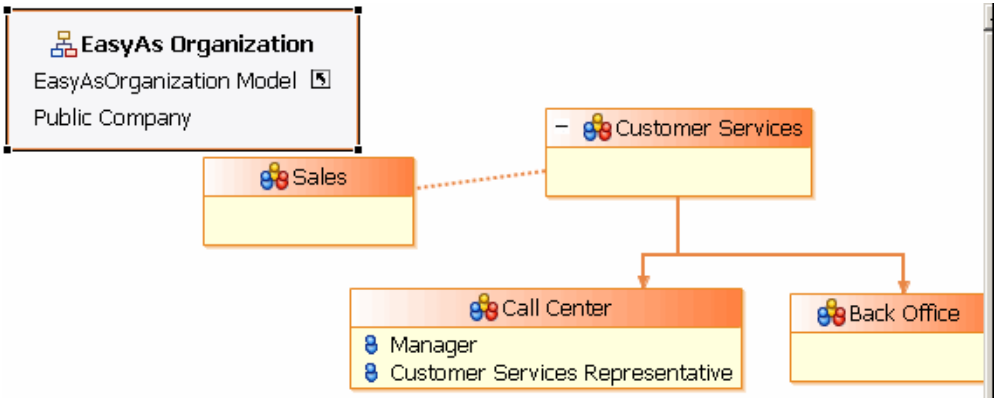
Hierarchy and Association

In the Organization Editor, the relationships between Organization Units are denoted by two similar types of Organization Unit Relationship:

- **Hierarchy**. This indicates a hierarchical relationship, for example between a department and its sub-departments.
- **Association**. This can be used to indicate any sort of non-hierarchical relationship, depending on your requirements. For example, relationships between Organization Units may be based on factors such as resource or work allocation.

In the diagram editor, you can see whether a relationship is hierarchical or not because a Hierarchy is represented by a solid line with an arrowhead, an Association by a dotted line. This difference is controlled by one property of the Organization Unit Relationship, the **IsHierarchical** property - see Hierarchy and Association Properties on page 84. You can edit the value of this property to change a relationship from Hierarchy to Association, or vice versa, simply by checking or unchecking a box on the **General** tab in the **Properties** view.

The following illustration shows a hierarchical relationship between the Customer Services Organization Unit and its sub-divisions Call Center and Back Office, and an Association between Customer Services and Sales.



Position

A Position represents a set of responsibilities for a job of work to be performed in an Organization Unit. A Position can only be assigned to one Organization Unit. It represents the responsibilities of the position within the context of the Organization Unit. The position of Administrator within the Human Resources Organization Unit may be different to the same position in the Finance Organization Unit, for example. An Organization Unit can contain many Positions.

You can specify an ideal number of people to have in a Position by specifying the **Number** field. This does not mean that you must have this number of resources in that position or that only resources with these requirements can fulfill this position. This just enables you to specify what the ideal requirements of the position are.

You can specify privileges and system actions associated with a Position. As an example of privileges the Accounts Organization Unit may be able to sign off expenses up to $500 for example, but the Accounts Manager may be authorized to sign off expenses up to $1000. Whether positions inherit the privileges specified for an Organization Unit is determined by the run-time environment to which the Organization Model is exported. It is not defined in the Organization Modeler itself.

For each Position, there are various properties you can assign. For example, you can specify the location of the Position and how long it should exist for, and you can also specify Capabilities and Resources for a Position. See Position Properties on page 85 for more information.

Group

A Group represents a job type within your organization. It allows resources to be grouped by their job characteristics. These may be general job characteristics or characteristics that apply in a particular context. For example, the A-Z Insurance Company might have groups for Loss Adjuster or Claims Handler. At the same time, you may have employees who have specializations in the Home Insurance sector. You may want to group these employees together based on these specializations, as shown below.

You do not assign Positions within the organization model, nor named users, to Groups when you are working with Organization Modeler. Users are assigned to groups at runtime using the Organization Browser in TIBCO Workspace.

Groups can be hierarchical, in other words a Sub Group can be created from a parent Group, or exist along side each other. All members of a sub-group are members of the parent Group.

You can specify system actions, capabilities and privileges for Groups. A Group can have as many Capabilities as you want. This is useful for example, if you require resources with particular capabilities. You can group resources together based on their Capabilities.

For each Group, there are various properties you can assign. For example, you can specify a description and purpose for the Group. See Group Properties on page 87 for more information.

Capability and Privilege

Capabilities represent the skills that are available within an organization, for example language skills or possession of a professional qualification. Once a Capability has been created it can be assigned to Groups and Positions. Capabilities assigned to a Group or Position represent "entry criteria" for that group or position: it is necessary to have that capability in order to be a valid member of the group or position.

Privileges represent the authority that an Organization Unit, Position or Group can have within an organization, such as the authority to approve expenditure up to a defined amount. You can assign Privileges to Groups, Positions and Organization Units. Privileges may be based on budgets and spending, work/product approval, resource allocation, or other factors. Specifying Privileges enables you to see the chains of authority throughout your organization.

You can also associate a **system action** with a privilege. System actions are tasks that might need to be authorized in some way. Like Privileges, they can be assigned to Organization Units, Positions or Groups. At run time, only users who hold the associated privilege are then allowed to execute that system action. See System Actions on page 115.

Whether positions inherit the privileges specified for an Organization Unit is determined by the run-time environment to which the organization model is exported. It is not defined in the Organization Modeler itself.

You can create categories for Capabilities and Privileges. Categories are a way of grouping your Capabilities and Privileges into meaningful units. You could create a Category for Language and then create Capabilities within that category for each language spoken in your organization; for example French, Polish, Chinese etc.

You can add extra information about a Capability or Privilege by assigning a Qualifier. For example, you may have created a Capability that represents an exam qualification but for a particular Position you may want to qualify that Capability by specifying a grade as well. To add qualifiers, check the **Has qualifier** box on the **General** tab in the Properties view. The **Qualifier** tab then becomes available. On that tab you can name and describe the qualifying information, and specify a data type that determines its allowable values, and if required specify a default value. For a description of the data types that qualifying information can have, see About Data Types on page 121. Then, when you assign the Capability or Privilege to a Group or Position, you can specify the value of the Qualifier by typing it into the **Value** field on the **Capabilities** or **Privileges** tab.

You can assign other properties to Capabilities and Privileges. For example, you can specify a purpose and/or description for the Capability or Privilege, see Capability Properties on page 89 and Privilege Properties on page 90 for more information.

Locations

Locations represent the locations that your organization uses. A location may be a place, an office building, or even a room in an office, depending on your requirements. You can create locations in the Organization Model and then assign them to Organizations, Organization Units and Positions. This enables you to see how your organization is distributed across its various locations.

You can also specify other properties for Locations. See Location Properties on page 91 for more information.

Resources

Resources are used to specify items such as people, equipment or buildings. You can use Resources within your organization model to identify such items and define how they relate to Organization Units and Positions.

When you create a resource, you can assign various properties to it. See Resource Properties on page 92.

A resource can be assigned a Resource Type if Resource Types are defined in your schema, and will have any Attributes defined for that type. For example, you may want to define attributes of **FirstName** and **Surname** for a Resource that represents an employee, or **TelephoneNumber** for both an employee and for another type of Resource that represents a meeting room. See Schema on page 122 for more information about Schemas and Resource Types.

The following Resource Types are provided in the default Schema:

- Durable Resource Type. You can use this for items like buildings which have a long lifespan.
- Consumable Resource Type.
- Human Resource Type. A Human Resource Type must always be present in the Schema, and there can be only one Human Resource Type present in the schema.

See Creating a Schema on page 43 for details of creating a Schema with or without using the standard Types. See Resource Type Properties on page 96 for more details of the Human Resource Type.

Resources and BPM

Any Resources that you have defined are not exported to BPM, and are not used at runtime.

The only exception to this is that the Human Resource type, and any attributes defined for it, are exported to BPM.

Therefore, a Human Resource type must always be present in the Schema. Even if you choose not to include the standard Types in your Schema, a Human Resource type is always present, and it cannot be deleted from the Schema.

Entities within the organization model are mapped to Resources as defined in TIBCO Workspace. You can:

- Map BPM Resources, which represent actual user IDs obtained from an LDAP-compliant directory, to Positions and Groups in the organization model,

- Map attributes assigned to the organization model's Human Resource Type to LDAP attributes in the LDAP sources you have used.

See the chapter "Using the Organization Browser" in the *TIBCO Workspace User's Guide* for details of mapping resources.

This means that any sort of information that you have reflected in defining Resources within Organization Modeler, including for example if you have used Resources to represent anything other than users, will not be represented in BPM after deployment.

Queries

The use of queries is not currently supported in the Organization Model.

Queries are a way of identifying an entity within the organization model to use as a suitable participant for a task in a business process.

You can specify queries either:

- As strings of text. These are not checked or validated within Organization Modeler. A business analyst might enter the query as free text, which a solution designer can implement when the analysis is implemented as a process for a specific destination environment,
- As Resource Query Language (RQL). This is a scripting language which you can use to write queries specifying which entity within the organization model should be selected for work allocation. It is described in the BPM implementation documentation. Organization queries entered in RQL are validated in TIBCO Business Studio.

System Actions

System actions are actions that a user may wish to perform at runtime but that need to be authorized, or need to be restricted to users with a certain level of authority. These actions might include, for example re-allocating work-items, skipping work-items, viewing another user's work list, or administering resources.

This authorization is implemented by associating system actions with **privileges** within Organization Modeler. See Capability and Privilege on page 110 for more details about privileges. See the BPM *Concepts* documentation for an introduction to system actions.

In Organization Modeler:

- For the Organization Model, the System Actions tab of the Properties view lists all the system actions that are available, and any privileges with which each is associated.
- For Organization Units, Positions and Groups, the System Actions tab of the Properties view lists the subset of system actions that are available for that class of entity, and any privileges with which each is associated.

In all these cases you can associate a system action with one or more privileges. As described in Capability and Privilege on page 110, privileges can have qualifiers which determine the level of the privilege. At run time, only users who hold the associated privilege with any required level of qualifier (or if more than one privilege is associated with a particular system action, users who hold **all** the associated privileges) are then allowed to carry out that system action.

As well as possessing the correct privileges, users may need to belong to an appropriate **user access set** in order to perform a particular system action. See the TIBCO Workspace documentation for more information on user access sets.

The **Organization Admin** system action allows the user to see organization models other than the one they are part of, and allows them to view process instances started by members of these organizations.

At runtime, BPM maintains a list of system actions and of privileges, as defined in the organization model, and thus determines whether a user is authorized to carry out a particular action.

System Actions Table

The following system actions can be associated with privileges at the Organization Model level:

Application Configuration	*Not currently used* This is used in Workspace to control access to the Configuration Administrator.
Auto Open Next Work Item	Open the next work item automatically after the previous one is opened
Browse Model	Browse the organization model
Bulk Cancel Process Instance	
Bulk Purge Process Instances	
Bulk Resume Process Instance	
Bulk Surpend Process Instance	
Cancel Process Instance	Cancel a process instance.
Cancel Work Item	*Not currently used* Cancel a work item.
Change Allocated Work Item Priority	
Change Any Work Item Priority	
Close Other Resources Items	Close work items that are currently allocated to other users
Contribute Gadget	Contribute a gadget in Openspace.
Create Resource Admin	Create resources and edit resource information.
Delete Calendars	Delete one or more calendars.
Delete LDAP Admin	Delete LDAP containers.

Delete Resource Admin	Delete resources.
Edit Hub Policy	Edit Hub Policy in Openspace.
Export LDAP Admin	Export LDAP container and resource mapping information.
Halted Process Administration	
Handle Process Migration	Authorizes the migration of a process from one version to another. If you do not have the system action set, you cannot migrate the process.
Import LDAP Admin	Import LDAP container and resource mapping information.
LDAP Admin	List resources in an LDAP container.
List Process Template Audit Trail	*Not currently used*
Manage Gadgets	Manage gadgets in Openspace.
Open Other Resources Items	Opens work items for other resources.
OpenspaceFeatureSetA	Associated with lockdown in Openspace. See *TIBCO Openspace Customization Guide*.
OpenspaceFeatureSetB	Associated with lockdown in Openspace. See *TIBCO Openspace Customization Guide.*
OpenspaceFeatureSetC	Associated with lockdown in Openspace. See *TIBCO Openspace Customization Guide.*
Open Work Item Audit Trail	*Not currently used*
Organization Admin	*n/a* Allows the user to see organization models other than the one they are part of.
Pend Work Item	View "pended" work items, i.e., work items that have been hidden until a specified date/time, or period of time has expired.
Purge Process Instances	*Not currently used*

Query Process Instance	View list of process instances.
Query Process Template	View list of process templates.
Read Calendars	Read a calendar.
Read Parameters	*Not currently used* This is used in the Organization Browser to control viewing and editing resource attributes.
Read Push Destinations	*Not currently used* This is used in the Organization Browser to control viewing and editing push destinations.
Reallocate To Offer Set	Reallocate the work item to the offer set
Reallocate Work Item To World	Reallocate the work item to all users
Reschedule Work Item	Reschedule a work item.
Resolve Resource	Used by many functions to access resource information.
Resource Admin	View list of resources in a group or position.
Resume Process Instance	Resume an instance of a process that has been suspended
Schedule Work Item	*Not currently used by any public API* Schedule a work item
Set Deadline Expiration	Set the time for a deadline to expire
Set Priority	Set the priority for a work item
Set Resource Order Filter Criteria	Set filter criteria for a resource order
Show Process Instance Audit Trail	*Not currently used*
Skip Work Item	*Not currently used by any public API* Skip a work item in a work list

Start Business Service	Start an instance of a business service.
Start Process	Start an instance of a process
Suspend Process Instance	Suspend a currently running instance of a process
Suspend Work Item	Not currently used. (There is no means to suspend a work item -- you can suspend a process instance, which causes associated work items to become suspended -- but no separate functions for work items.)
User Admin (Business Studio actually shows that as "User Settings")	Controls whether or not user settings will be persisted.
View Hub Policy	View a hub policy in Openspace.
View Global Work List	
View Work List	Create a supervised work view that either contains work items offered to an organizational entity, or that are currently in the Inbox of an individual resource.
Work Item Allocation	Allocate a work item
Write Calendars	Edit Calendars.
Write Parameters	*Not currently used by any public API* Edit resource attributes in the Organization Browser.
Write Push Destinations	Edit push destinations for an organization or resource.

The following system actions are considered "scoped", that is, they are set on specific groups, organization units, and positions using the Organization Modeler. The scoped system actions are specifically used to control access to, and functions from, supervised work views.:

Close Other Resources Items	Controls whether you can choose to cancel a work item in a supervised work view for another resource.

Open Other Resources Items	Open work items that are currently allocated to other users
Reallocate To Offer Set	Controls whether you can choose to reallocate a work item to a specific person in the original offer set.
Reallocate Work Item To World	Reallocate the work item to all users
Set Resource Order Filter Criteria	Set filter criteria for a resource order
View Work List	Controls access to supervised work views: • To create a supervised work view for an organizational entity, you must possess the privilege assigned to the system action, BRM.viewWorkList, on the group, organization unit, or position. • To create a supervised work view for an individual resource, you must possess the privilege assigned to the system action, BRM.viewWorkList, on a specific position to which the resource has been mapped.
Work Item Allocation	Allocate a work item

About Data Types

The following elements of an Organization Model can have data types:

- Schema Type attributes, see Attributes on page 126.
- Qualifying information for Capabilities and Privileges, see Capability and Privilege on page 110.

The table below describes the data types that these elements can have.

Type	Description
Text	Any combination of alphanumeric characters.
Integer	A whole number, including zero and negative numbers.
Boolean	A value of True or False, or blank.
Decimal	Any number, positive or negative, up to the number of decimals you specify.
DateTime	A date and time in the format of the locale on your machine.
Date	Any date in the format of the locale on your machine.
Time	Any time in the format of the locale on your machine.
EnumSet	A data type that can contain a list of values. Selecting this type enables you to specify a set of enumerated values.
Enum	An enumerated type. Selecting this type enables you to specify an enumerated value. Each Enum is one of the list of values that make up an EnumSet.

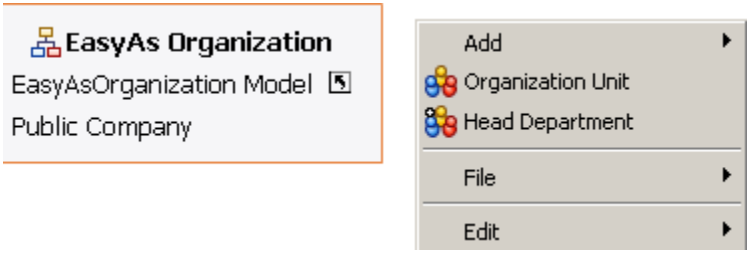
Schema

Organization Modeler enables you to create an embedded Schema for use in the definition of your organization model.

Using a Schema, you can specify Types that you intend to use for certain elements within your organization. By setting up these types, you can create a generic model or template for your organization model. Defining different Types of Organization, Organization Unit, Position, Resource and so on, and choosing and defining Attributes for those Types, enables you to extend the model provided by the default schema until it is a close representation of the components that make up the organization that you wish to model.

The containment relationships between components can also be defined in the schema's Types. For example, from an Organization element that you have assigned an Organization Type, Organization Modeler enables you to create a child Organization Unit of a defined type, because the relationship between the two Types is described in the schema.

An example of this is shown below. One type of Organization - the Public Company organization type - is defined in the standard Schema. One type of Organization Unit element - Head Department - is defined as a Member of that Organization Type. If your Organization is defined as a Public Company, and you right-click in the Organization Editor to add an Organization Unit, you can select either an un-Typed Organization Unit, or a Head Department unit.



This allows you to create specific organizational structures in a Schema, so that when you need to create an instance of that structure in your organization model, you can use the structures defined in the Schema. For example, you may want to create an organization structure for temporary projects that consists of specific Organization Units and Positions. You can create this structure in a Schema then create an instance of a temporary project based on the Schema. See Using a Schema in an Organization Model on page 73 for more information.

The benefits of Schemas are:

- They can be used as templates for organization models that need to be exchanged between systems. Some extended enterprises refer to organizational concepts differently, for example, "region" vs. "district". By specifying types for particular

components, you create a generic schema for your Organization Model that enables it to be exchanged with other systems.

- They enable you to create a language or vocabulary to express the organization concepts that are applicable to your organization.
- They enable you to create additional semantic data for your existing organization concepts. For example, for a Position whose type is Manager, you could create an attribute called bonus to specify what type of bonus should be given.

When you use a Schema within the Organization Modeler you can either:

- Use the default Schema, which comes delivered with standard Types for the schema elements,
- Create an empty Schema which does not contain any standard Types (except for the Human Resource Type, which must always be present).

You can also decide to use a mixture of the two approaches by adding your own types to the default schema.

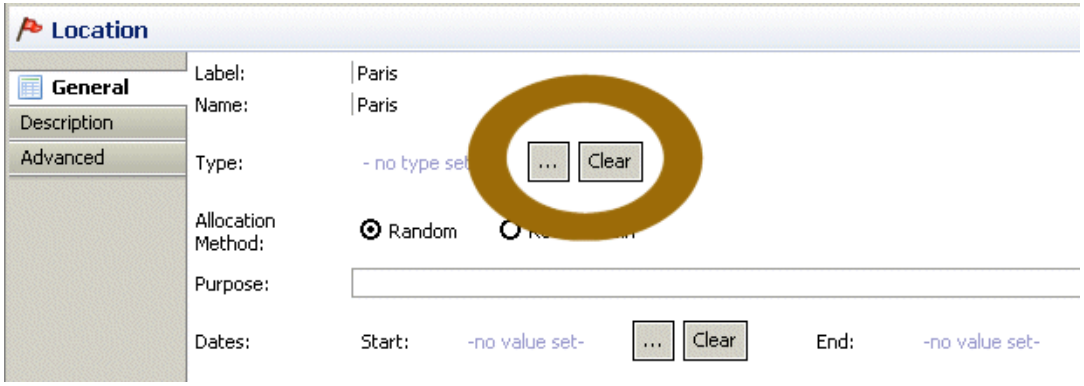
The Organization Modeler Default Schema

When you create certain components within your Organization Model, the Organization Modeler enables you to allocate standard types to these components. Not all the components within the Organization Modeler have a standard type. The following table describes the components that have types provided in the default schema, what they are, and what members each of those types contains:

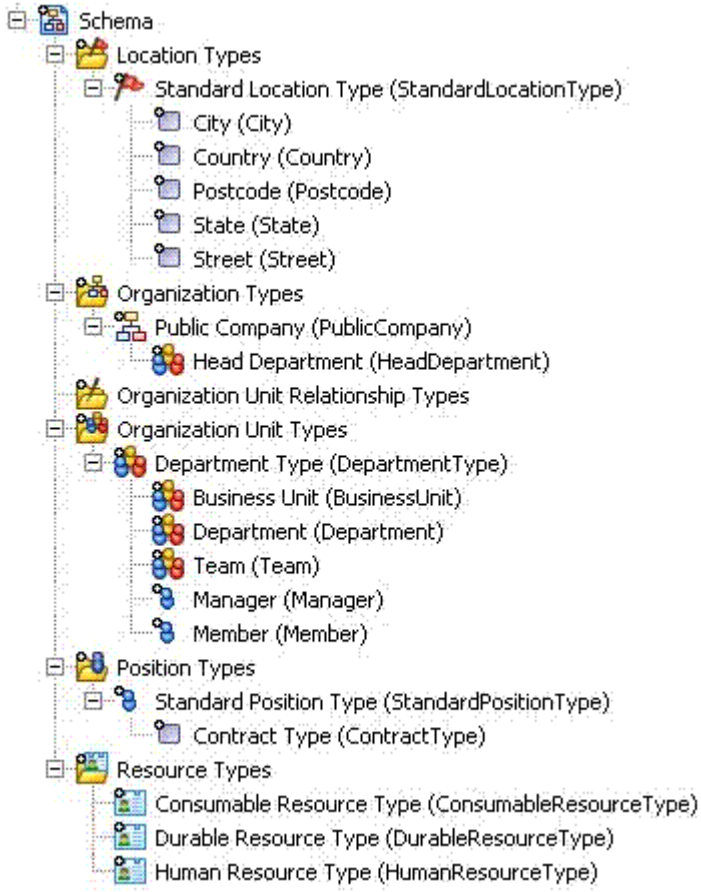
Component	Standard Type	Unit Members	Position Members
Organization	Public Company	Head Department	
Organization Unit	Department Type	Department Business Unit Team	Manager Member
Position	Standard Position Type		
Location	Standard Location Type		
Resource	Human Resource Type Consumable Resource Type Durable Resource Type		

When you create an Organization Model, you can decide whether to use the Standard Types provided by the Organization Modeler, create new Types, or create your own Schema, depending on your requirements.

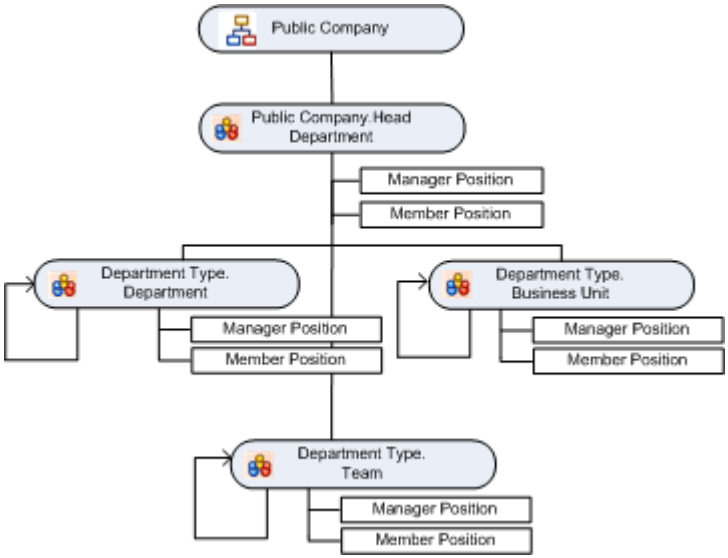
When a component that has a standard type defined for it is created, you can assign the type in the **Properties View** for the component. By default no type is assigned, as in the following illustration. You can assign a type by clicking ... and selecting either the standard type or any other type that has been defined.



The following diagram shows the default schema as it appears in the Project Explorer, if you chose to apply it when creating the project:


Schema
Location Types
Standard Location Type (StandardLocationType)
City (City)
Country (Country)
Postcode (Postcode)
State (State)
Street (Street)
Organization Types
Public Company (PublicCompany)
Head Department (HeadDepartment)
Organization Unit Relationship Types
Organization Unit Types
Department Type (DepartmentType)
Business Unit (BusinessUnit)
Department (Department)
Team (Team)
Manager (Manager)
Member (Member)
Position Types
Standard Position Type (StandardPositionType)
Contract Type (ContractType)
Resource Types
Consumable Resource Type (ConsumableResourceType)
Durable Resource Type (DurableResourceType)
Human Resource Type (HumanResourceType)

The following diagram shows how the default schema reflects an organizational structure.



In this diagram:

- **Public Company** is the delivered Organization type.
- **Head Department** is an Organization Unit type that is a member of **Public Company**. This type has a multiplicity defined as **0..1**, so there can only be one **Head Department**.
- **Department**, **Business Unit** and **Team** are the other Organization Unit types delivered. All of these need to be in a hierarchical relationship with another typed organization unit as their parent. The hierarchy does not need to be that shown in the diagram, however; any one of the delivered organization units can be the parent of any other.

Attributes

For each of the Types in the schema you can create Attributes. An Attribute enables you to add some extra semantic information to each individual type. For example, for a Position whose type is Manager, you could create an Attribute called bonus to specify what type of bonus should be given to this Position. The Standard Position Type in the default schema has the attribute **Contract Type** assigned to it. The Standard Location Type delivered in the default schema has the following attributes already defined:

- Country
- State
- City

- Street
- Postcode

When you create an attribute, you must specify a data type that determines its allowable values. For a description of the data types an attribute can have, see About Data Types on page 121.

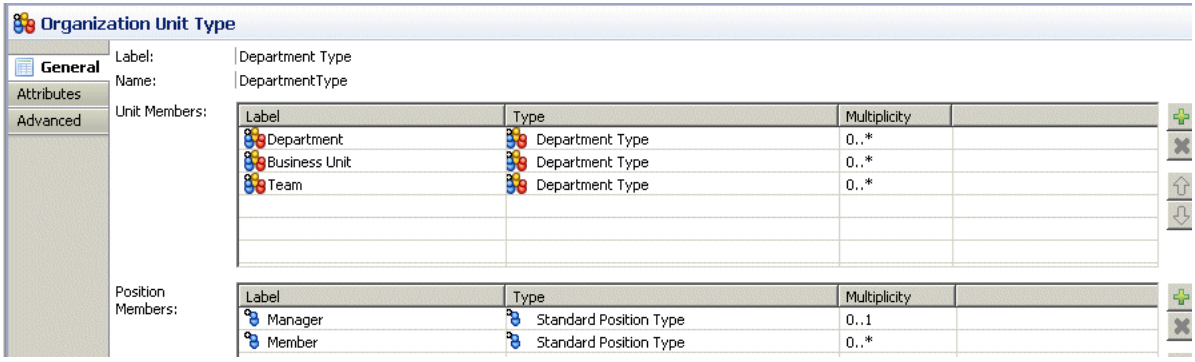
Do not change the data type of an Attribute assigned to any Type in the Schema, once the Organization Model including it has been deployed to the BPM Directory Engine. Since multiple versions of a Model can exist in BPM, changing the data type of an Attribute between versions can create inconsistencies.

Members

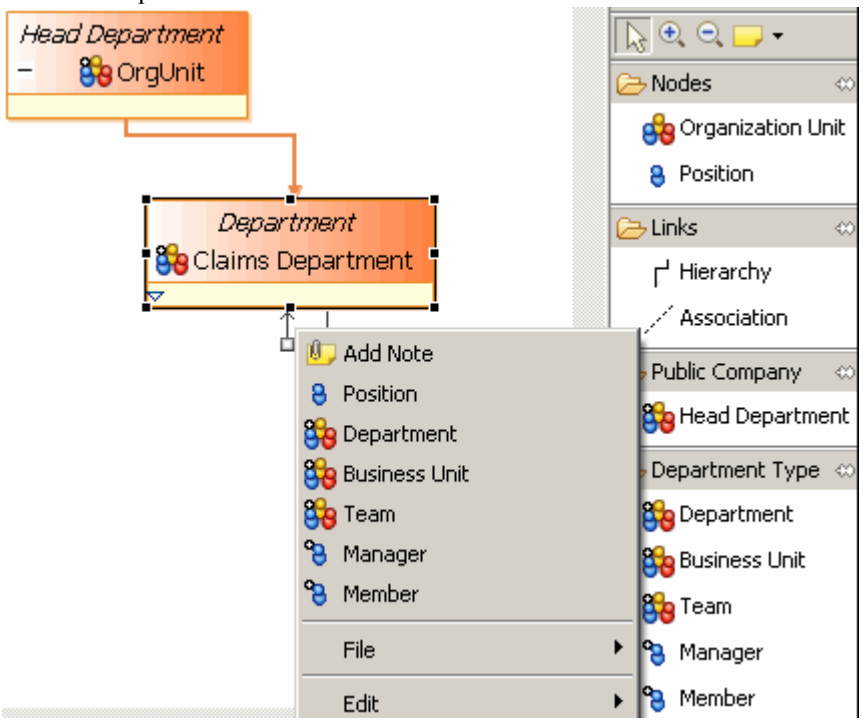
For the Organization Type and Organization Unit Type you can create Members. The following table describes what Members you can create for each type:

Type	Members
Organization Type	Unit Members
Organization Unit Type	Unit Members, Position Members

The use of Members enables you to group together a set of Position Types and Organization Unit Types into meaningful units. Once you have created your Organization Unit Types and Position Types, you can create a template structure by grouping them together within an Organization Type or Organization Unit Type as Members of that Type. An illustration of the Department type delivered in the default schema is shown below.



The Department, Business Unit and Team Organization Unit Types have been grouped together in the Department organization unit type as unit members of the type. This means that when an Organization unit element is created in an Organization Model using the default schema, these elements are available from the **Add Child** menu, from the context menu or from the palette. This is illustrated below.



When you subsequently add an Organization Unit to your Organization diagram, you can select one of these Elements from the Element field on the **General** tab, and assign it to the new Organization Unit.

See Using a Schema in an Organization Model on page 73 for more information.

Multiplicity

You can specify whether TIBCO Business Studio needs to allow for multiple copies of an element. The following table describes the multiplicity values you can specify:

Indicator	Meaning
0..1	No instances or one instance (optional).
0..*n*	Multiple instances from zero to *n* where *n* is greater than zero.

Indicator	Meaning
0..*	Any number of instances; * denotes that there is no limit.
n..m	Multiple instances where *n* and *m* are zero or more. For example: 1..2 1..* The latter would mean that there must be at least one instance, but there is no upper limit.