

TIBCO iProcess[®] Modeler

Basic Design

Software Release 11.4
July 2013

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, Two-Second Advantage, TIBCO ActiveMatrix BusinessWorks, TIBCO Business Studio, TIBCO EnterpriseMessage Service, TIBCO Hawk, TIBCO iProcess, TIBCO iProcess Suite, and TIBCO Rendezvous are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Enterprise Java Beans (EJB), Java Platform Enterprise Edition (Java EE), Java 2 Platform Enterprise Edition (J2EE), and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle Corporation in the U.S. and 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.

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 © 1994-2013 TIBCO Software Inc. ALL RIGHTS RESERVED.

TIBCO Software Inc. Confidential Information

Contents

Preface	vii
Related Documentation	viii
TIBCO iProcess Modeler Documentation	viii
Other TIBCO Product Documentation	ix
Typographical Conventions	x
Connecting with TIBCO Resources	xiii
How to Join TIBCOCommunity	xiii
How to Access TIBCO Documentation	xiii
How to Contact TIBCO Support	xiii
Chapter 1 Defining a Procedure	1
Placing Procedure Objects	2
Selecting Objects	5
Editing Objects	6
Linking Procedure Objects	7
A Normal Link	7
A Withdraw Link	8
A Deadline Link	8
A Deadline Withdraw Link	8
Routing the Business Process	9
Assigning Addressees	9
Single-User Addressees	9
Multiple User or Group Addressees	10
Roles	11
Fields	12
Defining Step Status Options	13
Enabling Steps to be Forwarded	13
Enabling the Contents of a Form to be Copied	13
Stop Work Items Being Deleted on Withdraw	14
Preventing a Case from Being Suspended	14
Chapter 2 Creating Fields and Forms	15
About iProcess Fields	16
What are Single Instance Fields?	16
What are Array Fields?	16

- Defining Fields 18
- Creating a Form 21
 - Inserting Fields into Forms 21
 - Editing Marked Fields 23
 - Defining Field Help 23
 - Using Embedded and Ampersanded Fields 24
 - Marking Application Fields 27
 - Removing Fields From Forms 31
 - Using Conditional Text to Dynamically Change a Form 31
 - Inserting a Use File 33
 - Using Form Commands 34
 - Editing Your Form 35
 - Changing the Form Type of a Step 35
- Chapter 3 Controlling Data Input On Forms 37**
 - Using Field Validations 38
 - Copying Field Values 40
 - Using Field Calculations 41
 - Conditional Calculations 43
 - Calculating Text Fields 44
 - Calculating the Case Description 44
 - Using Delimiters and Key Words 46
 - Using iProcess Tables 48
 - Defining a Table Field 48
 - Marking Table Fields in your Form 49
- Chapter 4 Using Deadlines in Procedures 51**
 - Defining a Deadline 52
 - Drawing the Deadline Link 54
 - Using Deadlines on a Sub-Procedure 55
 - Dynamically Recalculating Deadlines 56
- Chapter 5 Using Conditional Actions 59**
 - Defining a Conditional Action 60
- Chapter 6 Withdrawing Steps From the Procedure 63**
 - Example of Using a Withdraw Action 64
 - Defining a Withdraw Action 65
 - Defining a Deadline Withdraw Action 66
 - Defining a Withdraw Action on a Sub-Procedure 67

Chapter 7 Defining Waits in the Procedure	69
Example of Using a Wait	70
Defining a Wait Action	71
How the iProcess Engine Processes Waits	71
Viewing Step Status on the TIBCO iProcess Engine	71
Using Waits in Loops	72
Using Waits with Conditions	79
Using Waits with Withdraw	80
Chapter 8 Making Procedures Easier to Follow	83
Setting TIBCO iProcess Modeler Options	84
Using Swim Lanes	85
Enabling Swim Lanes	85
Swapping Between Swim Lane Types	89
Configuring Swim Lanes	90
Changing Step Icons	97
Annotating Procedures for Clarity	98
Changing the Object Label Position	100
Working with Links	101
Setting Link Labels and Icons	102
Changing Link Styles and Animation	104
Using Routers to Simplify Visual Layout	106
Using Complex Routers to Simplify Procedure Logic	107
Using GOTOSTEP to Simplify the Procedure Routing	110
Zooming In and Out of a Procedure	111
Changing Procedure Orientation	112
Using the Snap-To Grid	113
Saving a Procedure as an Image	114
Chapter 9 Customizing the Process Step Definer	115
Colors	116
Select Font	117
Dynamic Scroll	118
Show Field Names	119
Line Length	120
Tabs	121
Nesting Level	122

Appendix A Troubleshooting Procedure Definitions 125

An Under Construction Symbol Appears on a Step 126

Changing the Currency Unit 127

Index 129

Preface

This guide explains and demonstrates the basic principles of procedure design using a combination of the following products:

- TIBCO iProcess[®] Workspace (Windows)
- TIBCO iProcess Modeler

Topics

- [Related Documentation, page viii](#)
- [Typographical Conventions, page x](#)
- [Connecting with TIBCO Resources, page xiii](#)

Related Documentation

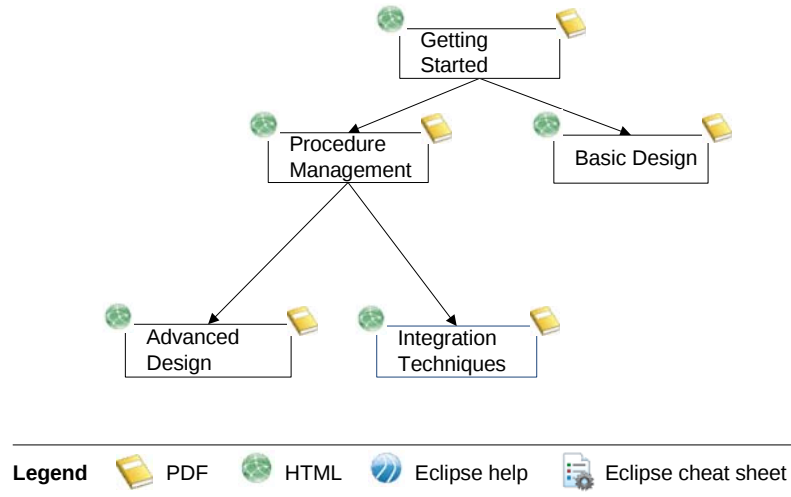
This section lists documentation resources you may find useful.

TIBCO iProcess Modeler Documentation

The following documents form the TIBCO TIBCO iProcess Modeler and TIBCO iProcess Workspace (Windows) documentation set, which are supplied with the TIBCO iProcess Workspace (Windows) software:

- *TIBCO iProcess Workspace (Windows) Installation* Read this manual for instructions on site preparation and installation.
- *TIBCO iProcess Workspace (Windows) Release Notes* Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.
- **TIBCO iProcess Suite Documentation** This documentation set contains all the manuals for TIBCO TIBCO iProcess Modeler, TIBCO iProcess® Workspace (Windows), and other TIBCO products in TIBCO iProcess® Suite. The manuals for TIBCO TIBCO iProcess Modeler and TIBCO iProcess Workspace (Windows) are as follows:
 - *TIBCO iProcess Workspace (Windows) User's Guide*
 - *TIBCO iProcess Modeler Getting Started*
 - *TIBCO iProcess Modeler Procedure Management*
 - *TIBCO iProcess Modeler Basic Design*
 - *TIBCO iProcess Modeler Advanced Design*
 - *TIBCO iProcess Modeler Integration Techniques*
 - *TIBCO iProcess Expressions and Functions Reference Guide*
 - *TIBCO iProcess Workspace (Windows) Manager's Guide*

If you are new to iProcess procedure development, you are advised to follow the reading path shown next. The documentation road map shows the relationships between the books and online references in this product's documentation set.



Other TIBCO Product Documentation

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

- TIBCO ActiveMatrix BusinessWorks™
- TIBCO Business Studio™
- TIBCO Enterprise Message Service™
- TIBCO Hawk®
- TIBCO Rendezvous®

Typographical Conventions

The following typographical conventions are used in this manual.

Table 1 General Typographical Conventions

Convention	Use
<i>SWDIR</i>	TIBCO iProcess Engine installs into a directory. This directory is referenced in documentation as <i>SWDIR</i>. The value of <i>SWDIR</i> depends on the operating system. For example, • on a Windows server (on the C: drive) if <i>SWDIR</i> is set to the C:\swserver\staffw_nod1 directory, then the full path to the <code>swutil</code> command is in the C:\swserver\staffw_nod1\bin\swutil directory. • on a UNIX or Linux server if <i>SWDIR</i> is set to the /swserver/staffw_nod1 directory, then the full path to the <code>swutil</code> command is in the /swserver/staffw_nod1/bin/swutil directory or the <code>\$SWDIR/bin/swutil</code> directory. Note: On a UNIX or Linux system, the environment variable <code>\$SWDIR</code> should be set to point to the iProcess system directory for the <i>root</i> and <i>swadmin</i> users.
code font	Code font identifies commands, code examples, filenames, pathnames, and output displayed in a command window. For example: Use <code>MyCommand</code> 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, <code>MyCommand</code> is enabled: <code>MyCommand [enable disable]</code>

Table 1 General Typographical Conventions (Cont?)

Convention	Use
<i>italic font</i>	Italic font is used in the following ways: • To indicate a document title. For example: See <i>TIBCO ActiveMatrix BusinessWorks Concepts</i>. • To introduce new terms. For example: A portal page may contain several portlets. <i>Portlets</i> are mini-applications that run in a portal. • To indicate a variable in a command or code syntax that you must replace. For example: <code>MyCommand PathName</code>
Key combinations	Key name separated by a plus sign indicate keys pressed simultaneously. For example: <code>Ctrl+C</code>. Key names separated by a comma and space indicate keys pressed one after the other. For example: <code>Esc, Ctrl+Q</code>.
	The note icon indicates information that is of special interest or importance, for example, an additional action required only in certain circumstances.
	The tip icon indicates an idea that could be useful, for example, a way to apply the information provided in the current section to achieve a specific result.
	The warning icon indicates the potential for a damaging situation, for example, data loss or corruption if certain steps are taken or not taken.

Table 2 Syntax Typographical Conventions

Convention	Use
[]	An optional item in a command or code syntax. For example: <code>MyCommand [optional_parameter] required_parameter</code>
	A logical OR that separates multiple items of which only one may be chosen. For example, you can select only one of the following parameters: <code>MyCommand param1 param2 param3</code>

Table 2 Syntax Typographical Conventions (Cont?)

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

Connecting with TIBCO Resources

How to Join TIBCOCommunity

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

How to Access 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.

Chapter 1 **Defining a Procedure**

This chapter describes the different types of procedure objects you can use to define your procedure and how to link them together. Each procedure object performs a specific task such as displaying a form for a user to fill out or updating information on an external system.

Topics

- [Placing Procedure Objects, page 2](#)
- [Selecting Objects, page 5](#)
- [Editing Objects, page 6](#)
- [Linking Procedure Objects, page 7](#)
- [Routing the Business Process, page 9](#)
- [Defining Step Status Options, page 13](#)

Placing Procedure Objects

The TIBCO iProcess Modeler Tool Bar shows the different objects that are available to the procedure definer. These are:

Object		Description
Pointer		This is the default tool. It is used to select objects that have been added to a procedure. After selecting, they can be moved or deleted. Double-clicking an object in a procedure with the pointer tool displays the Step Definition dialog.
Router		The router is used when you want a line to follow a particular route between two objects to improve clarity. When drawing a line between two objects, right-click to place a router at the cursor position. There is no toolbar icon for the router.
Complex Router		The complex router is a modeling object used to simplify complicated procedures. There are several ways of using the complex router. For more information see Using Complex Routers to Simplify Procedure Logic on page 107 .
Step		The Step is the most frequently used object. It allows you to define a step to display, specify an addressee to whom the step should be sent, and optionally a deadline by which the step (work item) must be completed.
Script		Scripts can be created and run from the business process definition. See “Using Scripts” in <i>TIBCO iProcess Modeler- Advanced Design</i> guide for more information about creating scripts.
Event		Events enable the flow of a case to be controlled and the data associated with it to be changed, externally to iProcess. For more information, see “Using Events” in the <i>TIBCO iProcess Modeler - Integration Techniques</i> guide.

Object	Description
Condition 	A condition is a decision point in the business process. For example, in a form where approval has to be given or refused, the approval field can be interrogated and the business process will branch depending on the value of the field. For more information, see Defining a Conditional Action on page 60.
Wait 	A wait is a synchronization point in a business process where parallel paths join together again. For more information, see Defining a Wait Action on page 71.
Stop 	A stop indicates the end of that branch of the procedure, or the end of a branch of the procedure. It is optional but does help to clarify the business process definition in the iProcess Modeler.
Annotation 	Selecting this object enables you to enter text to document your procedure.
Sub-procedure 	The sub-procedure tool allows you to specify a call to a new or existing sub-procedure. For more information, see “Defining a Call to a Static Sub-Procedure” in the TIBCO iProcess Modeler - Advanced Design guide.
Dynamic Sub-procedure 	The dynamic sub-procedure tool is used to specify a call to one or more sub-procedures. The sub-procedures that are run are only determined when a case of the procedure is run. For more information, see “Defining a Dynamic Call to Multiple Sub-Procedures” in the TIBCO iProcess Modeler - Advanced Design guide.

Object	Description
Graft	The graft tool is used when you want to start sub-procedures from an external application and attach them to the main procedure. For more information, see the <i>TIBCO iProcess Modeler - Integration Techniques</i> guide.
EIS Report	The EIS Report allows you to define a report based on the field values held in the cases of your procedure. See <i>TIBCO iProcess Workspace (Windows) Manager's Guide</i> for more information.
Start	The Start object is not on the tool bar but is always present when you open the TIBCO iProcess Modeler. It indicates the start of the procedure.
EAI Step	The Enterprise Application Integration (EAI) step enables you to interact with third party software systems such as relational databases or legacy systems. Different types of EAI step are used to interact with specific applications, for example, the SQL EAI step is used to interact with a SQL Server. Refer to “Using EAI Steps” in the <i>TIBCO iProcess Modeler - Integration Techniques</i> guide for more information.
Align Horizontal	The Align horizontal tool horizontally aligns all selected objects with the currently focused object.
Align Vertical	The Align vertical tool vertically aligns all selected objects with the currently focused object.
Layout Procedure	The Layout Procedure tool changes the chart orientation from left-to-right to top-down.
Snap	The Snap tool snaps the selected objects to the nearest grid square.

Selecting Objects

You can click an object with the pointer tool to select a single object. To select multiple objects, lasso the objects by clicking and dragging the cursor over the objects. You can also select multiple objects using the **CTRL** key. Click an object, then hold the **CTRL** key while you click the next object. Both objects are selected.

Editing Objects

The following table contains a list of edit commands and descriptions.

Command	Description
Copy	Select one or more objects in the procedure and click  or Edit > Copy . The selected items are copied to the clipboard.
Cut	Select one or more objects in the procedure and click  or Edit > Cut . The selected items are removed from the procedure and placed on the clipboard.
Paste	After selecting the items and clicking  or  , click  , then click the target location in a procedure and the selected items are pasted. To cancel the Paste operation, press Escape . Depending on the number of steps being pasted and the type of conflicts that occur, a Wizard might be displayed to guide you through the Paste operation.
Move	Select one or more objects in the procedure and drag the selected objects to the new location within the procedure. To cancel the Move operation, press Escape .

Linking Procedure Objects

The link tool is automatically selected when you move the cursor near an object that supports links. To cancel a link operation, press **Escape**.

The way in which you link procedure objects dictates the action that is carried out in a procedure flow. In some circumstances, you might want a procedure map to flow from right-to-left or top-to-bottom instead of the default flow of left-to-right. In a horizontal (left-to-right) procedure flow (unless the object side is already used for a different purpose):

- A release action link can be started from the right or left side of an object.
- A deadline action link can be started from the bottom or top of an object.
- A process step link can be ended on the left or right side of an object.
- A withdraw step link can be ended on the top or bottom of an object.

In a vertical (top-to-bottom) procedure flow (unless the object side is already used for a different purpose):

- A release action link can be started from the bottom or top of an object.
- A deadline action link can be started from the right or left of an object.
- A process step link can be ended on the top or bottom of an object.
- A withdraw step link can be ended on the left or top of an object.

When you move the cursor to the side of an object, the default link action appears as a tool tip (after a short delay). To override the default, press the **ALT** key. When ending a link, you can override the default end-of-link type by pressing the **SHIFT** key.

Each type of link is described fully in the following sections. See [Working with Links on page 101](#) for more information.

A Normal Link

In a normal link, the second step is an action that is carried out when the first step is released.

A Withdraw Link

A withdraw link means that the step being connected to will be withdrawn from the work queue. For example, you might have two steps that are sent out in parallel but only one needs to be actioned and released. In that instance you can use a withdraw link to withdraw the second, now superfluous, step. See [Defining a Withdraw Action on page 65](#) for more information.

A Deadline Link

When you have a step that a user must release by a certain date and time or within a specific time period, you can put a deadline on the step so that another action (step) is carried out if the deadline expires. See [Defining a Deadline on page 52](#).

A Deadline Withdraw Link

Combining a deadline and a withdraw link allows you to withdraw a step when a deadline expires.

Routing the Business Process

Routing the business process tasks is achieved by the use of addressees. There are different types of addressees that you can use to route the business process.

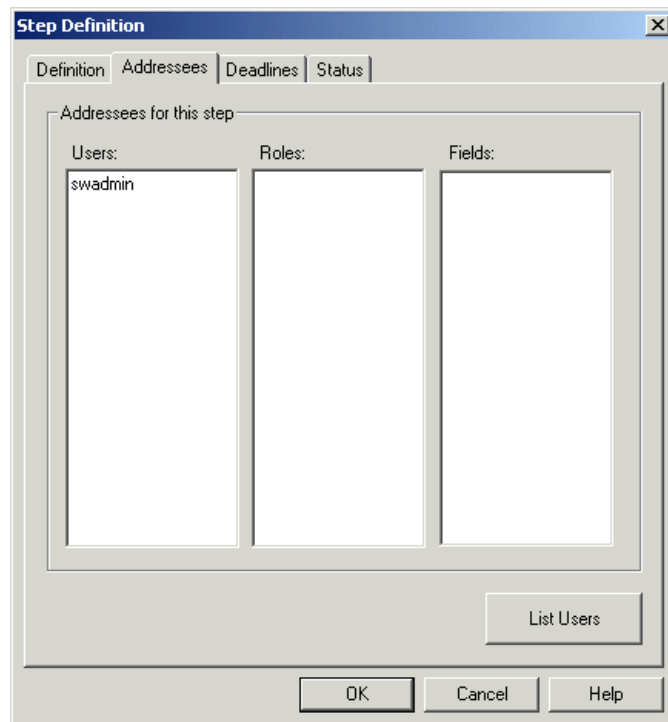
Every step that is to be delivered to a work queue must have an addressee. The addressee is the user who is responsible for completing the work item.

Assigning Addressees

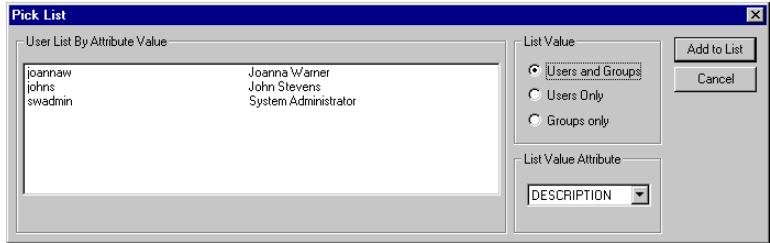
For a detailed explanation of how to assign an addressee to a step, see the *TIBCO iProcess Modeler - Getting Started* guide.

Single-User Addressees

A step can have a single addressee.



If you want a step to go to one specific user, enter a user name in the **Users** column. To see a list of all users on your iProcess installation, click **List Users**.



Highlight the user(s) you want to add to the list of addressees and click **Add to List**. An alternative is to use the special assignment of **sw_starter**. This will route a work item to the user who started that particular case.



Users must have been previously added to the iProcess installation. You can do this using the **User Manager** in the TIBCO iProcess Administrator. For more information, see “Managing iProcess Users” in *TIBCO iProcess Workspace (Windows) Manager’s Guide*.

Multiple User or Group Addressees

If you have a step that you want to send to more than one user, you can either use groups or multiple users.



Like users, groups must have been previously added to the iProcess installation.

For multiple users, enter all the user names in the **Users** column. A copy of the step will be sent to each user listed. This can be a useful way of sending the same information to a number of people. Each user is responsible for releasing their copy of the work item and the procedure will only progress when *all* users have released the work item.

Alternatively, a group can be entered in the **Users** column and the work item will be sent to that group queue. For details on how to set up a group, see “Managing Groups” in *TIBCO iProcess Workspace (Windows) Manager’s Guide*. When a work item goes to a group queue, any member of the group may open the work item, complete any input fields and release it. When the work item is open, it appears as a grayed out entry in the group queue of the other group members who are prevented from opening the item. Once the work item is released, it disappears from the other members’ group queues.

Roles

A role is a job title or function and is unique to a particular user or group. A step can be addressed to a role as shown in the following example. Multiple roles can be entered in the **Roles** column.

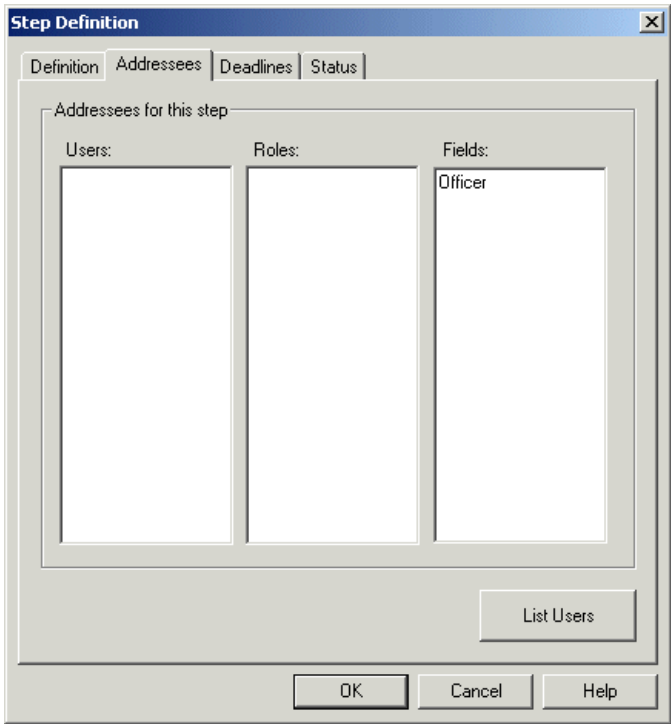
The screenshot shows a 'Step Definition' dialog box with a tabbed interface. The 'Addresssees' tab is selected. Inside the dialog, there is a section titled 'Addresssees for this step' which contains three columns: 'Users:', 'Roles:', and 'Fields:'. The 'Roles:' column contains the text 'Supervisor'. Below these columns is a 'List Users' button. At the bottom of the dialog are 'OK', 'Cancel', and 'Help' buttons.

Users:	Roles:	Fields:
	Supervisor	

It is often better to specify a role rather than a particular user name so that if a user leaves the company or is promoted, it is not necessary to amend the procedure as the work items will automatically be sent to the new user who holds that role. For information on how to assign roles to users, see “Managing Roles” in *TIBCO iProcess Workspace (Windows) Manager’s Guide*.

Fields

You can incorporate a field into your procedure so that the name of a user, group or role can be entered into the field at run time. That field can then be used as an addressee in subsequent steps to decide the routing of the business process. As for users and roles, multiple fields can be entered in the **Fields** column.



This is called dynamic routing, or variable addressees, as it gives you the flexibility to route a step based on case data instead of using hard-coded addressees.



A field can contain a comma-separated list of addressees.

Defining Step Status Options

The following sections describe the step status options you can configure for each step in your procedure:

- [Enabling Steps to be Forwarded on page 13](#)
- [Enabling the Contents of a Form to be Copied on page 13](#)
- [Stop Work Items Being Deleted on Withdraw on page 14](#)
- [Preventing a Case from Being Suspended on page 14.](#)
- [Using Form Commands on page 34.](#)

Information about setting the step priority is described in “Setting Priority at Step Level” in the *TIBCO iProcess Modeler - Advanced Design* guide.

Enabling Steps to be Forwarded

If you want to enable users to forward a step to other users, you need to enable the Forward permission for that step:

1. Right-click the Step and click **Status**.
2. In the Permissions group box, click **Forward**. Click **OK**.

When the **Forward** permission is checked, the user who receives this work item in their queue may forward it to another user. See “Forwarding Work Items from a Queue” in *TIBCO iProcess Workspace (Windows) User’s Guide* and *TIBCO iProcess Workspace (Windows) Manager’s Guide* for more information on forwarding work items, and the QSUPERVISOR and USERFLAG attributes that need to be used to define the correct user permissions.

Enabling the Contents of a Form to be Copied

If you want to enable users to copy the entire contents of a form for a work item they receive, you need to enable the Edit permission for the step:

1. Right-click the Step and click **Status**.
2. In the Permissions group box, click **Edit**. Click **OK**.

Users can then copy the form contents (including field data) by choosing **Copy All** from the Form **Edit** menu.

Stop Work Items Being Deleted on Withdraw

Select the **Don't delete work items on withdraw** option. If this option is selected, and the deadline on an outstanding step expires or it is withdrawn as an action (release or deadline expire) of another step:

- the deadline actions are processed.
- the step remains outstanding (the step remains in the work queue or the sub-procedure case is not purged).
- when the step is released (or the sub-procedure case completes) the normal release actions are not processed but the case field data associated with the release step (e.g. the field values set in a normal step whilst in a work queue or the output parameters of a sub-case) is applied to the main case data.

Preventing a Case from Being Suspended

The **Ignore Case Suspend** check box determines whether or not the step is still processed when a case is suspended by an iProcess Objects or SAL application:

- If **Ignore Case Suspend** is not checked (the default option), the step is *not processed* while the case is suspended. This means that:
 - work items generated by the step are marked as unavailable and cannot be opened (until the case is re-activated).
 - deadlines on work items generated by the step are not processed. The date and time at which deadlines are due is not affected, and deadlines continue to expire. However, no actions are processed when a deadline expires. When the case is re-activated, any expired deadlines are immediately processed.
- If **Ignore Case Suspend** is checked, the step is *still processed* as normal while the case is suspended. This means that:
 - work items generated by the step can still be opened.
 - deadlines on work items generated by the step are still processed.

Cases can only be suspended and re-activated from an iProcess Objects or SAL application. Audit trail messages indicate whether a case is active or suspended. Refer to the iProcess Objects documentation for more information about suspending cases.

Chapter 2 **Creating Fields and Forms**

This chapter describes how to use and create fields in iProcess and then use the Process Step Definer to create forms.

Topics

- [About iProcess Fields, page 16](#)
- [Defining Fields, page 18](#)
- [Creating a Form, page 21](#)

About iProcess Fields

Fields are used in iProcess to store business data related to a case of a procedure such as customer names, order values and stock items. The data can be manipulated by iProcess as the case progresses - for example, adding fields together to get a total price of an order or concatenating the contents of a surname field and a forename field.

There are a number of iProcess system fields already defined (such as SW_STARTER and SW_CASEDESC) but you need to create fields that are relevant to the information you need to capture in your process. For example, if you are capturing order details, you will need to create fields for the customer's name, account number, delivery address, order value, and so on.

These fields can be added to a form which the user completes when they receive the work item in their work queue. After releasing the work item, the data stored in the field is saved.

There are two types of field you can use: *single instance* or *array*. A single instance field is a field that contains just one element of data - for example, a name. An array field can contain multiple elements of data - for example, 10 names.

What are Single Instance Fields?

For the majority of fields you create, you will use single instance fields. This means that just one data element will be stored in the field. For example, in a field that you have defined called CUSTNAME, a name of Jane Doe could be stored in one case. You can use expressions and functions to manipulate field data, for example, concatenating data or summing totals.

What are Array Fields?

Array fields are defined in the same way as single instance fields but they enable you to store more than one piece of data in them. For example, an array field called CUSTNAME could store the names of several customers instead of just one.

When using dynamic sub-procedure calls and graft steps, you need to use array fields to provide the multiple data elements required for each sub-case that will be started. For example, you might have a procedure in which a separate sub-procedure needs to be run for each type of account the customer has. During a case of the procedure, the customer indicates that they have three accounts so the ACCOUNT array field will be populated with the names of the three accounts.

If a dynamic sub-procedure call has been defined so that one sub-procedure is run for each account that a user holds, three sub-cases will be started. Each sub-case is passed a unique account number from the ACCOUNT array field.

Refer to “Using Array Fields” in the *TIBCO iProcess Modeler - Advanced Design* guide for more detailed information about array fields.

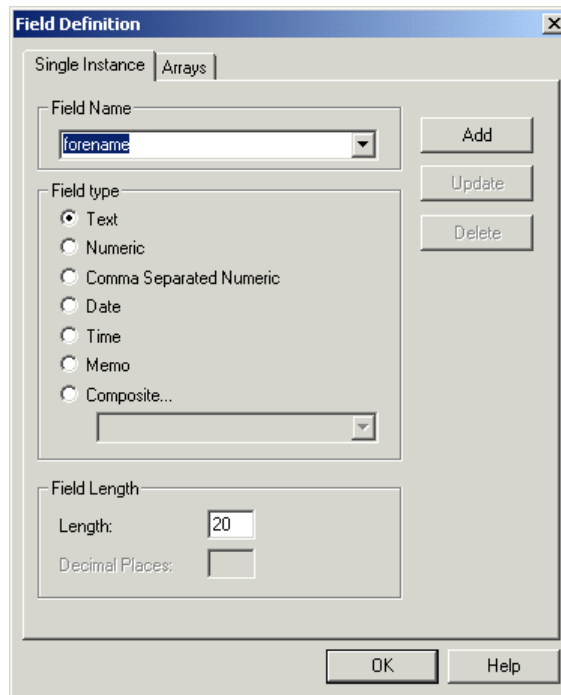
Defining Fields

A field must be defined before you can use it in forms, scripts, EAI steps, etc. You can either define your fields from the TIBCO iProcess Modeler before creating individual steps in your procedure, or you can create fields as you need them when creating forms. To define a field:

1. Click **Field > Definition** in the TIBCO iProcess Modeler (or **Field > Define** from the Process Step Definer).

The **Field Definition** dialog is displayed showing the **Single Instance** tab.

To define single instance fields, continue to step 2. To define array fields, click the **Array** tab before continuing to step 2.



The image shows the 'Field Definition' dialog box with the 'Single Instance' tab selected. The 'Field Name' dropdown menu contains the text 'forename'. The 'Field type' section has radio buttons for 'Text' (selected), 'Numeric', 'Comma Separated Numeric', 'Date', 'Time', 'Memo', and 'Composite...'. Below these is a small dropdown menu. The 'Field Length' section has a 'Length' input field with the value '20' and a 'Decimal Places' input field. On the right side of the dialog are buttons for 'Add', 'Update', and 'Delete'. At the bottom are 'OK' and 'Help' buttons.

2. In the **Field Name** box, enter a name for the field. Please note the following:
 - Field names can be up to 15 characters long and can contain letters, digits and underscore characters but they must start with a letter. Field names are converted to uppercase irrespective of how they are originally entered.
 - Array field names can be up to 8 characters. When you set the length value this determines the length for each array element. Refer to “Using Array

Fields” in the *TIBCO iProcess Modeler - Advanced Design* guide for more information about using arrays.

- Do not create a field with the same name as a system field (e.g. SW_CASEREF). iProcess does not allow you to select the new field or use it in your form as it will always default to the original system field.



It is not possible to change an existing field between **single instance** and **array**.

3. Select a **Field Type**. The available options are:

Field Type	Description
Text	Any characters can be entered up to the length you select (maximum 255 characters).
Numeric	Any number, positive or negative, up to the length and number of decimals you select (maximum length 18 including decimals, decimal point and sign, with no more than 8 decimals).
Comma Separated Numeric	As numeric but iProcess automatically displays commas to separate the thousands, for example 1,234,567.89.
Date	A date, defaulting to the format dd/mm/yyyy although this can be changed by your iProcess Administrator. <i>CAUTION - You must make changes to the date format before cases are started. Doing so while the system has live case data will corrupt the data.</i>
Time	A time (24 hour clock) in the format hh:mm
Memo	Large amounts of text (stored in a separate file).

Field Type	Description
Composite	The field refers to an iProcess Table, which is selected in the Tables box. (Creating a composite field creates an instance of that table's record definition within the procedure, from which table sub-fields can be accessed.)

- 4. If you want to change the field length, click in the **Length** box at the bottom of the window.
- 5. Click **Add**. The field details you entered are saved and you can define another field.



The length of a numeric field includes the decimal point and decimal places, for example, a length of 8 with 2 decimal places gives us 00000.00.

- 6. Click **OK** when you have finished adding fields.

Creating a Form

You can find detailed information on how to create a basic form in the *TIBCO iProcess Modeler - Getting Started* guide.

This section looks in more detail at the functionality available to you as the form designer, including:

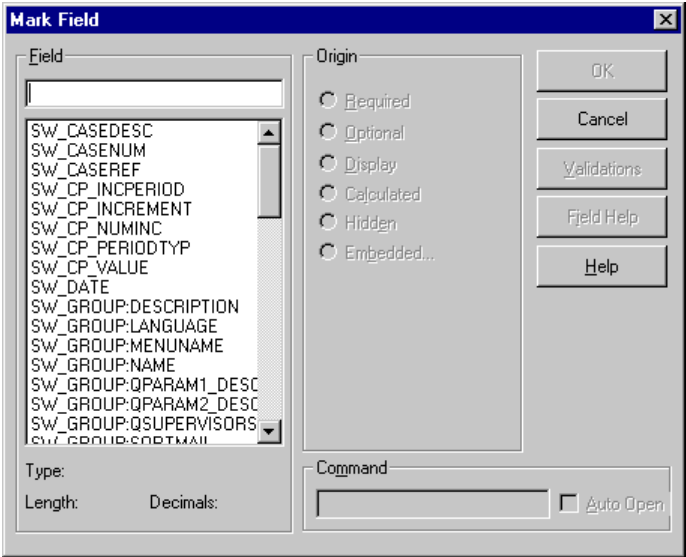
- [inserting fields into forms](#)
- [editing marked fields](#)
- [defining field help](#)
- [using embedded and ampersanded fields](#)
- [application fields](#)
- [removing fields](#) from forms
- [using conditional text](#)
- [inserting use files](#)
- [using form commands](#)
- [editing forms](#)

Inserting Fields into Forms

To insert a field in a form:

1. In the Step Definer, place the cursor where you want the field to appear.
2. Click **Field > Insert**.

The **Mark Field** dialog is displayed.



- 3. Select the field type you want to use from the **Field** list.
- 4. Select the **Origin** of the field. The origin defines how the field is going to be used in the form. The origins are:

Origin	Meaning
Required	The user must fill in this field. If a Required field is not completed the form cannot be released. In TIBCO iProcess Workspace (Windows) required fields appear on the form in red.
Optional	The user can enter information into this field but does not have to. The form can be released without an optional field being filled in. Optional fields appear in blue (white at run time).
Display	The current value for the field is displayed in the form but it cannot be changed.
Calculated	A value for the field is calculated and displayed based on a specified calculation.

Origin	Meaning
Hidden	A value is calculated but not displayed in the form.
Embedded	Displays the value of the field (but not any remaining spaces up to the length of the field). This origin is often used in letters as at run time it appears as part of the form body rather than a field. See Using Embedded and Ampersanded Fields on page 24 .

5. Depending on the field origin you have selected, click the following buttons:
 - **Validations**, which allows you to limit the data the user can input for Required or Optional fields. See [Using Field Validations on page 38](#).
 - **Calculations**, which allows you to define the calculation used to calculate the value of a Calculated or Hidden field. See [Using Field Calculations on page 41](#).
 - **Field Help**, which allows you to define help text that the user can display to help them fill in the field. See [Defining Field Help on page 23](#).
6. If you want to define a command that will run when the field is opened, enter it in the **Command** section.
 If you also click the **Auto Open** check box, the command is run automatically when the user presses ENTER or moves off the field after changing its value.
7. Click **OK** when you have finished. The field is displayed at the cursor position.

Editing Marked Fields

To edit a marked field from the Step Definer, do the following:

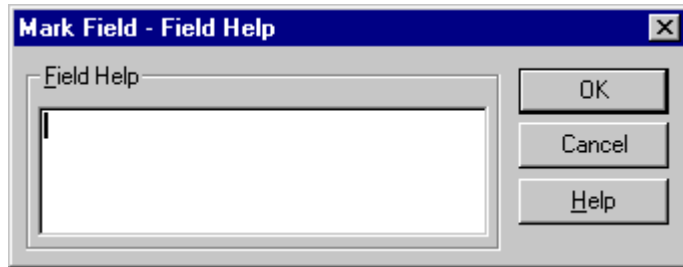
1. Click **Field > Modify** or double-click the field box.
 The **Mark Field** dialog is displayed.
2. Make any changes required and click **OK**.

Defining Field Help

When you add a field that is either **Required** or **Optional** you can enter help text that the user can display for assistance when they are filling in the field.

1. From the Process Step Definer, click **Field > Insert** (or **Field > Modify** if you have already inserted your field).
2. In the **Mark Field** dialog, click **Field Help**.

The **Mark Field - Field Help** dialog is displayed.



3. Enter your Help text and click **OK**.

The field is displayed on the form with a Help button  at the end of it. At run time, the user can click this button to display the Help text.

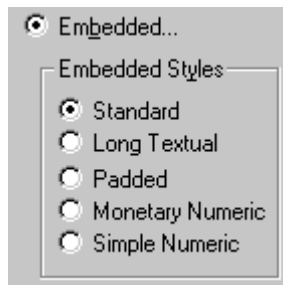
Using Embedded and Ampersanded Fields

Within your procedure you might want to include a memo to a colleague, or a letter to a customer or supplier. You might want to include some of the information that is held in the fields of your procedure, but you don't want it to appear as a form. In this circumstance you can use embedded fields.

To use embedded fields in your form:

1. In the Step Definer, click **Field > Insert**.
2. Select the field and choose an origin of **Embedded**.

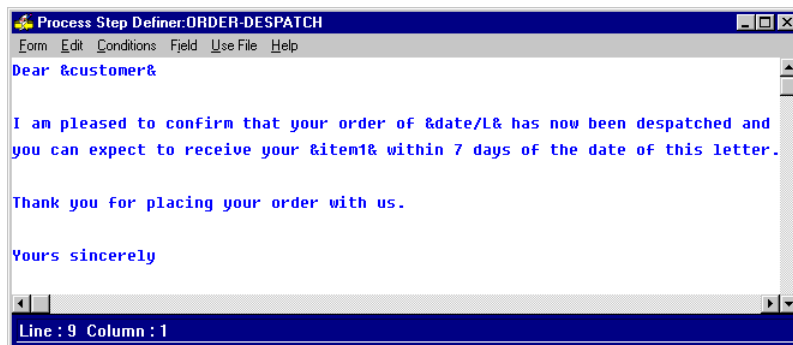
On selecting **Embedded**, an additional pop-up allows you to select the style option.



The options available to you will depend on the field type:

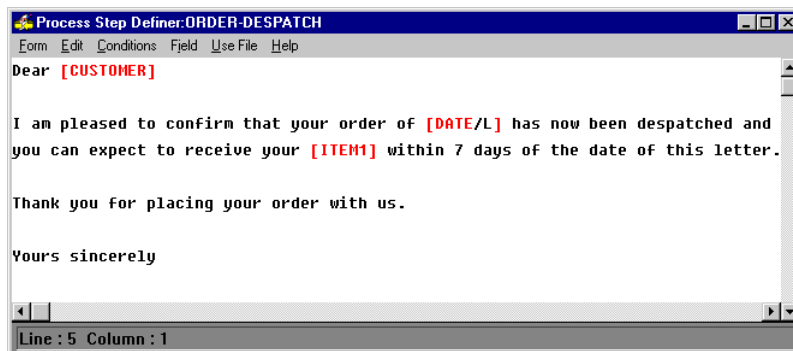
Field Type	Description
Standard	All types: appears like Display but without padding or justification.
Long Textual: /L	The following types are available: - Numeric: 120 is shown as one hundred and twenty - Date: 31/03/2000 is shown as 31st March, 2000 - Time: 23:30 is shown as 11:30 pm
Padded: /P	The following types are available: - Text: Left justified - Numeric: Right justified
Monetary Numeric: /M	Numeric only: 12.34 is shown as twelve pounds and thirty four pence (or with different currency units according to your system configuration) See Changing the Currency Unit on page 127 for more information.
Simple Numeric: /N	Numeric only: 120 is shown as one two zero

As an alternative to marking an embedded field you can type the field name within ampersands (&). You can use the style options by typing / and the style abbreviation, for example /P for padded.

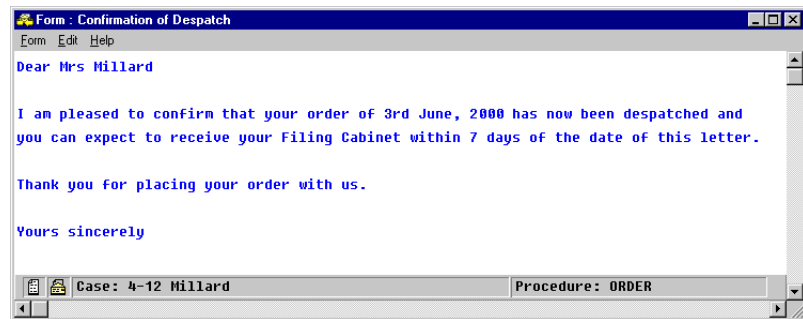


In the example above we have three ampersanded fields, customer, date and item1.

When you re-open the form after saving it you will see that iProcess has converted any field names you typed in to red field names in square brackets. This is how embedded fields appear.



At run time, the form will appear as follows:



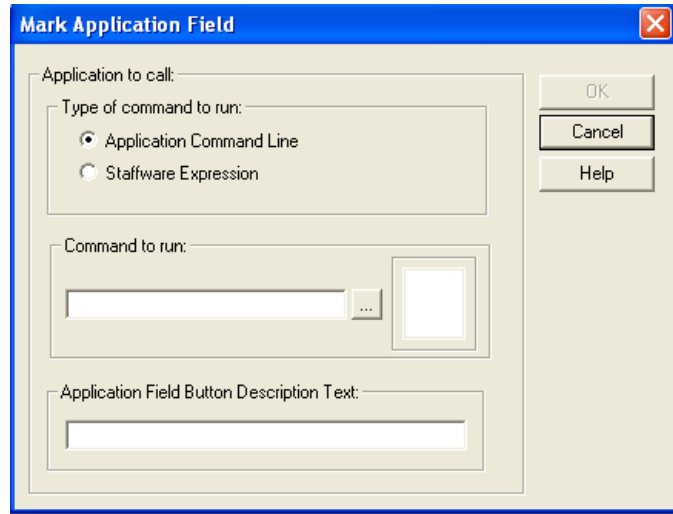
In this example, the fields Customer, Date and Item1 contain the values of Mrs Millard, 3rd June 2001 and Filing Cabinet, which all appear as if they have been typed directly into the letter.

Marking Application Fields

You use **Application Fields** when you want the user to be able to launch a program from within a form. When you mark an **Application Field**, you specify the program to be run and the text that will appear on the button. When the user clicks the button, the program is run.

1. On your form, place the cursor where you want the **Application Field** to be displayed.
2. Click **Field > Insert Application Field**.

The **Mark Application Field** dialog is displayed.



3. In the **Type of command to run** section, choose:
 - **Application Command Line**, if you want to run a program.
 - **iProcess Expression**, if you want to run an iProcess expression or script.
4. In the **Command to run** dialog, enter the name of the program, iProcess expression or iProcess script that you want to run.

If you are entering the name of a program, you can:

- Enter a full pathname. The icon that will appear on the button is displayed to the right of the pathname.
- Enter a simple filename. The program must exist on the Windows search path.
- Insert an iProcess field value in the command, by entering the field name in ampersand (&) characters. For example, the command **Notepad.exe**

&myfile& would, at run time, open the file defined by the value of the MYFILE field.

- Click  to select a program by browsing through your folders.

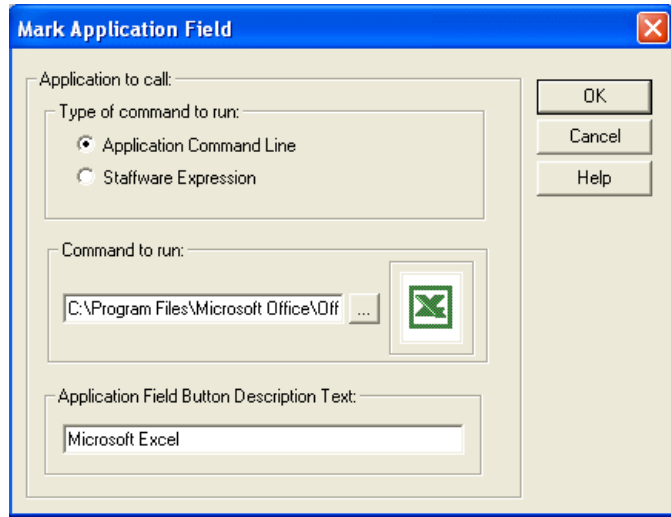
This will open the **Application Field Command** window that enables you to browse through your computer's file structure to locate the program you want to run.



When you have located the program you want to run, click **Open** and you are returned to the **Mark Application Field** dialog.

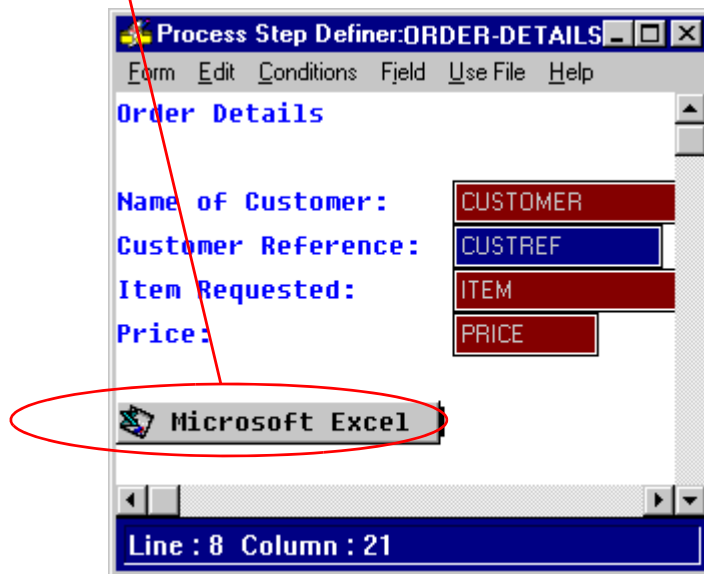
The path to the program you want to run is displayed in the **Command to run** dialog.

5. You can now enter any **Description Text** you want displayed on the application field button so that the user knows which program will be run.



6. Click OK. The **Application Field** button now appears on your form.

Application Field Button



Removing Fields From Forms

To remove a field from a form, click on the field and press **Delete**.



This does not delete the field from the list of defined fields. It only deletes this particular field marking. If you want to remove a field from the field definition list, you need to open the **Field Definition** dialog and delete it from there - see [Inserting Fields into Forms on page 21](#) for information about the field definition dialog.

Using Conditional Text to Dynamically Change a Form

You can change how the form appears to the user dynamically, based on the information that is entered into the fields. For example, on a loan application form, if the salary of the first applicant is not sufficient for the mortgage requested, then a second applicant's details must be entered. At run time, the section asking for the second applicant details will only be visible if the condition equates to true.

To enter a text condition on your form:

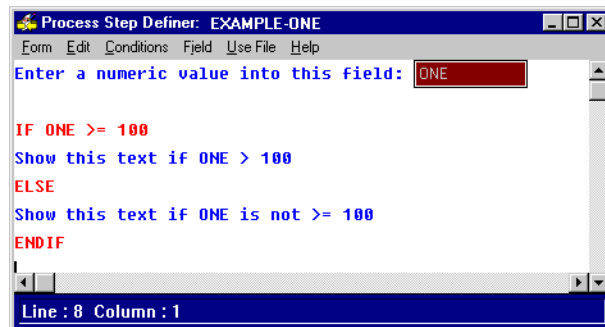
1. Click **Conditions > Insert** menu, then click **IF**.
2. Enter the condition and the label in the dialog box and click **OK**.

Any valid expression can be entered. For more information about expressions, please see “Using Expressions” in *TIBCO iProcess Expressions and Functions Reference Guide*.

3. Enter into the form all of the text and fields that you want the user to see if the expression evaluates to true.



4. You can also enter an **ELSE** from the **Conditions** menu if you want to show alternative text and fields to the user if the condition evaluates to false. The construct would then look like this:



5. Finally, you must close the condition by choosing **ENDIF** from the **Conditions** menu.

You can nest IF.....ENDIF or IF.....ELSE.....ENDIF conditions one inside the other up to a maximum of 20 deep.

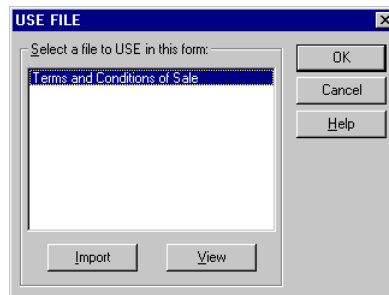
As nested conditions can become quite complex to follow, you can display the nesting levels at the side of the Step Definer form. See [Nesting Level on page 122](#) for instructions on how to do this.

Inserting a Use File

Use files are text files that reside in a specific area on the TIBCO iProcess Engine and can be read into a form of a procedure whenever and wherever they are required. This means that the text can be created once in a word processor and read into as many forms as you like. It also means that the text only needs to be updated once and the latest version will always be used in the forms which makes the maintenance of procedures much easier and more cost effective.

To insert a **Use File**, do the following:

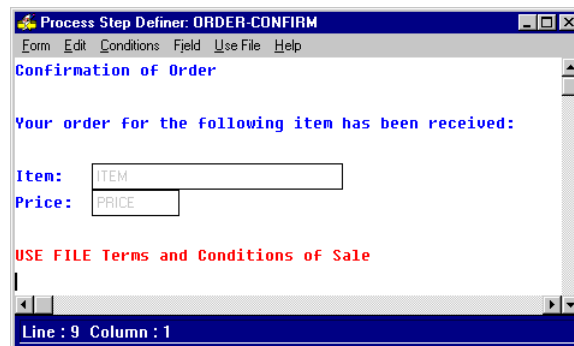
1. From the Step Definer, click **Use File > Insert**.
2. Choose the file you want from the dialog and click **OK**.



To add another file to the list displayed, you need to **Import** it. Click **Import** and you can browse through your directory structure for the file you want.

If you want to look at the contents of a file before inserting it into your form, highlight the file in the list box and click **View**.

When you have inserted your Use file, it will appear in your form as follows:



At run time the contents of the file are displayed.



Use Files must be ASCII (plain text), can be up to 250 lines in length, and you can only show fields in the text by using Ampersanded Fields. See [Using Embedded and Ampersanded Fields on page 24](#).

Using Form Commands

Form Commands are instructions to iProcess to execute a specific command at a given point in the procedure. As indicated by their name, Form Commands are attached to the form and there are three kinds.

Command Type	Description
Initial	This command is run when the work item form is opened from the user's Work Queue.
Keep	This command is run when the form is returned to the user's Work Queue.
Release	This command is run when the form is released.

You define a Form Command in the **Step Status** dialog.

Select the Step that you want to put a form command on and choose **Status** from the **Step Definition** dialog.

The screenshot shows the 'Step Definition' dialog box with the 'Status' tab selected. The 'Form Commands' section contains three text boxes: 'Initial:', 'Keep:', and 'Release:'. The 'Release:' text box is circled in red and contains the text 'winrun("c:\program files\microsoft office\office\winrun",3)'. Below this, the 'Step Priority' section has a 'Base Priority Value (1-999):' field with 'SW_CP_VALUE'. The 'Automatic Priority Escalation' section has four fields: 'Increment (negative to raise):' with 'SW_CP_INCREMENT', 'Number of increments (-1=unlimited):' with 'SW_CP_NUMINC', 'Increment Period:' with 'SW_CP_INCPERIOD', and 'Period Type ("m"ins, "h"ours, "d"ays):' with 'SW_CP_PERIODTYP'. At the bottom, the 'Permissions' section has three checkboxes: 'Forward', 'Edit', and 'Ignore Case Suspend'. The 'OK', 'Cancel', and 'Help' buttons are at the bottom right.

This example has a Release Command specified that calls a third-party application. An iProcess Command can be any iProcess expression, but will usually be one of the following:

- A function call to run an **external program** (as above), such as SERVERRUN to run a program on the Server, or WINRUN to run a program on the iProcess Workspace. See “Functions to Call External Programs” in *TIBCO iProcess Expressions and Functions Reference Guide* for more information.
- A call to an iProcess **script** in the format **CALL (script1)**. See “Calling Scripts” in the *TIBCO iProcess Modeler - Advanced Design* guide and “CALL” in *TIBCO iProcess Expressions and Functions Reference Guide* for more information.
- An **assignment** expression to give a new value to a field, for example:
FIELD2 := SUBSTR (FIELD1, 1, 2)

This would assign part of FIELD1 to FIELD2 when the command is run. See “Using Expressions” in *TIBCO iProcess Expressions and Functions Reference Guide* for further information on assignment expressions.

Editing Your Form

You can edit text in a form by using the standard **Cut**, **Copy** and **Paste** features. The selected text can include fields and can be copied either within a form or to another form window. If you copy the selected text to another application such as a word processor, any fields will be converted to fieldnames.

Using the **Copy All** option from the **Edit** menu enables you to copy the entire form so that you can paste it into another form either as part of the existing procedure or a different procedure.



You can copy and paste forms between different procedures. If fields marked in the source form do not already exist in the destination procedure they are automatically created. If fields already exist but have conflicting types, a warning dialog is displayed and the field will **not** be marked in the copy.

Changing the Form Type of a Step

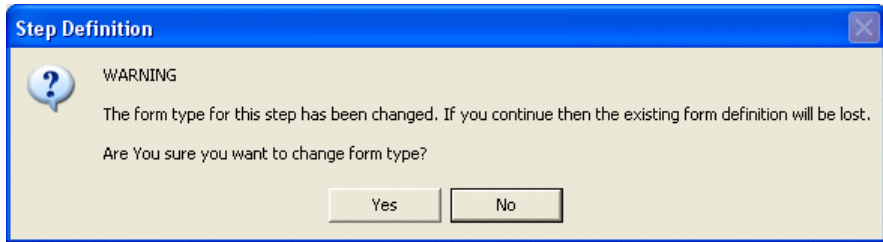


If you change the form type of an existing step from **iProcess Form** to **Formflow form (FORMFLOW)** or vice versa, then the existing form definition is lost.

To change the form type of a step:

1. Right-click on the step and click **Properties**. The **Step Definition** dialog is displayed.

2. From the **Form Type** drop-down list, select a new form type and click **OK** or **Edit**, depending on your requirements. The following warning is displayed:



Click **Yes** to save the step with a new form type or **No** to return to the **Step Definition** dialog.

Chapter 3

Controlling Data Input On Forms

This chapter explains how to control data input on your forms.

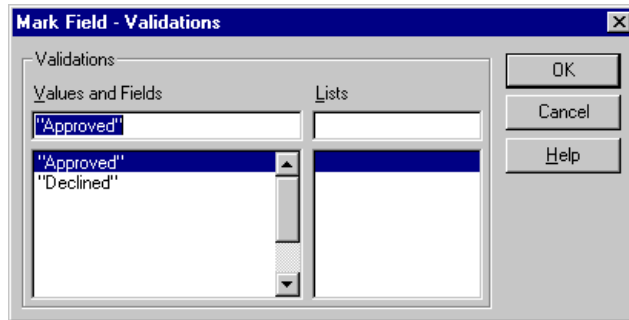
Topics

- [Using Field Validations, page 38](#)
- [Using Field Calculations, page 41](#)
- [Using Delimiters and Key Words, page 46](#)
- [Using iProcess Tables, page 48](#)

Using Field Validations

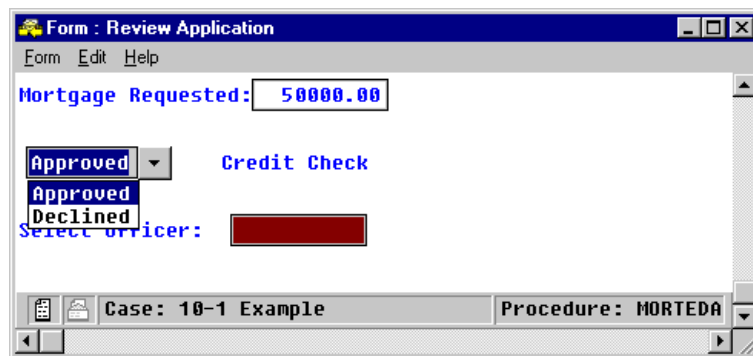
When you define a field that is either **Required** or **Optional**, you can enter validations so that the information the user enters is limited to certain options that are selected from a drop-down list.

From the **Mark Field** dialog, choose **Validations**.



In the example above, two possible values have been entered, “Approved” and “Declined”.

When a user receives this step in their queue, the form will look like this:



The button next to the field enables you to open and close the drop-down list. Use the arrow keys to move up and down the list and when the choice you want is highlighted, press **Return** to select it.

As well as entering text values (which must be entered in double quotes) in the **Values and Fields** column, you can also enter iProcess Expressions or Field names. Each must equate to the same type as the field, for example, Text or Numeric. VLDFILE and VLDQUERY functions can be used to add items from an external database to the list. Special values such as SW_ANYTHING or SW_BLANK can also be used. See *TIBCO iProcess Expressions and Functions Reference Guide* for more information.

In the **Lists** column you can enter the name of an iProcess list. If you have imported the sample procedures supplied with iProcess, the list **YESNO** will have been defined. Entering this in the **Lists** column will give you a drop-down list at run time with the values of **YES** and **NO**. Other lists may be defined through the **List Manager** in the **iProcess Administrator**.



Note that the list **YESNO** will not be present unless you have imported a procedure that uses it.

If you add a new list or make any changes to existing lists in the TIBCO iProcess Administrator, you need to log out of iProcess and log back in again before the changes you have made will be visible in the TIBCO iProcess Modeler.

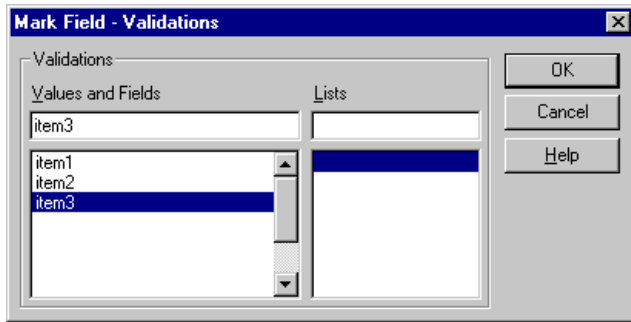
Each column can contain up to 10 items. If there are entries in both columns, they will all be displayed in the drop-down list.



If the field that has the validation on it is a composite key field then by entering the word **TABLE** in the lists section, a drop-down list of the tables key field values is displayed at run time. See [Using iProcess Tables on page 48](#) for more information.

Copying Field Values

You can copy the value of one field to another by entering the name of a field in the **Values and Fields** column of the **Validations** dialog. The value of the field appears in a drop-down list at run time. If you enter more than one field, you can see the values of all of them in the drop-down list and can select from them.



Using Field Calculations

You can use calculations to determine the value of a field. For example, in an ordering procedure you might want to calculate the value of all the items ordered. Or, you might need to calculate the delivery date of an item based on the order date and the known lead time for that item.

Field calculations are carried out on opening the form and every time a dependent field is changed.



Self-referential calculated or hidden fields (for example, **field1=field1+1**) can be evaluated multiple times. This is because forms are rebuilt several times as a work item is processed and the calculation is evaluated every time the form is rebuilt. Therefore, using the example above, this means that **2** could be added to **field1**. To overcome this, use self-referential calculated fields in an initial or release command. This way the calculation is only executed on an initial or release of the form.

A calculated or hidden field has the option of **Calculations** instead of **Validations**.

The calculation is entered in the left column and you can optionally enter a condition so that the calculation is only carried out if the condition is met, see [Conditional Calculations on page 43](#).

The following table shows you the operations that can be carried out by iProcess calculations.

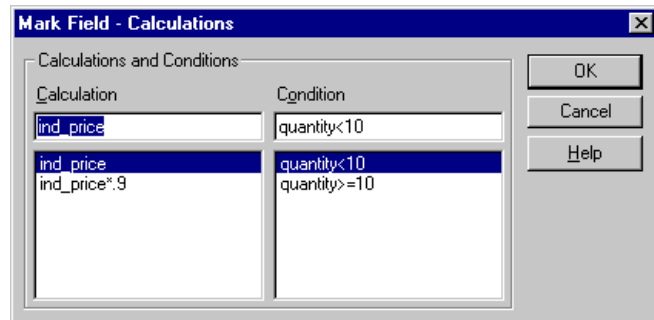
Operation	Definition
+	add (when adding two text fields, the values are concatenated).
-	subtract

Operation	Definition
*	multiply
/	divide
^ or **	to the power of
date	To calculate the number of days between two dates: DATE1 - DATE2 The following formula can be used in date calculations: @days/weeks/months/years@ For example, to add one year to the DATE1 field: DATE1 + @0/0/0/1@
time	To calculate the number of minutes between two times: TIME1 - TIME2 For example, to add 15 minutes to the TIME1 field: TIME1 + 15
Comparison operators	Compare the value of one field to another
=	equals
<>	does not equal. For example: DATE1 <> DATE2
<	less than. For example: FIELD1 < FIELD2
>	greater than
<=	less than or equal to
>=	greater than or equal to
()	parentheses

Operation	Definition
AND <i>or</i> OR	logical operators. For example: (FIELD1 < FIELD2) AND (DATE1 <> DATE2)

Conditional Calculations

Conditional calculations are ones that are only carried out if a particular condition is met. The conditions are entered in the right side of the **Mark Field - Calculations** dialog. In the following example, the first calculation is only performed if the value of the field **Quantity** is less than 10; if it is greater than or equal to 10, then the second calculation is performed.



The operators that you can use are:

Operator	Description
=	equal to
<>	not equal to
<	less than
>	greater than
<=	less than or equal to
>=	greater than or equal to

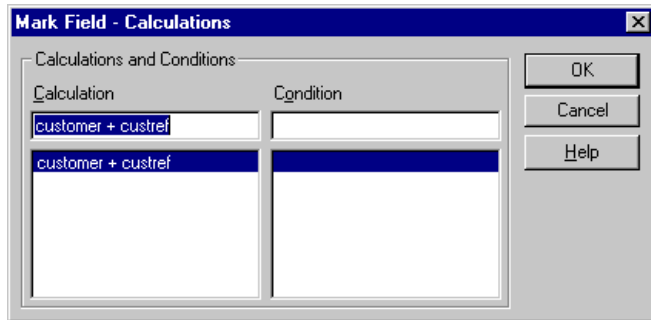
Up to five calculations with conditions may be entered. iProcess looks at each in turn and acts upon the first condition that is met. Any remaining calculations and conditions are disregarded. The final calculation can be left without a condition so that if the first 4 conditions fail, the final calculation will be carried out.

If there are no conditions, then only one calculation should be entered.

Calculating Text Fields

It is not just numeric fields that can be calculated: text fields can also be calculated.

The example below shows how a field calculation can be the sum of two other fields.



In the form at run time this gives the customer name followed by the customer reference. For example:

Text calculation:	Ms L McGuinness 099476
-------------------	------------------------

Calculating the Case Description

This method of calculation can also be used to calculate fields such as `sw_casedesc`, the case description.

To calculate the case description rather than entering it when you start a case, do the following:

1. Click **Procedure > Status**.
2. Click **Hidden for Case Description at Start**.

iProcess does not allow a case description to be entered at case start when **Hidden** is selected.

3. In the first form of the procedure use **Field > Insert** to place the **sw_casedesc** field on the form (probably as a hidden field) and then add a calculation as described above.

The calculated case description is displayed for each subsequent work item in the work queues. It will not initially be available to the first step as the iProcess Engine will not be updated with the case description until the form is either kept or released.

Using Delimiters and Key Words

If you want to compare a field value with a real value, you must enclose the real value within delimiters appropriate to the field type as shown below:

Field Type	Delimiters	Example
Numeric	None	NUM = 2
Text	“ ” Quotes	Name = “Fred”
Date	!! Exclamation marks	Effective_Date>!07/10/99!
Time	# # Hashes	Time = #08:00#

Key words, or system values, are special iProcess words that have particular values and meanings. These words can be used as fields and are always present in the **Field Definition** dialog. The following table details each of the key words.

Key Word	Meaning
SW_CASEDESC	The case description of the current case as entered by the user starting the procedure or can be calculated, see Calculating Text Fields on page 44 .
SW_CASENUM	The case number of the current case, allocated sequentially by iProcess.
SW_CASEREF	The case reference number of the current case in the format x-yy, where x is the number of the procedure and yy is the number of the case.
SW_DATE	The system date.
SW_HOSTNAME	The name of the host node for the procedure.
SW_NODENAME	The name of the iProcess Engine.
SW_PRODESC	The description of the procedure (up to 24 characters).
SW_PRONAME	The name of the procedure (up to 8 characters).
SW_STEPDESC	The description of the step (up to 24 characters).

Key Word	Meaning
SW_STEPDESC2	An extra value for adding further description about the current step.
SW_STEPNAME	The name of the step (up to 8 characters).
SW_TIME	The system time on the iProcess Engine.

There are also certain key words that apply just to sub-procedures. These are:

Key Word	Meaning
SW_MAINCASE	The top level procedure's case number.
SW_MAINPROC	The top level procedure's name.
SW_MAINHOST	The host where the top level procedure resides.
SW_PARENTCASE	The parent procedure's case number.
SW_PARENTPROC	The parent procedure's name.
SW_PARENTHOST	The host where the parent procedure resides.
SW_PARENTREF	Internal information on the parent given in a text string as follows: pname^pnnum count^ccrnum^step name^step description^call depth

These fields are read-only and are added to all new sub-procedures. If a main procedure is changed to a sub-procedure, these fields will be added to the procedure by the TIBCO iProcess Modeler.

Any of these special iProcess words can be used in your forms by selecting them in the **Mark Field** dialog. Alternatively, they can be used to calculate the value of a field.

In addition, SW_GROUP, SW_USER and SW_STARTER can be used in conjunction with any of their attributes, for example, SW_STARTER:DESCRIPTION will give the description of the user who started the case of the procedure. See "Expressions" in *TIBCO iProcess Expressions and Functions Reference Guide* for more information.

Using iProcess Tables

iProcess tables can be accessed from all procedures and are easily referenced from the form. iProcess tables should only be used for static information that does not need to be updated very often. As iProcess tables are not automatically updated when new data is entered into a form, they are not suitable for uses such as a customer database.

Structurally, iProcess tables are similar to database tables. Table records are associated with fields, and one key field value serves as a unique identifier for each record. For example, a user enters a value in the key field. If this value corresponds to a record in the table, the values of the other fields in the table will automatically display.

Tables are created and stored on the iProcess Engine. For information on how to create tables, see “Managing iProcess Tables” in *TIBCO iProcess Workspace (Windows) Manager’s Guide*.

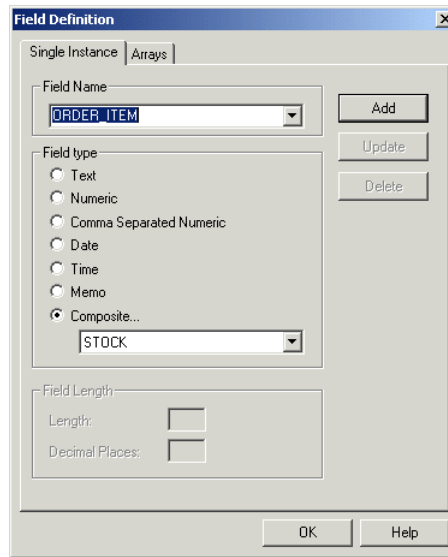
To use a table in a form, you must first [define](#) it and then [mark](#) it.

Defining a Table Field

To define a table field in your form, do the following:

1. Click **Field > Define** and enter the name you want to call your field.
2. Click **Composite** for the field type.
3. From the drop-down list of tables, choose the table you require.

4. Click **Add**.

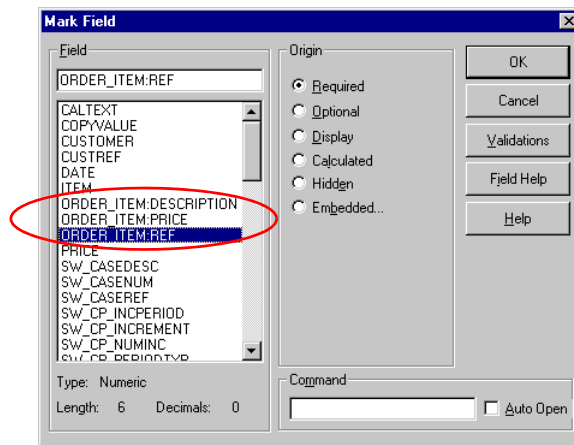


Marking Table Fields in your Form

To mark a table field in your form, do the following:

1. Click **Field > Insert** and choose the table field you want to insert.

The table field you defined will be shown in the list of fields several times, once for each field in the table, as shown below.



2. Select the field you want, choose the origin and click **OK**.

3. In your form, mark the key field in the table and then mark the remaining table fields as required.

At run time, when the key field is entered, the remaining table fields will automatically be completed with the values from the table.

Chapter 4

Using Deadlines in Procedures

Deadlines can be placed on a step to ensure that a work item is completed within a specified time period. If the deadline expires, the deadline actions will be processed. Deadlines can, if required, be reset on outstanding work items.

Topics

- [Defining a Deadline, page 52](#)
- [Using Deadlines on a Sub-Procedure, page 55](#)
- [Dynamically Recalculating Deadlines, page 56](#)

Defining a Deadline

1. From the **Step Definition** dialog, click **Deadlines**. You can select a step and right-click to access the **Deadlines** tab.

The **Deadline** tab is displayed.

Choose deadline period or expression

Enter the deadline period here.

Step Definition

Definition | Addressees | **Deadlines** | Status

Deadlines on this step are defined by:

☒ Period ☐ Expression

Deadline Period

Years: Months: Weeks:

Days: Hours: Minutes:

Action on deadline expiry

☐ Withdraw form from queue on expiry

A deadline is set under either of these conditions:

OK Cancel Help

2. Select whether your deadline is to be based on a **Period**, such as 2 weeks, or on an **Expression**, then enter the deadline period or expression.



When considering how to define a deadline, you should bear in mind whether or not the deadline may need to change after the step has been sent out. See [Dynamically Recalculating Deadlines on page 56](#) for more information.

If you select **Expression**, this section will automatically change to allow the input of the expression, as shown below.

Enter the deadline expression here.

The screenshot shows the 'Step Definition' dialog box with the 'Deadlines' tab selected. Under 'Deadlines on this step are defined by', the 'Expression' radio button is chosen. The 'Deadline Expression' section contains a 'Date Expression' field with the text 'ap_date + @0/2/0/0@' and an empty 'Time Expression' field. Below this, the 'Action on deadline expiry' section has an unchecked checkbox for 'Withdraw form from queue on expiry'. At the bottom, there are two empty text boxes under the heading 'A deadline is set under either of these conditions'. The 'OK', 'Cancel', and 'Help' buttons are at the bottom right.

In this example, a date expression has been defined as the date of application plus two weeks using the formula of *@days/weeks/months/years@*. The expression is evaluated at the time the step is sent out. For more information on expressions, see *TIBCO iProcess Expressions and Functions Reference Guide*.

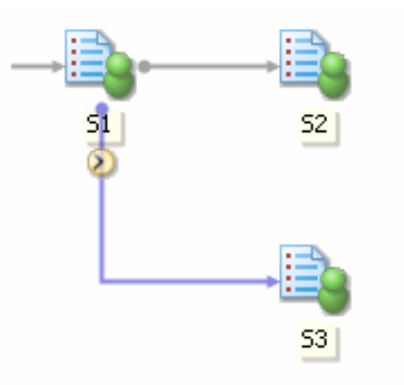
If working days are set, and you want the deadline to use them you need to be aware of the following:

- If you set a time expression or specify a date such as SW_DATE, the deadline does not take into account working days. For example, if working days are set to Monday to Friday and you are testing the procedure on a Sunday, the deadline will be set for Sunday rather than the first working day (Monday).
- If you specify a Date calculation such as SW_DATE + @0/0/0/0@, the deadline uses working days and therefore the deadline will be set to Monday.

3. (Optional) Select the **Withdraw form from queue on expiry** check box if you require the work item with the deadline to be withdrawn from the work queue when the deadline expires (unless the work item is open at the time of expiry). The deadline actions will be processed. This feature is often used in situations where an escalation process is used.
4. (Optional) Enter any deadline conditions required. You can set a deadline to only take effect if a certain condition is true. For example, a deadline could be set if the anticipated completion date for the property purchase is less than 4 weeks after the application date. This condition is evaluated when the step is sent out.

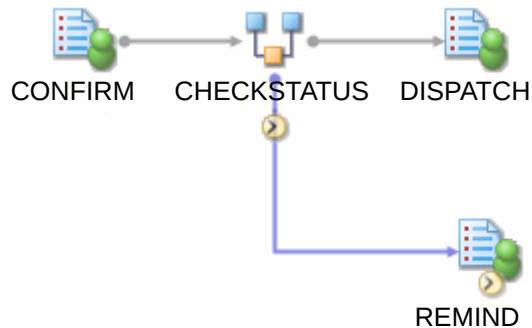
Drawing the Deadline Link

When you have created the step(s) that are to be actioned on expiry of the deadline you can draw the deadline link(s). The link must be drawn from the bottom of the first step (**S1** in the example below) to the left of the second (**S3** in the example below).



Using Deadlines on a Sub-Procedure

A deadline can be set on a sub-procedure object in the same way as any other step. Once the deadline expires, the deadline actions are performed and if **Withdraw form from queue on expiry** is selected, then the sub-case (and any child sub-cases it has initiated) will be closed. In this instance, the output data is not copied back into the parent procedure's case data.



A deadline set on a call to a sub-procedure will not be seen as a deadline in the Work Queue Manager for the sub-procedure, as the steps within a sub-procedure can have their own deadlines.

Dynamically Recalculating Deadlines

A deadline (with or without a condition) is evaluated and/or calculated when a work item is sent out. It then remains in force until either the work item is released or the deadline expires.

In some situations, however, you may want to reset a deadline on a work item while it is still outstanding. For example, you may have used a deadline to set a review date for a customer's case in 6 months' time, but then for some reason want to bring that review date forward to 3 months' time.

You can force the iProcess Engine to re-calculate its deadlines on all outstanding work items for a case by triggering an event on a particular step of the case. The event must:

1. update one or more field values used in setting the deadline - either in the expression that is used to calculate the deadline, or in an expression that is used to determine whether a deadline is set.
2. set a flag that informs the iProcess Engine that it should recalculate deadlines for the case.

To trigger such an event, you can use any of the following methods:

- the `SWDIR\bin\swutil EVENT` or `SWDIR\util\swbatch EVENT` commands. See "Issue an Event" in *TIBCO iProcess swutil and swbatch Reference Guide* for more information about these commands.
- the `TRIGGEREVENT` function. See *TIBCO iProcess Expressions and Functions Reference Guide* for a detailed description of this function.
- iProcess Objects. See the relevant iProcess Objects Programmer's Guide and help system for more information.

When the iProcess Engine detects the event, it recalculates the deadlines on all outstanding work items, as shown in the following table.

Condition Type	Expression deadline is...	Period deadline is...
No condition	Re-evaluated and set to the new value.	Not recalculated.
Condition now evaluates as true	Re-evaluated and set to the new value.	Calculated as a period from the original date that the work item was sent out.
Condition now evaluates as false	Removed.	Removed.



You cannot use an event in this way to recalculate a **Period** deadline that is not triggered by a condition, because no field values are involved in the deadline's calculation. The only way to force a re-calculation of such a deadline is to build logic into your procedure allowing you to withdraw the step and then resend it with the new deadline. However, if you do this any changes made to the work item while it has been in the user's queue will be lost.

Chapter 5

Using Conditional Actions

Within most business processes there is a point at which a decision has to be made and different paths followed depending on the outcome of the decision. For example, an application for a mortgage is approved or rejected, an item is in stock or out of stock. A conditional action is the point in the process where the decision is automatically made by iProcess according to the rules you specify.

Topics

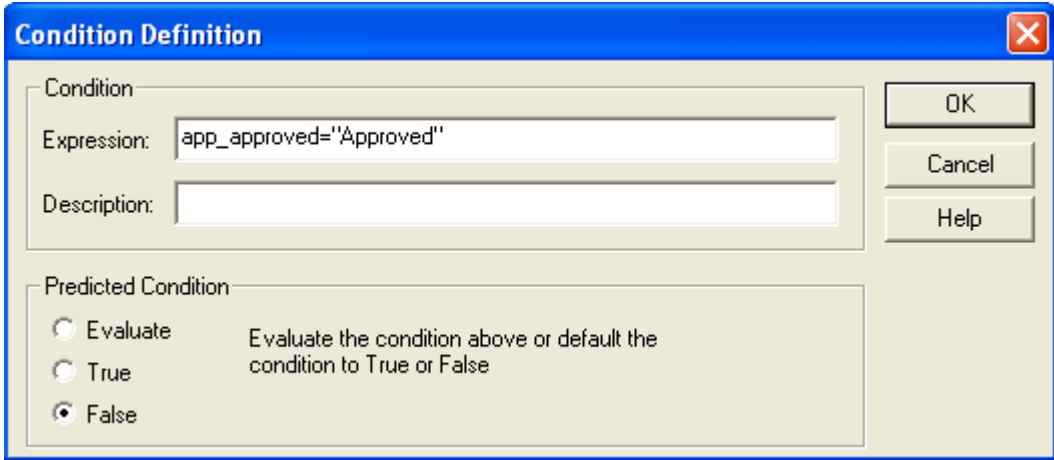
- [Defining a Conditional Action, page 60](#)

Defining a Conditional Action

To define a conditional action:

1. Click the condition object  from the Toolbar, place it on the TIBCO iProcess Modeler chart and click.

The **Condition Definition** dialog appears.



The **Condition Definition** dialog box is shown. It has a blue title bar with the text "Condition Definition" and a red close button. The dialog is divided into two main sections: "Condition" and "Predicted Condition".

In the "Condition" section, there are two text fields: "Expression:" and "Description:". The "Expression:" field contains the text "app_approved='Approved'".

In the "Predicted Condition" section, there are three radio buttons: "Evaluate", "True", and "False". The "False" radio button is selected. To the right of the radio buttons, there is a text label: "Evaluate the condition above or default the condition to True or False".

On the right side of the dialog, there are three buttons: "OK", "Cancel", and "Help".



If case prediction is enabled on your procedure, you can use the Predicted Condition settings. See “Using Case Prediction to Forecast Outstanding Work Items” in the *TIBCO iProcess Modeler - Advanced Design* guide for more information.

2. Enter the condition and click **OK**.

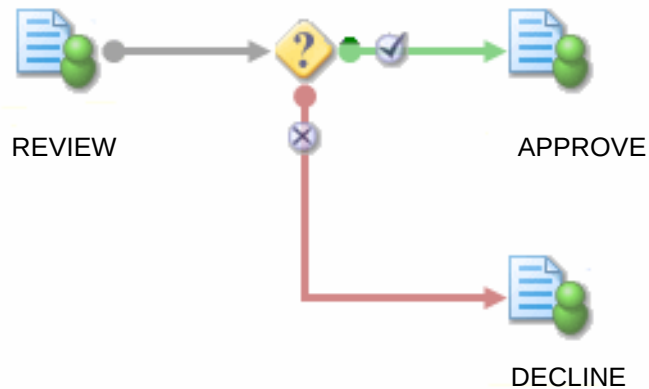
See *TIBCO iProcess Expressions and Functions Reference Guide* for more information on how to enter a valid condition expression.



Any fields that you use in this expression must already be defined.

You will also need to define the step(s) to be actioned if the condition proves true and those to be actioned if the condition proves false.

3. Draw a link from the right side of the preceding step to the left side of the condition object. Then:
 - For when the condition evaluates to True, draw a line from the right side of the condition to the left side of the next step(s) as indicated by the check mark on the object.
 - For actions to be taken when the condition evaluates to False, draw a line from the bottom of the condition to the left side of the next step(s).



A condition can have more than one action following each of its true and false branches. A condition can also be directly followed by another condition.

Chapter 6

Withdrawing Steps From the Procedure

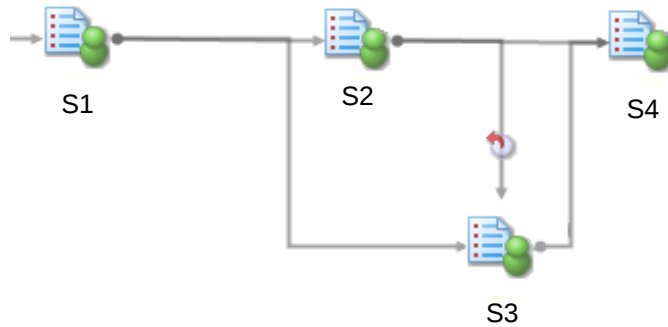
If you have steps that become redundant during the running of a case, you can define the procedure so that they are withdrawn from the work queues. You do this by defining a withdraw action on the step. An example of where this can be used is when you have two steps sent out in parallel, but if one is released, the other becomes unnecessary and can be withdrawn from the work queue.

Topics

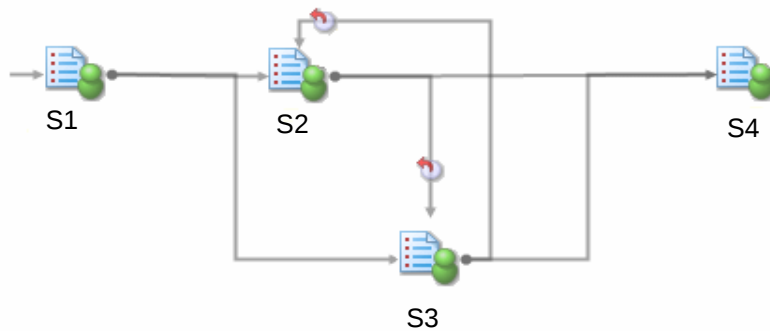
- [Example of Using a Withdraw Action, page 64](#)
- [Defining a Withdraw Action, page 65](#)
- [Defining a Deadline Withdraw Action, page 66](#)
- [Defining a Withdraw Action on a Sub-Procedure, page 67](#)

Example of Using a Withdraw Action

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

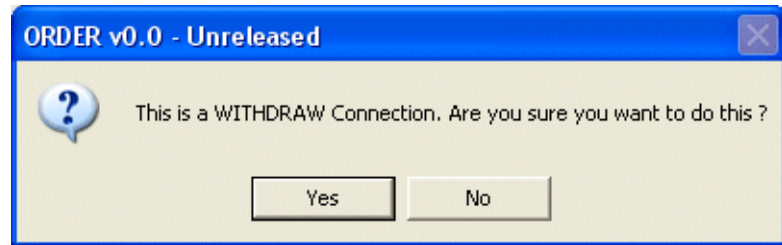


Here there are 2 withdraw actions. If **S2** is released first then **S3** will be withdrawn but if **S3** is released first then **S2** will be withdrawn.



Defining a Withdraw Action

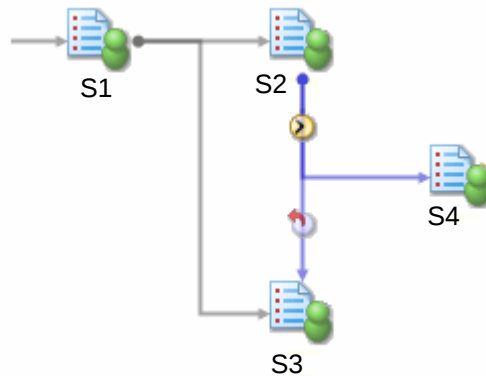
Draw a link from the right side of the step to the top of the step to be withdrawn. A message box appears asking you to confirm that you want to define a withdraw action. Click **Yes**.



Defining a Deadline Withdraw Action

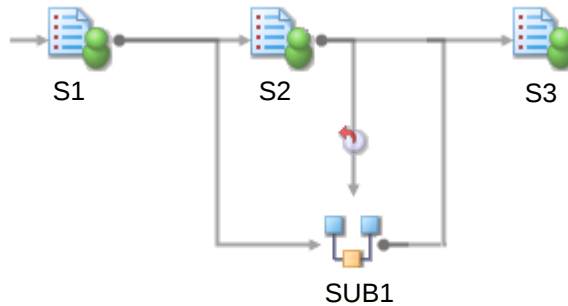
Draw a link from the bottom of the step with a deadline to the top of the step to be withdrawn. A message box appears asking you to confirm that you want to define a withdraw action. Click **Yes**.

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



Defining a Withdraw Action on a Sub-Procedure

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



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

Defining Waits in the Procedure

You can insert a wait into your procedure to pause the flow of a case until a number of steps have finished. You do this by inserting a Wait object on your procedure. You can use Waits to synchronize multiple concurrent paths within the procedure. A Wait:

- *begins* when the step which precedes it is released. (This is the step which has a line drawn from its right edge to the Wait object's left edge.)
- *finishes* when the steps which it is waiting for are released or withdrawn. (These are the steps which have lines drawn from their right edges to the top or bottom of the Wait object.)
- must be the only action following a step.

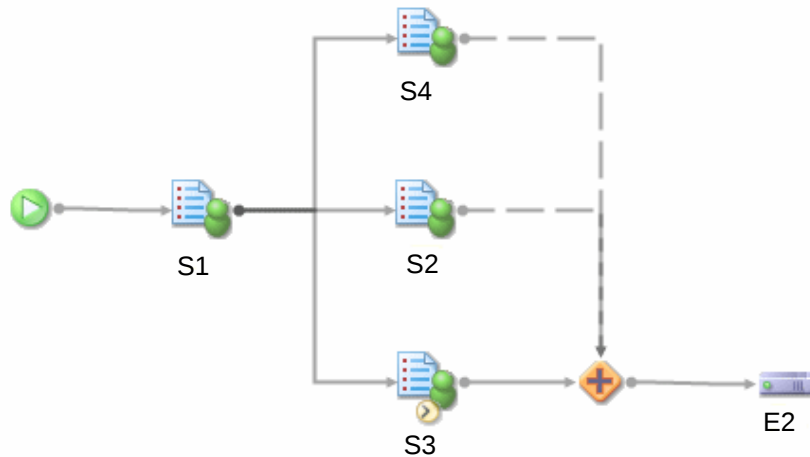
Topics

- [Example of Using a Wait, page 70](#)
- [Defining a Wait Action, page 71](#)

Example of Using a Wait

In the following example, the Wait begins when step **S3** is released, and finishes when **S2** and **S4** are also released:

1. When **S3** is released, the Wait is processed.
2. The procedure waits until **S2** and **S4** have also been released.
3. This means that **E2** will not be processed until **S2**, **S3** and **S4** have been released.



It is important to realize that the Wait is not processed until **S3** is released. If **S2** or **S4** are released before **S3**, **E2** is **not** processed, because the Wait has not yet been processed.

Defining a Wait Action

To add a wait to your procedure, do the following:

1. Click the Wait icon  from the SPD Toolbar.
2. Place the Wait object at the appropriate place on your procedure. This depends on how you require the wait action to work - see:
 - [Using Waits in Loops on page 72](#)
 - [Using Waits with Conditions on page 79](#)
 - [Using Waits with Withdraw on page 80.](#)

How the iProcess Engine Processes Waits

The iProcess Engine maintains its own record of the current status of each step in an active case. A step can be:

- **Withdrawn.** The step has been withdrawn.
- **Outstanding.** The step has been sent out to a queue (or external system).
- **Released.** The step has been released by a user (or external system).
- **Not Processed.** The step has not yet been processed by the server.

The server uses this step status value to control when Waits begin and finish.

Viewing Step Status on the TIBCO iProcess Engine

You can use the following command (on the TIBCO iProcess Engine) to view the current status of each step in each active case for a procedure:

```
SWDIR\util\plist -D [nodename] procname
```

where

- *SWDIR* is the iProcess system directory, where the TIBCO iProcess Engine is installed.
- *nodename* is the name of the iProcess Engine node (optional).
- *procname* is the name of the procedure (main procedure or sub-procedure) you want to view.

The status of each step in each active case for procedure *procname* is shown. Status is shown by one of the following characters:

Character	Status
W	Withdrawn
O	Outstanding
R	Released
.(Period)	Not Processed

The following example shows the output of this **plist** command for the procedure shown on [page 70](#):

- Step names are shown vertically.
- The case number is shown at the beginning of the final line.
- The status of each step (**R** for released or **O** for outstanding) is shown on the final line under the step name.

A single case (case number 1) is active. S1 and S3 have been released, but S2 and S4 are Outstanding (in a work queue). Note that E2 has not been processed yet.

```
C:\swserver\staffw_nod1\util>plist -D waituse3
Diractive Status Listing for procedure: WAITUSE3
Case  : $$SSSES
      PP12344
      AA
      NN
      II
      CC
      12

=====

1      :  . .ROR.O
```

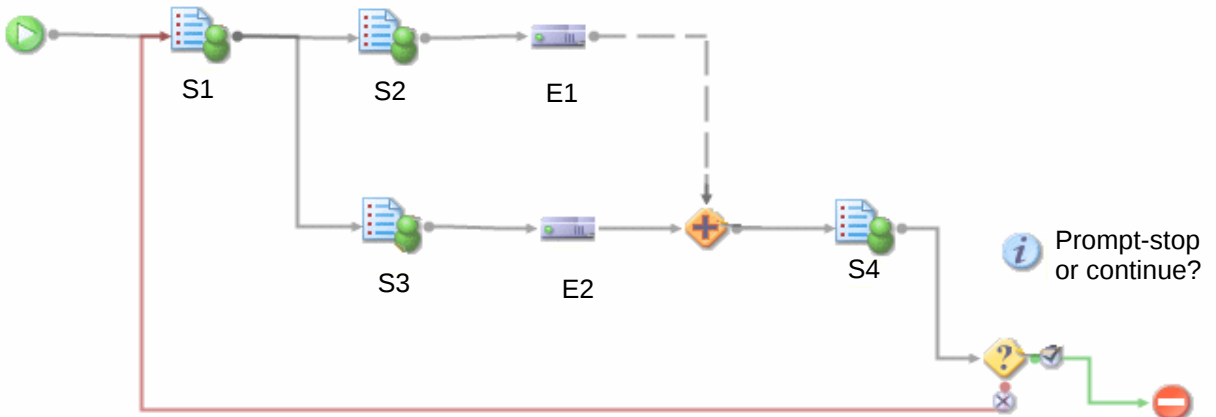
Using Waits in Loops

There are many ways of using Waits to control the flow of your procedure. For example, you can use them in loops,

Consider the example procedure shown below.

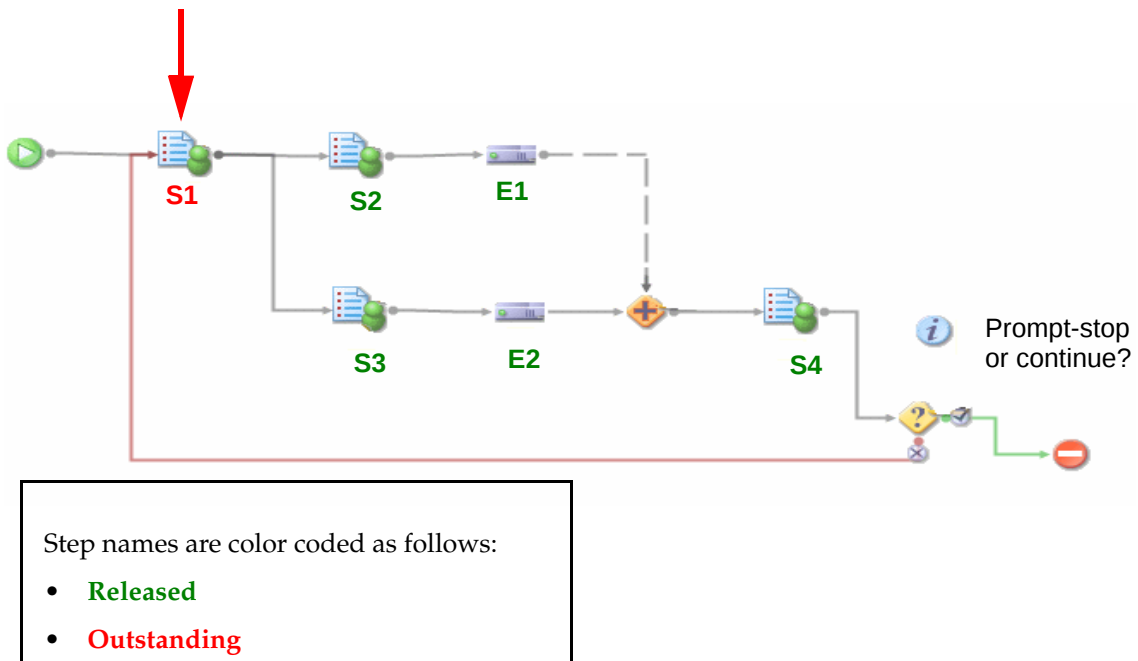
- When step **S1** is released, two parallel flows are triggered (step **S2** and EAI step **E1**, step **S3** and EAI step **E2**).

- Each flow displays a form and, when the form is released, runs an EAI step.
- A Wait is triggered when **E2** is released. The procedure waits until **E1** has also been released.
- **S4** is then sent out, and asks the user if they want to stop or continue.
- If the user opts to continue, **S1** is sent out again. (The loop can continue in this way until the user opts to stop.)



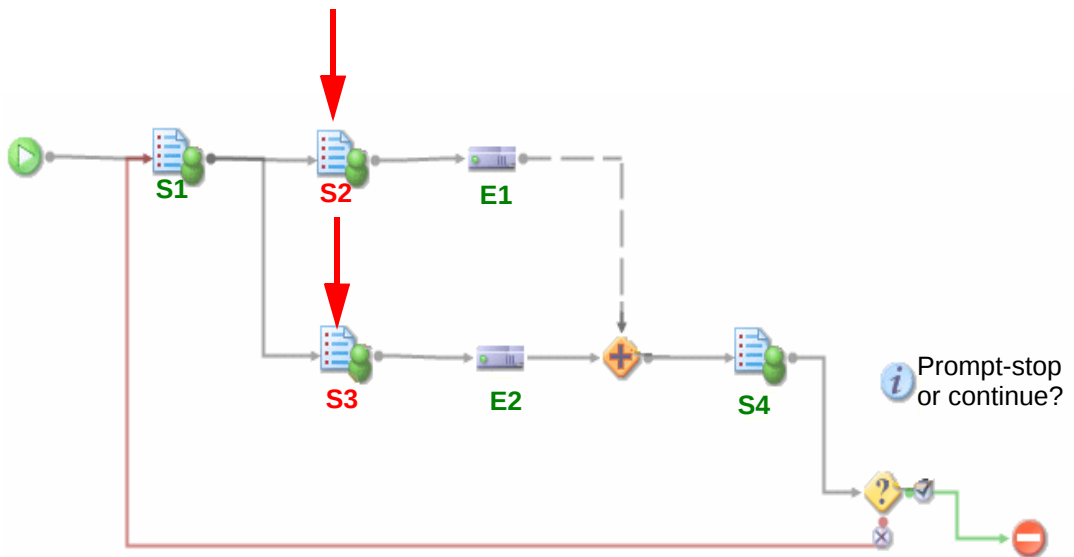
1. Suppose that a case is started and processed through. The user, on **S4**, chooses to continue. When the condition is processed, the procedure loops back to **S1**,

which has status **Outstanding**. All other steps still have status **Released** from the first pass through.

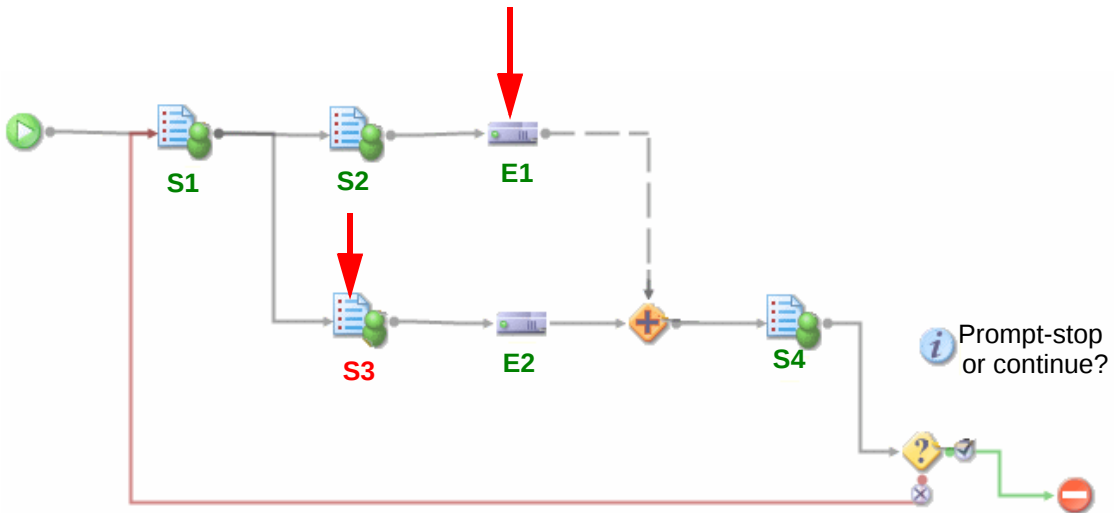


- The case now runs through the procedure again. When **S1** is released, **S2** and **S3** are sent out and their status is changed to **Outstanding**. *Note that E1 and E2*

still have status **Released**, because they have not yet been processed on this second pass through the procedure.

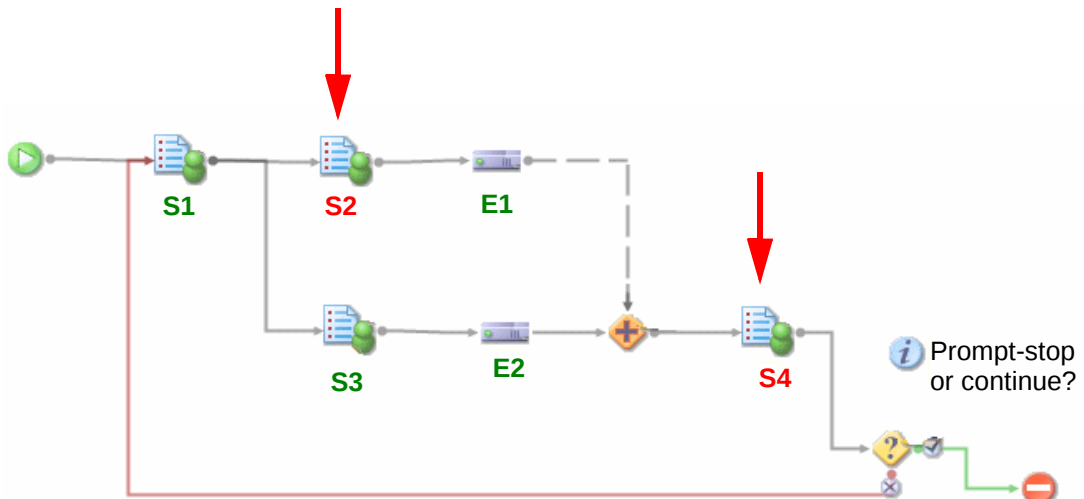


3. If **S2** and **E1** are processed and released, but **S3** is still **Outstanding**, the procedure waits. This is because the Wait is not processed until **E2** has been processed and released.



However, the situation is different if **S3** and **E2** are processed and released while **S2** is still **Outstanding**. In this case the Wait is processed when **E2** is released and, although the procedure should wait because **E1** has not yet been processed, **E1** still has status **Released** from the first pass through. Because

both **E1** and **E2** have status **Released**, the Wait is released and **S4** is sent out - which is not the desired result.

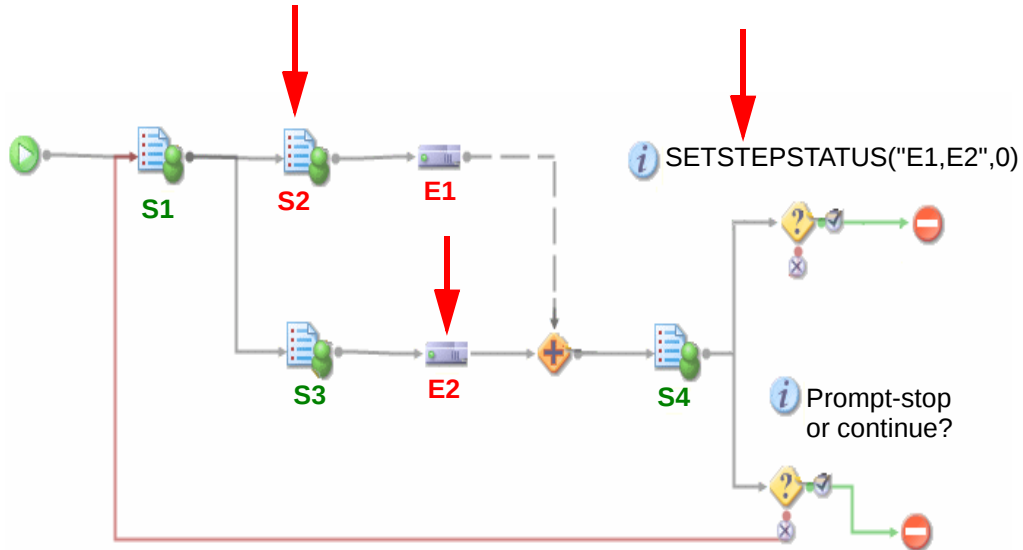


Using SETSTEPSTATUS to Control the Loop

If you want to use Waits in a loop in this way you should use the SETSTEPSTATUS function. You do this by inserting an extra, parallel condition after **S4** which uses the following condition expression:

```
SETSTEPSTATUS("E1,E2",0)
```

When **S4** is released, the condition is processed and the server resets the status of steps **E1** and **E2** to **Outstanding**. This ensures that when **S3** is released, the Wait will process in the desired fashion, and **S4** will only be sent out when **E1** and **E2** have been released.

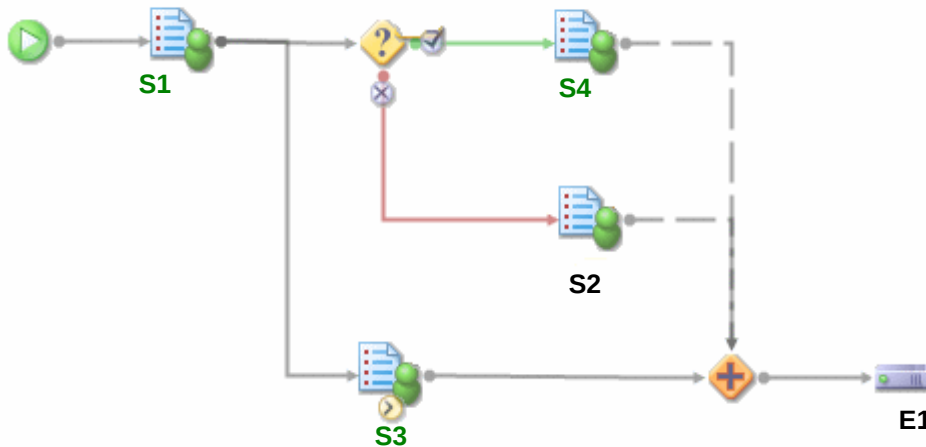


Step names are color coded as follows:

- Released
- Outstanding

Using Waits with Conditions

In the following example the intention is that when step **S3** is released, the procedure should wait until *either* **S2** or **S4** have been released. However, the Wait can never be triggered because one of the steps will never be sent out by the server. Its status will always be **Not Processed**.



Step names are color coded as follows:

- Released

Using SETSTEPSTATUS to Control the Wait

You can use the SETSTEPSTATUS function to control the Wait and ensure that it gets triggered, as follows:

- Insert a condition after **S4**, which uses the following condition expression:

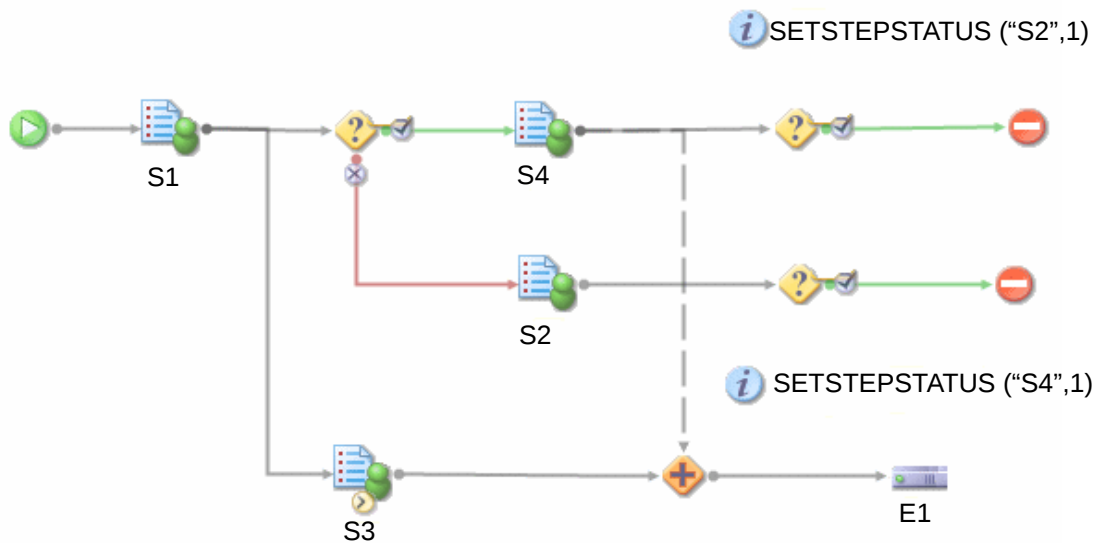
```
SETSTEPSTATUS ("S2",1)
```

If **S4** is released, this condition is evaluated and the status of **S2** is set to **Released**. The Wait is therefore triggered and **E1** is processed.

- Insert another condition after **S2**, which uses the following condition expression:

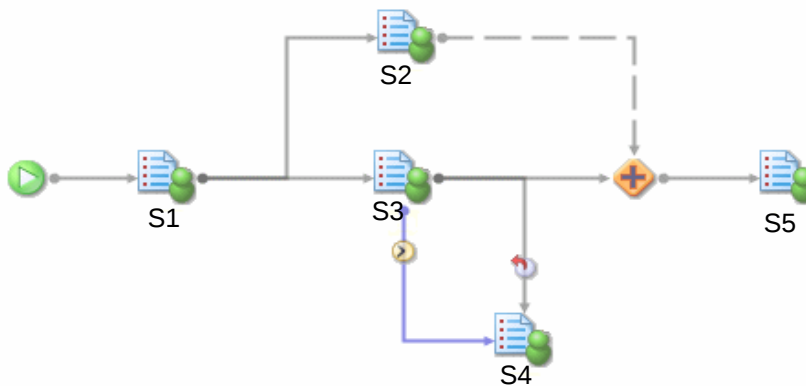
```
SETSTEPSTATUS ("S4",1)
```

If **S2** is released, this condition is evaluated and the status of **S4** is set to **Released**. The Wait is therefore triggered and **E1** is processed.

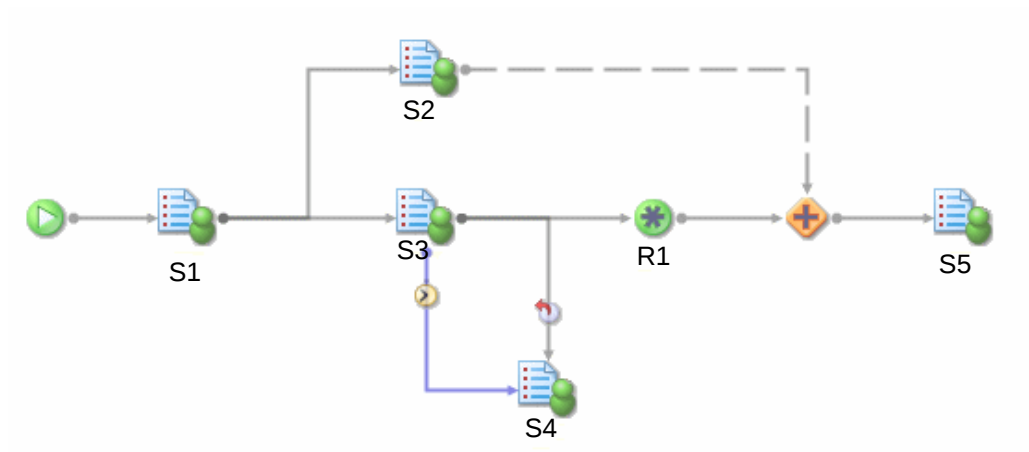


Using Waits with Withdraw

In the following example it is intended that **S4** will be withdrawn after **S3** is released but this doesn't happen and the step is never withdrawn. This is because iProcess expects a Wait to be the only action on a step.



The solution is to add a complex router **R1** between **S3** and the Wait. This allows the Wait to be the only action after the Complex Router and the Withdraw is actioned correctly.



Chapter 8

Making Procedures Easier to Follow

A small procedure can be very simple to follow but larger and more complex procedures are more difficult to read and edit in the TIBCO iProcess Modeler. This chapter describes some tools and techniques that you can use to make procedures easier to follow and understand:

Topics

- [Setting TIBCO iProcess Modeler Options, page 84](#)
- [Using Swim Lanes, page 85](#)
- [Changing Step Icons, page 97](#)
- [Annotating Procedures for Clarity, page 98](#)
- [Changing the Object Label Position, page 100](#)
- [Working with Links, page 101](#)
- [Setting Link Labels and Icons, page 102](#)
- [Changing Link Styles and Animation, page 104](#)
- [Using Routers to Simplify Visual Layout, page 106](#)
- [Using Complex Routers to Simplify Procedure Logic, page 107](#)
- [Using GOTOSTEP to Simplify the Procedure Routing, page 110](#)
- [Zooming In and Out of a Procedure, page 111](#)
- [Changing Procedure Orientation, page 112](#)
- [Using the Snap-To Grid, page 113](#)
- [Saving a Procedure as an Image, page 114](#)

Setting TIBCO iProcess Modeler Options

To view the TIBCO iProcess Modeler options, click **Options** on the **Menu Bar**.

In addition to the **Printing** options described in “Printing a Procedure” in *TIBCO iProcess Modeler - Procedure Management*, there are the following options:

- **Display Text.** You can choose to display the name, description, extended description, and/or link labels with objects and links.
- **Link Styles.** You can choose the type of link, the line width, and the type of start and end cap for the link. For more information on links see [Working with Links on page 101](#), [Setting Link Labels and Icons on page 102](#), and [Changing Link Styles and Animation on page 104](#).
- **Link Animation.** You can choose to animate the links and the speed of the animation. You can also select the link points (round, arrow, and multiple). For more information on link animation, see [Changing Link Styles and Animation on page 104](#).
- **Show Default Icons.** iProcess has standard icons it uses for each type of step, but you can change these icons to something more representative. Selecting this option will display the default icons and de-selecting the option will return the display to the changed icons. Showing default icons is not maintained between TIBCO iProcess Modeler sessions.
- **Confirm Withdraw Connections.** Selecting this option displays a confirmation screen when you draw a withdraw link connection. De-selecting the option allows you to draw a withdraw link without displaying the confirmation screen.

Using Swim Lanes

Swim lanes are a visual tool that enable you to organize how you display your iProcess Engine procedure objects. Swim lanes can be vertical or horizontal. By specifying category labels when you create your swim lanes, you can group procedure objects into categories of your choice.

You can choose whether to use swim lanes or not, depending on your requirements. This section describes:

- [Enabling Swim Lanes on page 85](#)
- [Swapping Between Swim Lane Types on page 89](#)
- [Configuring Swim Lanes on page 90](#)

Enabling Swim Lanes

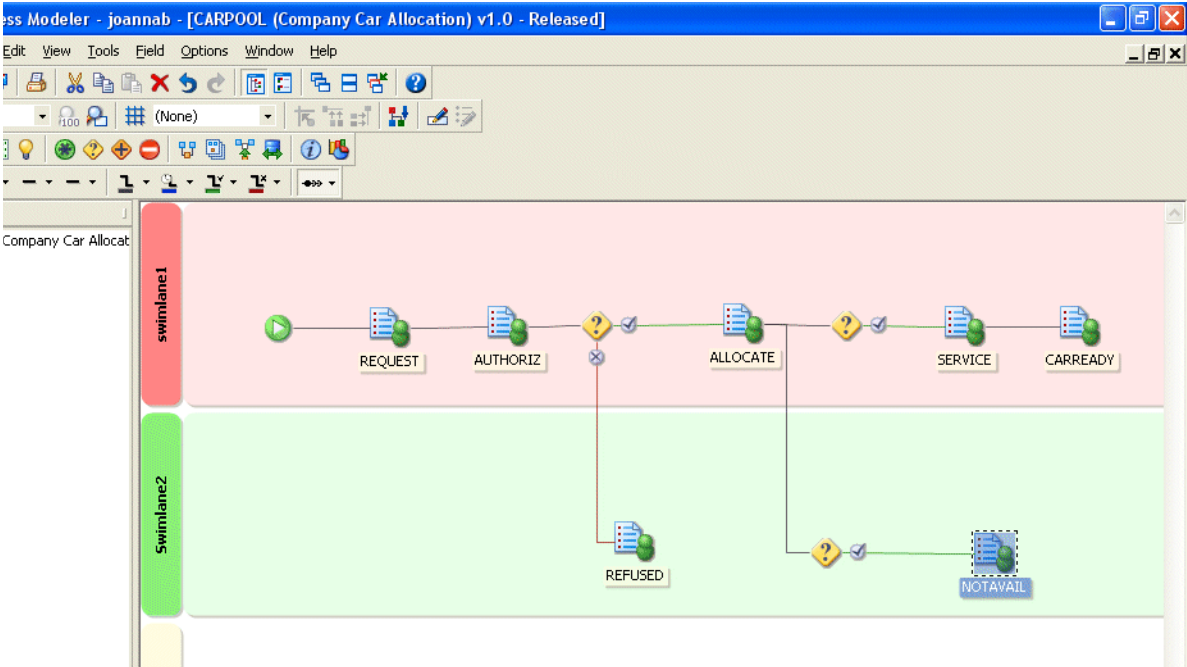
To enable swim lanes:

1. From the iProcess Modeler, click **Swim Lanes**.
2. Select one of the following from the drop-down list:
 - **None**. If you have swim lanes already enabled, you can select **None** if you want to disable them.

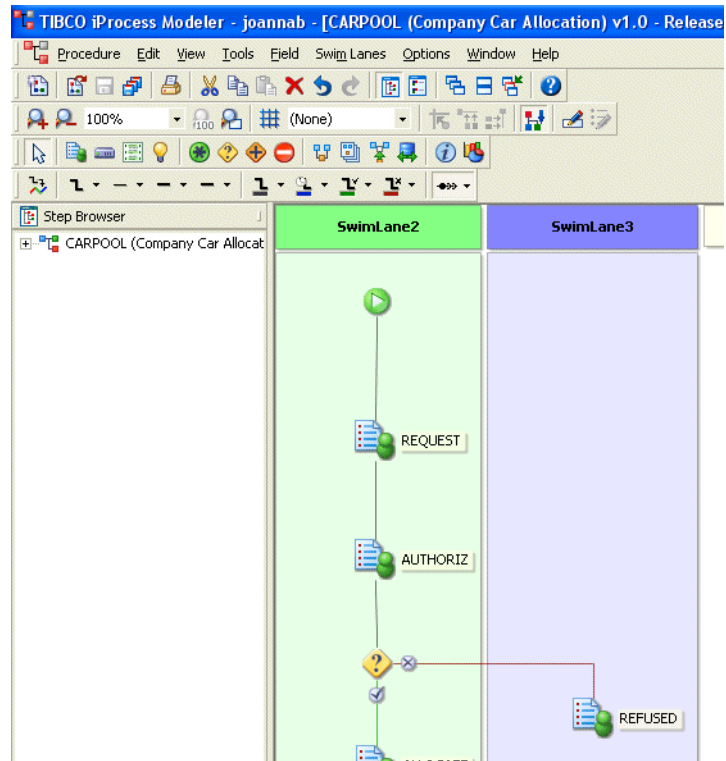


If you disable swim lanes then the procedure objects are de-categorized.

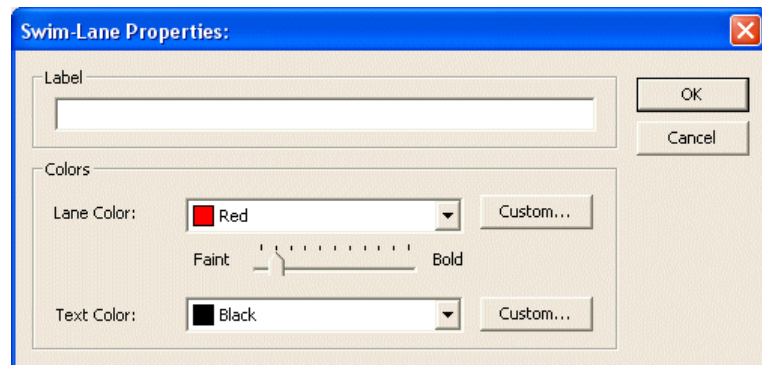
— **Horizontal.** Select **Horizontal** if you want horizontal swim lanes.



— **Vertical.** Select **Vertical** if you want vertical swim lanes.



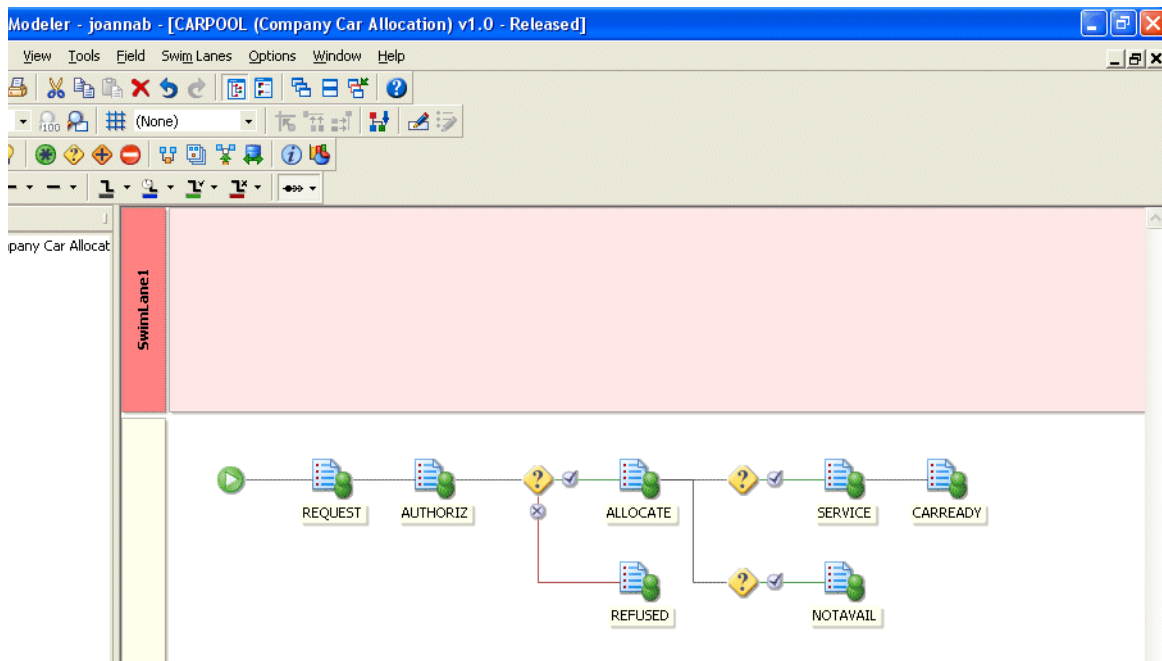
3. The **Swim-Lane Properties:** dialog is displayed.



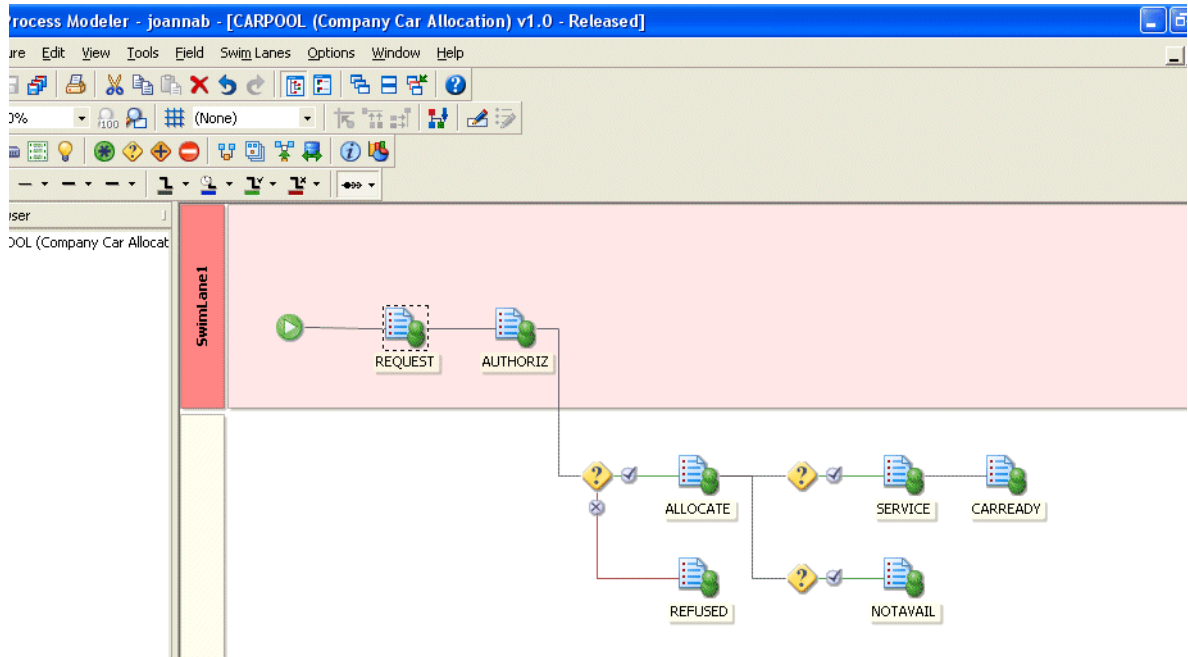
The following table describes the properties that you can set for a swim lane.

Property	Description
Label	The name of the category to be used for this swim lane. The name can be up to 128 characters.
Lane Color	The color to be used for the swim lane.
Text Color	The color to be used for the label text.

Once you have configured the properties of your swim lane, click **OK**. If you selected a swim lane type, a single swim lane is created. All the procedure objects are moved below the new swim lane.



4. Depending on your requirements, drag and drop your procedure objects to the swim lane you have just created.



5. To add more swim lanes, see [Adding Swim Lanes on page 90](#).
6. See [Configuring How Swim Lanes are Displayed on page 96](#) for information on how to configure how your swim lanes are displayed.

Swapping Between Swim Lane Types

Depending on your requirements, you can swap between swim lane types. For example, you can swap from horizontal to vertical and vice versa.

If you swap between horizontal and vertical swim lane types, the top-down layout mode is swapped.

If you swap from a swim lane type to no swim lanes:

- the procedure objects are de-categorized.
- any extra space around the objects is automatically compressed.

Configuring Swim Lanes

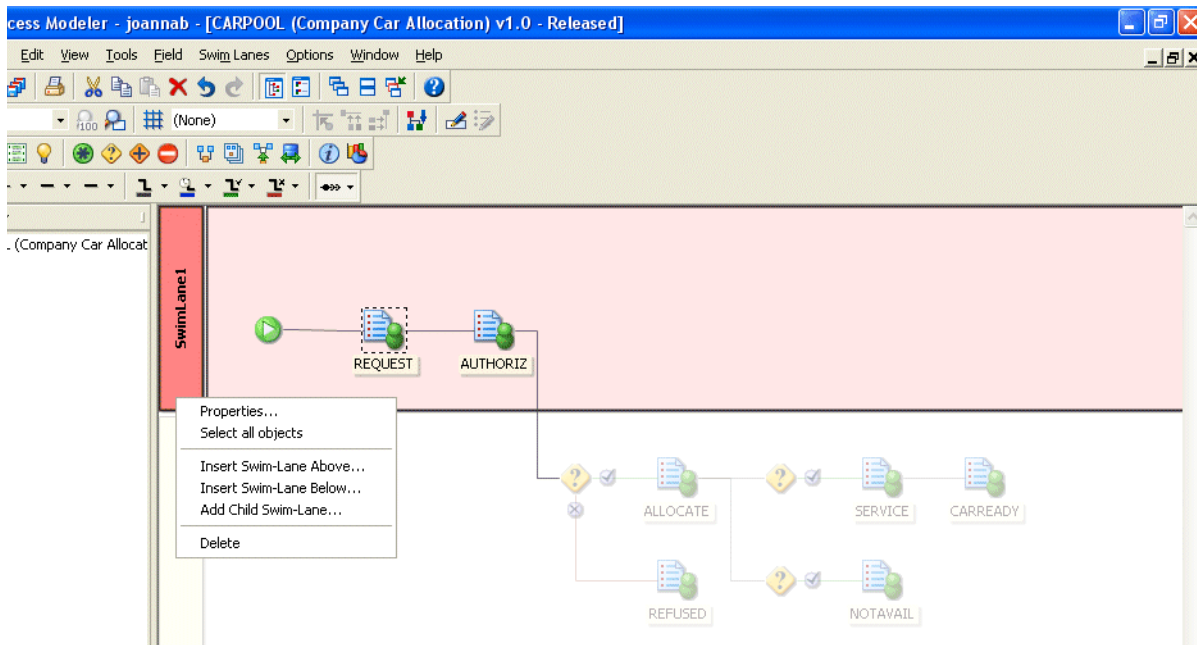
The following section describes how to edit the swim lanes. It describes:

- [Adding Swim Lanes](#)
- [Creating Child Swim Lanes](#)
- [Selecting Swim Lanes](#)
- [Resizing Swim Lanes](#)
- [Deleting Swim Lanes](#)
- [Editing Swim Lane Properties](#)
- [Configuring How Swim Lanes are Displayed](#)

Adding Swim Lanes

To add a new swim lane:

1. Right-click the header of the chosen swim lane.

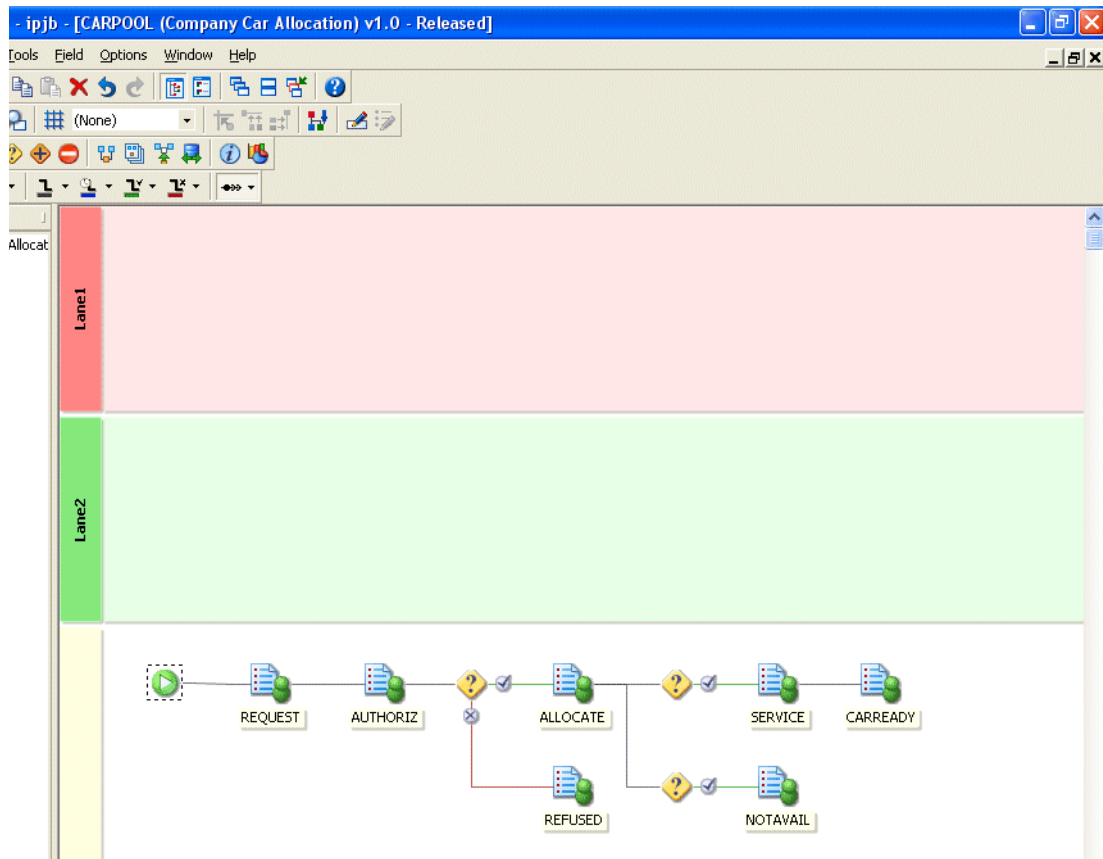


Depending on your requirements, select either:

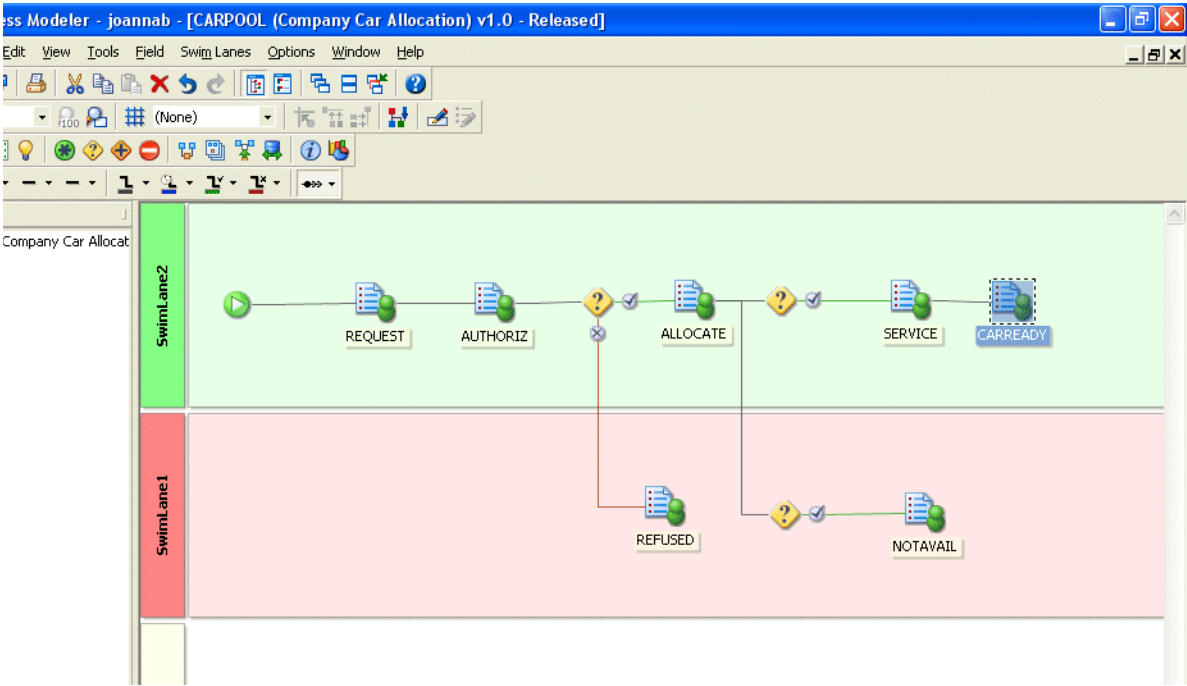
- **Insert Swim-Lane Above...**
- **Insert Swim-Lane Below...**

The **Swim-Lane Properties** dialog is displayed.

2. See [step 3](#) in [Enabling Swim Lanes](#) for information on how to configure the swim lane properties. Click **OK**. A new swim lane is inserted.



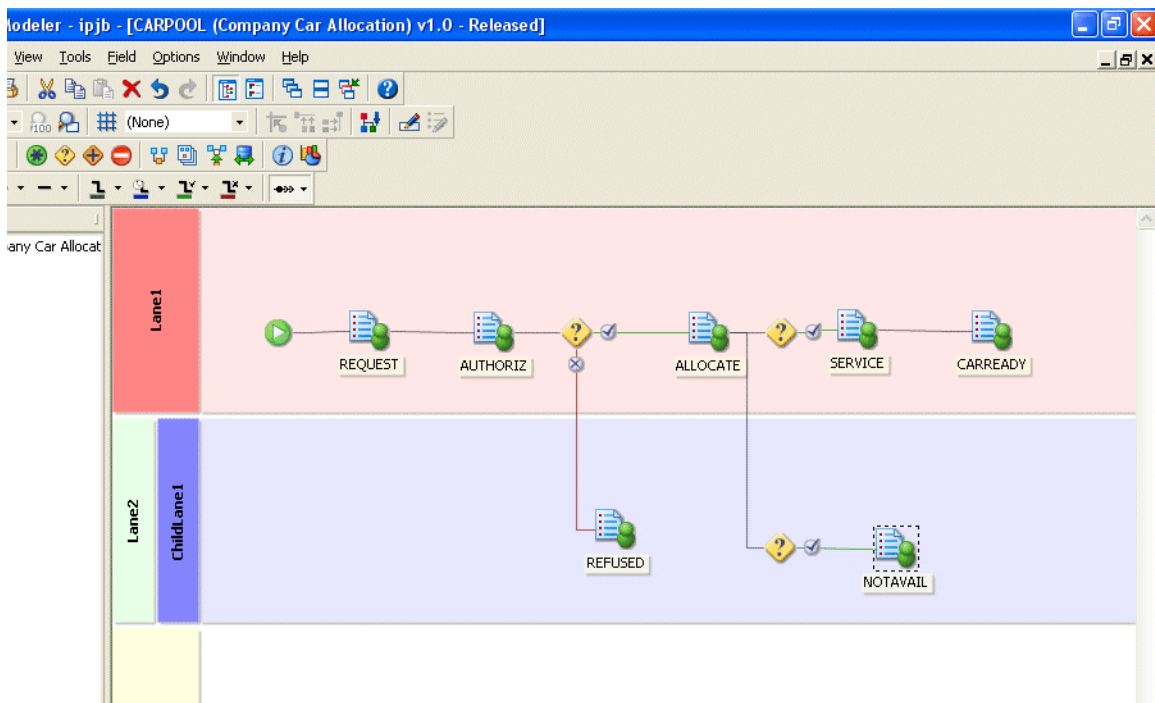
3. Depending on your requirements, drag and drop your procedure objects to the swim lane you have just created.



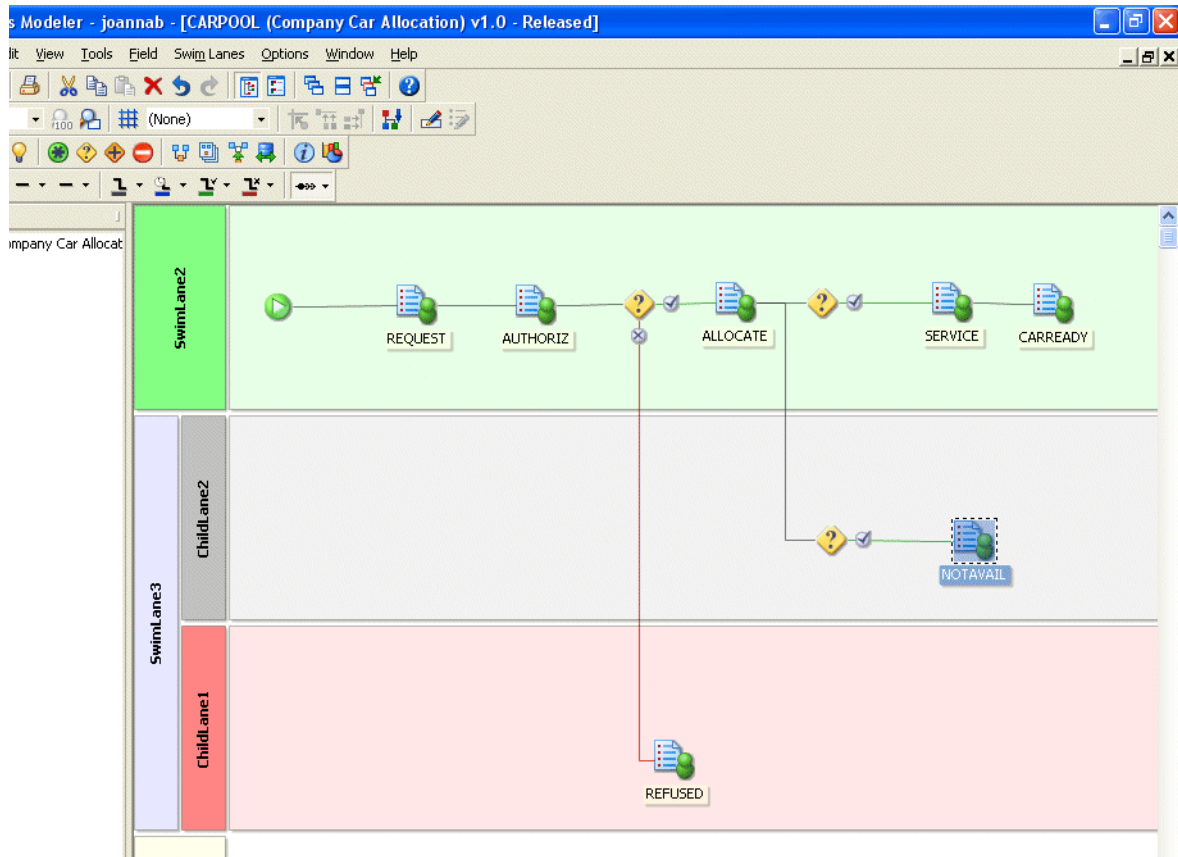
Creating Child Swim Lanes

You can nest swim lanes within other swim lanes, depending on your requirements. To do this, create a child swim lane in the swim lane where you want to nest other swim lanes. To create a child swim lane:

1. Right click the header of the swim lane you want to create a child swim lane from and select **Add Child Swim-Lane....** The **Swim-Lane Properties** dialog is displayed.
2. See [step 3 in Enabling Swim Lanes](#) for information on how to configure the swim lane properties.
3. Click **OK**. The child swim lane is created.



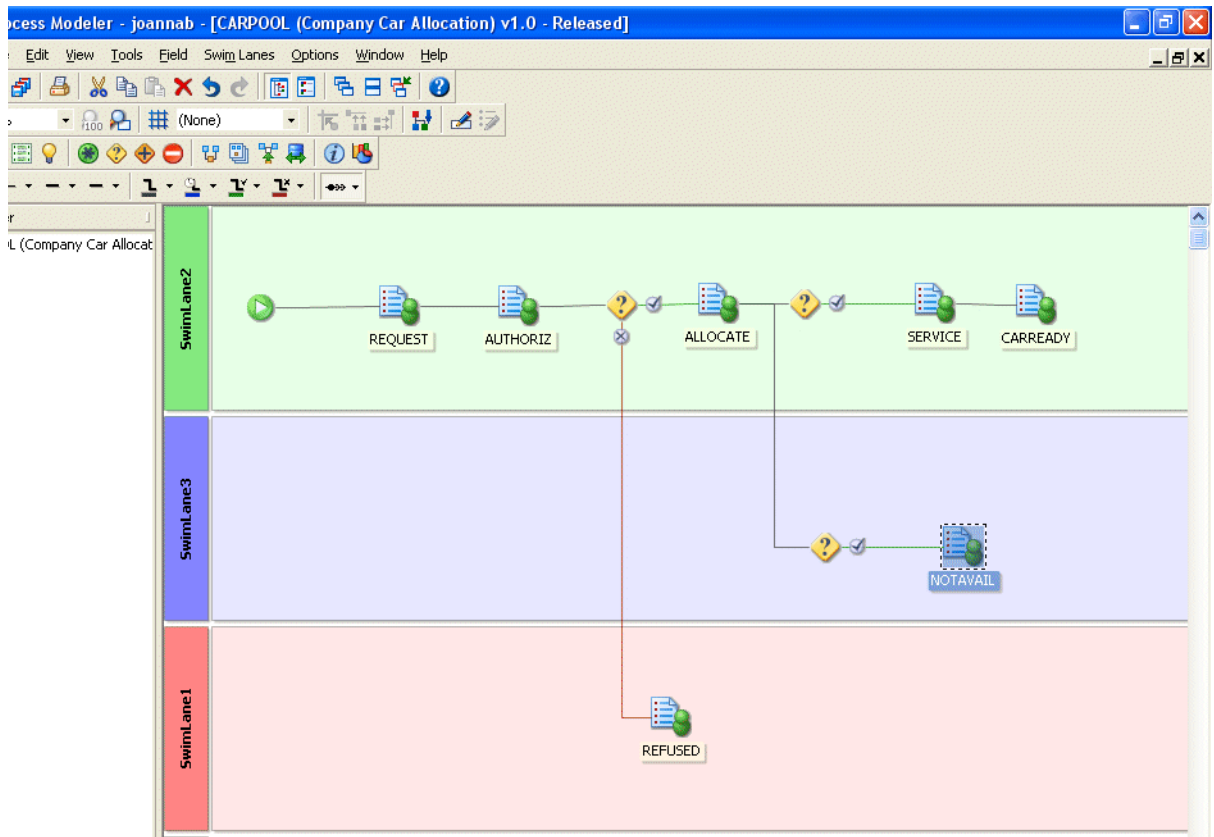
- Repeat steps 1 -2 for each child swim lane that you want to create.



- Depending on your requirements, drag and drop your procedure objects to the child swim lane(s) you have just created.

Selecting Swim Lanes

You can select a swim lane or multiple swim lanes, depending on your requirements, using CTRL + Click. This is useful, for example, if you want to delete more than one swim lane or you want to highlight particular procedure objects, as shown below.



You can also select all the procedure objects within a swim lane by right clicking the header of the swim lane whose procedure objects you want to select and clicking **Select all objects**.

To deselect a swim lane(s), press Escape or click on another swim lane.

Resizing Swim Lanes

You can resize a swim lane by dragging the separating lines between the swim lanes.

Deleting Swim Lanes



You can only delete a swim lane if it is empty, unless it is the last child swim lane. If it is the last child swim lane, the parent swim lane inherits the procedure objects.

To delete a swim lane:

1. Move the procedure objects from the swim lane you want to delete to another swim lane. This is because you cannot delete a swim lane if it contains procedure objects.
2. Right click the tab header of the swim lane you want to delete and select **Delete**. The swim lane is deleted.



If you are deleting multiple swim lanes, use the Delete key.

Editing Swim Lane Properties

To edit the properties of a swim lane:

1. Right-click the header of the swim lane whose properties you want to change and select **Properties**. The **Swim-Lane Properties** dialog is displayed.
2. See [step 3 in Enabling Swim Lanes](#) for information on how to configure the swim lane properties.

Configuring How Swim Lanes are Displayed

To configure how your swim lanes are displayed, click **Options > Swim Lanes**. You can select one of the following:

- **Plain Style**. This is the default option when you enable swim lanes. Select this if you want to display your swim lanes as squares.
- **Tablet Style**. Select this if you want to display your swim lanes as tablets.
- **Border**. Select this if you want to display your swim lanes with a border.
- **Color Header Only**. Select this if you only want the swim lane header to be displayed with a color rather than the whole swim lane.
- **Shadow**. Select this if you want to display your swim lanes with shadows.

Changing Step Icons

iProcess has default icons that are used to represent the different types of steps, but you can change these to other icons. For example, you might want to use a word processing icon to represent a letter or a database icon where an external database is being interrogated by iProcess.

To change an icon:

1. Right click on the step you want to change, then click **Change Icon**.

The **Select Object Icon** dialog is displayed.



2. Enter the file name containing the icons or click **Browse**. Choose an icon from those displayed and click **OK**.

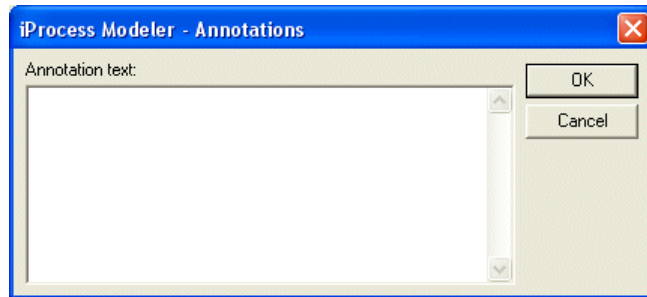
If you have already changed an icon and want to revert to the original icon, click **Use Default** and the default iProcess icon will be displayed.

Annotating Procedures for Clarity

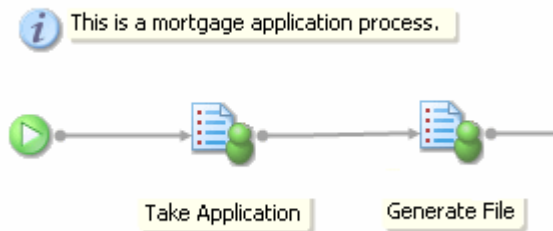
Annotating your procedures is an easy way to ensure clarity of your procedures.

To add an annotation:

1. Click the annotation tool  and place it in the appropriate position on your procedure.
2. Enter the text that you want displayed and click **OK**.



The annotation text appears with the  symbol on your procedure. You can re-position the annotation text by clicking and dragging the symbol.



Annotation is frequently used alongside Conditional Actions to show the decision being taken. A Conditional Action can also have an annotation that displays in the hover pop-up.

Annotation can also be associated with a particular step.

1. Select the required step in your procedure and right click.
2. Click **Annotations** and enter the information.

Annotation that is specific to a step is not displayed on the TIBCO iProcess Modeler procedure layout but can be accessed at any time by right-clicking on the object.

Changing the Object Label Position

Text labels for objects display in the following default positions:

Object	Text Label Position
Annotation	Right side. (You cannot reposition annotation text.)
Condition	Since conditions have at least one side that is not linked, the text label appears on the free side.
Start	No text label.
Elbow	No text label.
Wait	No text label.
All other objects	Bottom side for left > right layout mode.
	Right side for top > down layout mode.

You can override the default text position (except for annotation) by right-clicking on an object (or selected objects) and selecting **Object Label Position**, then selecting the text position. Selecting **Automatic** uses the text label default position. To choose which text displays (the object name, description, extended description, and link labels), select **Option > Display Text**.

Working with Links

Links between objects are selectable in their own right, just as objects are selectable. To select a link, you can click the link or label. To select multiple links, hold the **CTRL** key and click the additional links. This leaves the original links selected while selecting the additional links. Also, if you select a group of objects, any links between those objects are selected.

To move a link, click the link and drag it to the new position. (Note that moving a link adds a router to the link.) Press **ESC** to cancel the move.

To delete a link, select the link and press **Delete**.

You can right-click a link to access the following popup menu options:

Menu Option	Description
Link Label	Allows you to edit the link label parameters.
Add Router	Inserts a router at the point the mouse was clicked.
Go To Source	Selects the source object for the link and brings it into view (if not currently visible on screen).
Go To Target	Selects the target object for the link and brings it into view (if not currently visible on screen).
Reset Link Styles	Resets the link styles and color to those options currently selected on the link style toolbar.
Delete Link	Deletes the link without deleting the selected objects.

If you select multiple links (with no objects selected) and right-click a selected link, a popup menu containing the following menu options appears:

Menu Option	Description
Link Label	Allows you to edit the link label parameters.
Reset Link Styles	Resets the link styles and color to those options currently selected on the link style toolbar.
Delete Link(s)	Deletes the link without deleting the selected objects.

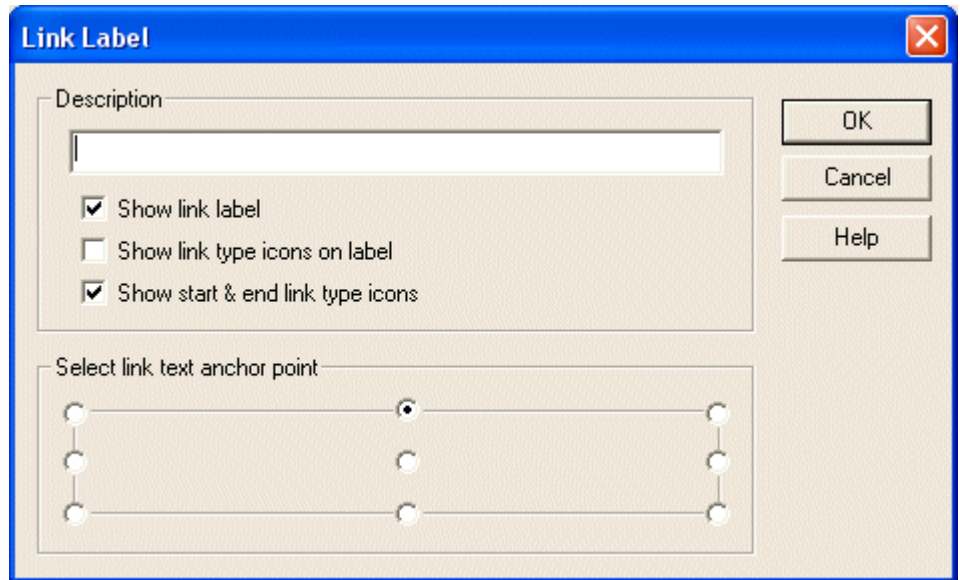
Setting Link Labels and Icons

You can assign a label to a link. The link label can consist of text and icons that identify the link type. The link type icons are:

Icon	Type
	Deadline
	Condition True
	Condition False
	Withdraw

The link icon displays on the link line close to the link start point (or end point in the case of the withdraw link icon). If the object text label is positioned on the same side as the link, the link icon appears further along the link line to avoid overlaying the text. You can elect to hide the start and end link type icons using an option on the **Link Label** dialog.

To access the **Link Label** dialog, double-click a link or right-click a link and select the **Link Label** menu option. The **Link Label** dialog appears.



The link label description text can be up to 128 characters. The text is truncated (or not displayed at all) if display space is limited. However, you can hover the cursor over the link and the full label text displays in a pop-up window.

Changing Link Styles and Animation

You can select a variety of link styles by using the Link Styles toolbar (or by selecting **Options > Link Styles**).

Style	Icon	Description
Convert		Converts the selected links to the styles and colors currently selected on the toolbar.
Flowchart		Allows horizontal and vertical lines.
Straight		Allows diagonal lines.
Curved		Allows best-fit curved lines through routers.
Line Width		Allows you to select the thickness of the lines by pixel.
Start-cap Style		Allows you to select a line start-cap style from a drop-down list. Your choices are None, Round, Square, and two arrow styles.
End-cap Style		Allows you to select a line end-cap style from a drop-down list. Your choices are None, Round, Square, and two arrow styles.
Normal		Allows you to set the Normal link color.
Deadline		Allows you to set the Deadline link color.
Condition True		Allows you to set the Condition True link color.
Condition False		Allows you to set the Condition False link color.
Animation		Allows you to turn on/off link animation, select the type of points (dots or arrows), and select the animation speed.

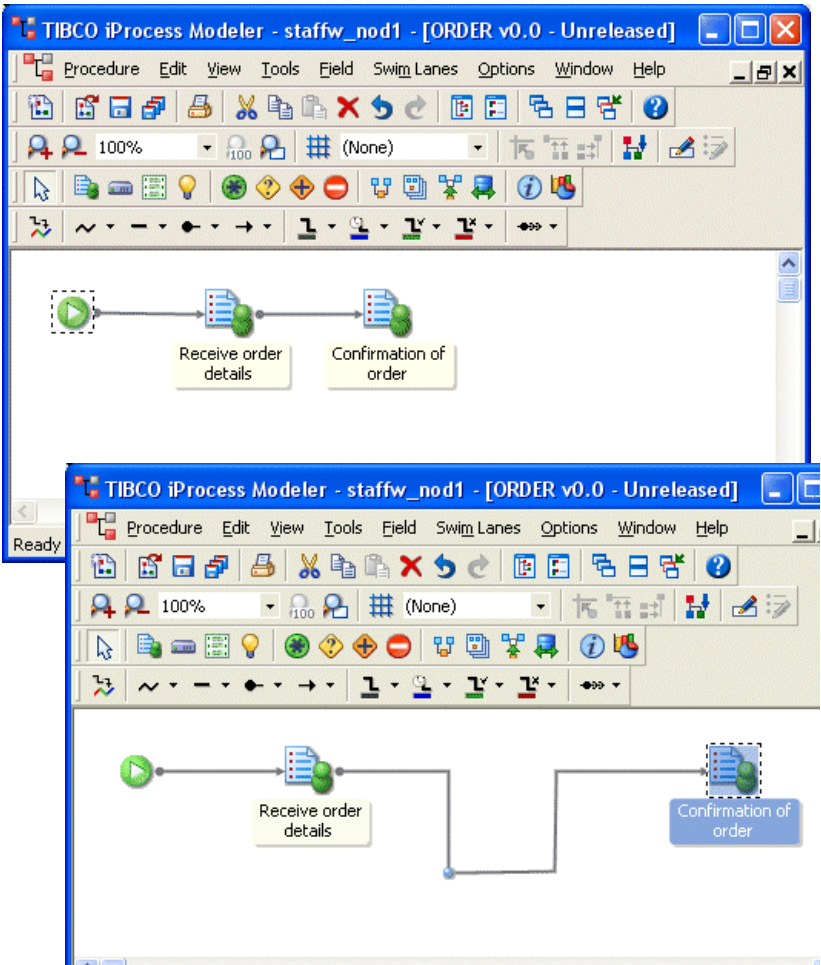
Changing a link style makes the change only for the selected link. To make the same changes to all links in a procedure, click  or select **Edit > Reset All Link Styles**.

Using Routers to Simplify Visual Layout

Routers are used to simplify the visual layout of a procedure. Using them allows you to define the route of the connecting line between two steps. Any number of routers can be placed anywhere on any type of link to give greater control over the layout.

When drawing a line between two steps, right-click to place a router at the cursor position. To add a router to an existing link, right-click the link and select **Add Router** or simply click and drag the link.

The following simple example shows first how the line would be automatically drawn by iProcess and then how it can be displayed when using a router.



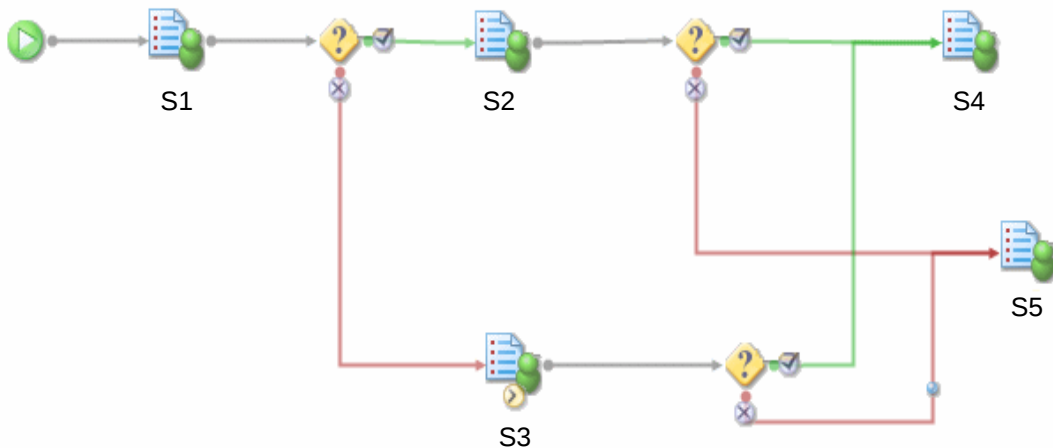
Using Complex Routers to Simplify Procedure Logic

A Complex Router is a step that is hidden to the user at run time and never appears in a work queue. The iProcess background process releases the Complex Router and processes its actions without any input from the user.

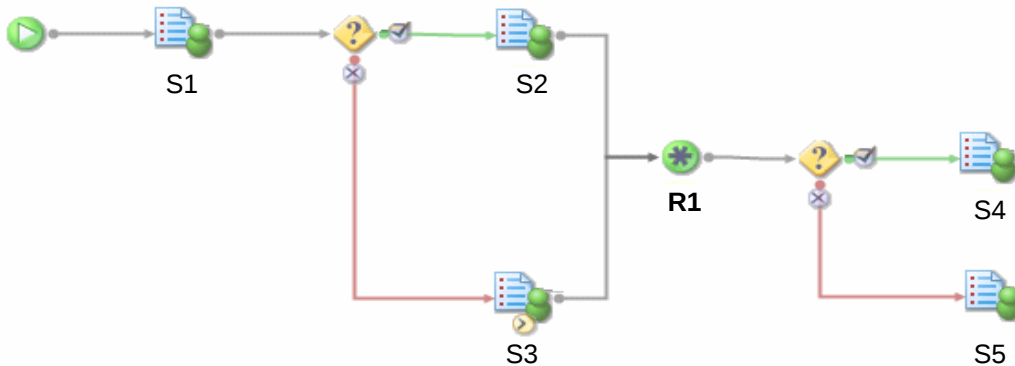
Complex Routers can be useful in the following situations:

- *To simplify procedure layout.* The following is an example:

A conditional action can only have one step leading to it, but it may be that two or more steps need to use the same condition to decide an action. In the example below, **S1** processes either **S2** or **S3**. Both of these steps process either **S4** or **S5**. When this happens, the resulting process is difficult to follow.



Using a Complex Router **R1** in this situation makes the process visually simpler, as you can see below.



- *To make the case number available to the first step.*

On the TIBCO iProcess Workspace, when the addressee (or one of the addressees) of the first step of a procedure is the same as the user starting the case, the form appears immediately. In this instance, the case number is not available to the first step as it is assigned by the iProcess background when the case start is processed and returned to the work queue.

By placing a complex router between the Start object and the first step you can overcome this problem. iProcess takes the Complex Router as being the first step, assigns the case number, then processes the next step which is the first step to be displayed to the user.

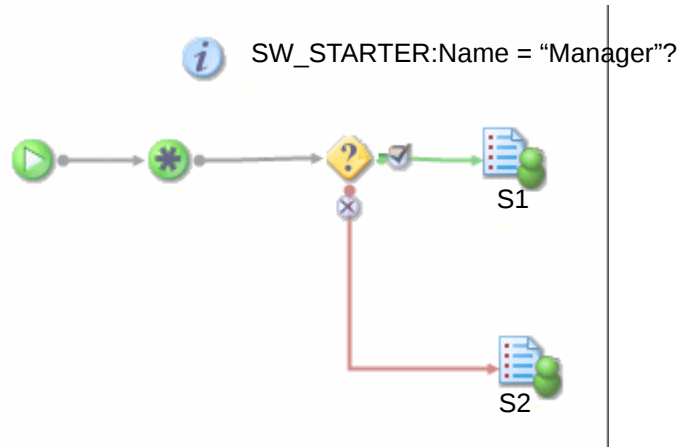


- *To provide conditional start steps.*

Placing a Complex Router immediately after the Case Start object allows you to conditionally set the start step, depending on the value of case data. (A Condition cannot be placed directly after the Case Start object.)

In the following example, as shown by the annotation, STEP1 is actioned if the case is started by the Manager user. If it is started by another user, STEP2 is actioned. (Note that STEP1 or STEP2 will be sent to the appropriate queue in

the normal way - the case starter will not receive the form for either when starting the case.)



Complex Routers cannot have withdraw or deadline actions and do not have entries in the Audit Trail.

Using GOTOSTEP to Simplify the Procedure Routing

You can alter the normal flow of your procedure using the GOTOSTEP expression. For example, if you want to jump to a specific step (perhaps for exception handling), you can jump directly to an exception handling step rather than trying to define the necessary workflow layout to do it (which can often get difficult to interpret).

On release of the current step, you can use GOTOSTEP to jump to another step. You can choose to process the current step or not. The GOTOSTEP expression has a flag that can be used to specify if the current workflow processing is continued or stopped when the workflow jumps to the new step.

For example, if you have a step where a script is run and an exception is raised, you can use the GOTOSTEP to jump to an exception handling step rather than trying to define a complicated workflow to handle the various workflow routes. Using this expression can cut down the amount of workflow “spaghetti” which can be created when defining a complicated procedure.

For more information and a detailed example about using the GOTOSTEP syntax, refer to “GOTOSTEP” in *TIBCO iProcess Expressions and Functions Reference Guide*.

Zooming In and Out of a Procedure

For ease of use when viewing a procedure you can zoom in and out or choose a particular part of a procedure to view.

Task	Menu Option	Toolbar Button
Enlarge the size of the procedure.	View > Zoom In	
Reduce the size of a procedure.	View > Zoom Out	
Set zoom level.	View > Zoom Level	
Enlarge a particular section of a procedure. Use on large procedures to give you a rectangular box which you can drag across your TIBCO iProcess Modeler window. If you place it over a particular section of your procedure and click, that section is enlarged.	View > Map	
Display the default size.	View > Normal	

Spinning the mouse wheel while pressing **CTRL** will zoom the view in and out.

Changing Procedure Orientation

The default orientation for procedures is left-to-right (horizontal). However, for some procedures, a top-down (vertical) flow makes more sense. The  button allows you to change the procedure orientation. Changing the orientation swaps the X and Y coordinates of each object and changes the link sides from right to bottom and from left to top. It also switches the default object side link types. See [Linking Procedure Objects on page 7](#) for more information.

Using the Snap-To Grid

The snap-to grid function allows you to align objects using a grid. When you move or place an object, the object is automatically snapped to the center of the nearest grid square. The following table contains the toolbar icons and a description of the toolbar functions:

Icon	Description
	Displays the snap-to grid.
	Allows you to select the size of the grid.
	Snaps the selected object to the center of the nearest grid square.

If a snap-to grid operation would cause objects to overlap, no objects are moved and you receive an error message.

Saving a Procedure as an Image

You can save a procedure map as an image file by clicking **Procedure > Save As > Image**. The **Save File** dialog appears. Specify the location, name, and type (JPEG, BMP, or PNG) of the output image. Click **Save** to view the image in the specified file and format at the current zoom level.



There is a limitation on the size of in-memory bitmaps. If the procedure is very large in the current zoom level a warning message is displayed. You can decrease the zoom level to make the view smaller until the size is under the limitation.

Customizing the Process Step Definer

The Process Step Definer has a number of setup options that can be changed to suit your way of working.

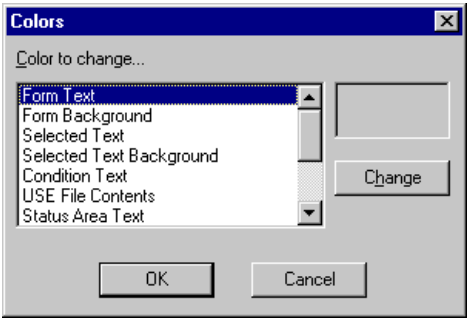
Open the Step Definer, click **Form > Setup** and you have several options, which are described in this chapter.

Topics

- [Colors, page 116](#)
- [Select Font, page 117](#)
- [Dynamic Scroll, page 118](#)
- [Show Field Names, page 119](#)
- [Line Length, page 120](#)
- [Tabs, page 121](#)
- [Nesting Level, page 122](#)

Colors

The colors that are used for the text and background in different parts of the form can be changed through the standard Windows **Colors** dialog.

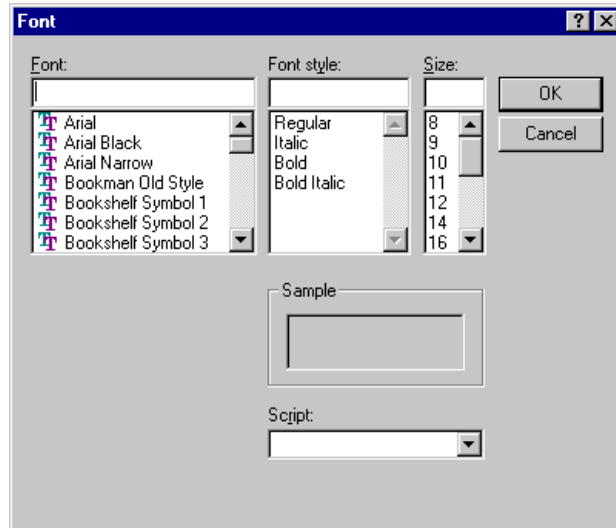


Click **Change** to display the color palette, make your selection and click **OK**.

Changes are implemented across all procedures and are saved between TIBCO iProcess Modeler sessions.

Select Font

Selecting this option takes you to the **Font** dialog allowing you to select the font you want and its style and size.



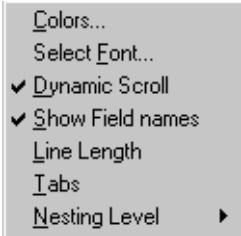
Changes made are per form and are saved between TIBCO iProcess Modeler sessions.

Dynamic Scroll

When **Dynamic Scroll** is on, the contents of the window move as you drag the scroll bar up or down. When it is off, the contents of the window will not move until you release the scroll bar.

Click on **Dynamic Scroll** to select it and a check mark appears to the left of the list. Click again to de-select it.

This option is per form and is not saved between editing sessions.



Show Field Names

When **Show Field names** is selected the names of fields appear in the input fields in the **Step Definer** as shown below.

The screenshot shows the 'Process Step Definer: MORTEDA-REVAPP' window. The menu bar includes 'Form', 'Edit', 'Conditions', 'Field', 'Use File', and 'Help'. The title bar says 'Mortgage Application Form'. The form contains the following fields with their names displayed inside:

- Customer Name: CUSTOMER_NAME
- Customer Address: CUSTOMER_ADDR1, CUSTOMER_ADDR2, CUSTOMER_CITY
- Customer Phone: CUSTOMER_PHON
- Salary: SALARY

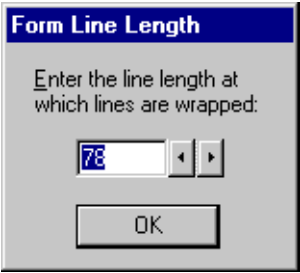
At the bottom, a status bar indicates 'Line : 1 Column : 1'.

Click on **Show Field names** to select it and a check mark appears to the left of the list. Click again to de-select it.

This option is set across all procedures and is saved between TIBCO iProcess Modeler sessions.

Line Length

Line length allows you to choose how long the lines in your form should be before they are wrapped.



The image shows a small dialog box titled "Form Line Length". Inside the dialog, there is a text label that reads "Enter the line length at which lines are wrapped:". Below this label is a text input field containing the number "78". To the right of the input field are two small square buttons with left and right arrow symbols. At the bottom of the dialog is a single button labeled "OK".

Enter the line length you want (to a maximum of 128 characters) and click **OK**.

Tabs

To change the distance in characters between tabs, select **Setup > Tabs**. Enter the distance you want and click **OK**.



The maximum tab length is 16 characters and the change is implemented across all procedures and is maintained between TIBCO iProcess Modeler sessions.

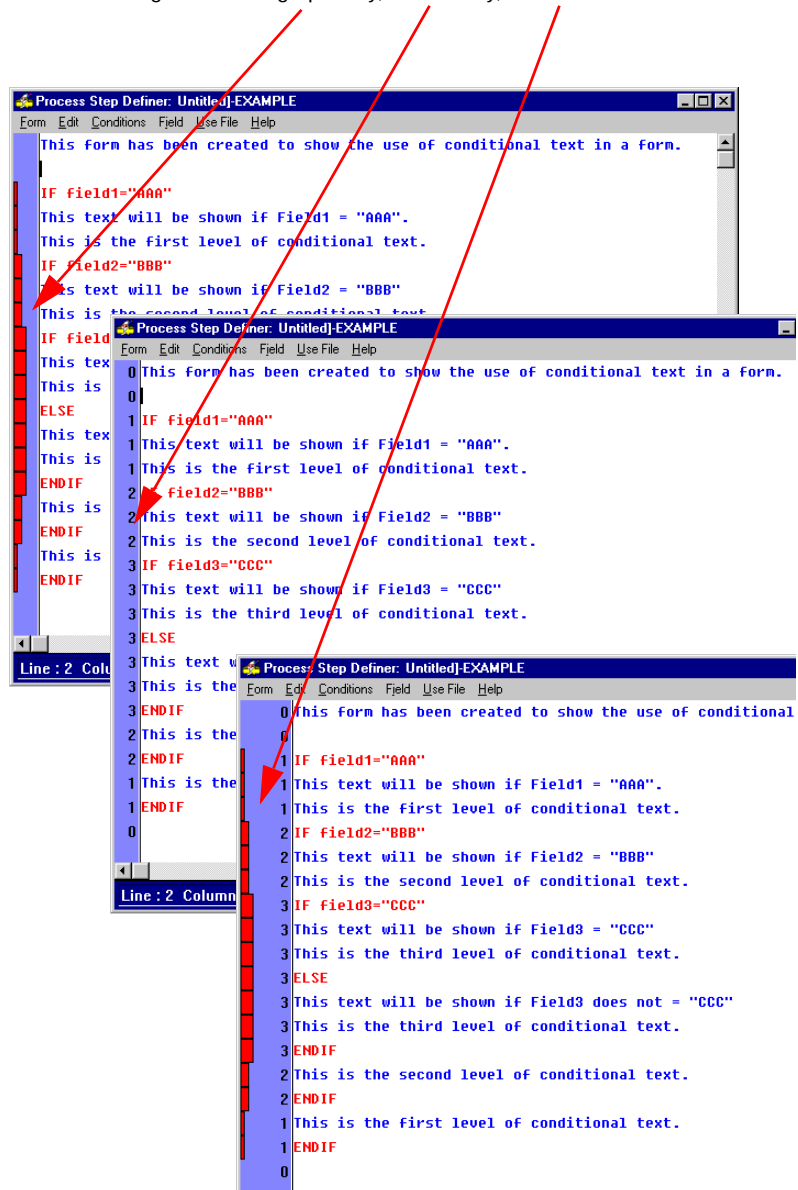
Nesting Level

When you use Conditional Text in your form (see [Using Conditional Text to Dynamically Change a Form on page 31](#)), you can nest up to 20 levels. To make it easier to follow when working on your form, you can choose to show the nesting levels either numerically or graphically.

This option is per form and is not saved between TIBCO iProcess Modeler sessions.

Select **Nesting Level** from the **Setup** menu on the form and choose **Numeric** or **Graphical** (or both).

Show the nesting level either graphically, numerically, or both.



Appendix A **Troubleshooting Procedure Definitions**

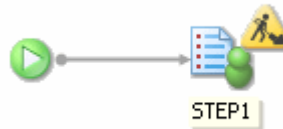
This appendix describes some typical problems you can encounter when defining and working with procedures.

Topics

- [An Under Construction Symbol Appears on a Step, page 126](#)
- [Changing the Currency Unit, page 127](#)

An Under Construction Symbol Appears on a Step

Problem One or more steps in your procedure definition show an Under Construction sign over the step icon.



Description When defining procedures, you need to be aware that steps need to have a certain amount of basic information. If this information is not defined, iProcess warns you by placing an Under Construction sign on the step.

Solution The following list describes the possible solutions you can try for each step type where you see a red question mark.

- **For a normal step**, select the step and right-click to select **Addressees**. Make sure an addressee has been entered for this step.
- **For a Sub-procedure call step**, select the step and right-click to select **Sub-Procedure**. Make sure a sub-procedure is chosen in the **Sub-Procedure Name** field.
- **For a Dynamic Sub-procedure call step**, select the step and right-click to select **Dynamic Sub-Procedure**. Make sure the **Sub-Procedure Name Array** field contains an array field and that all the required input fields are mapped.
- **For an EAI step**, select the step and right-click **Description**. Make sure a valid script definition has been entered and check that an EAI type has been selected.
- **For a Wait step**, make sure it is connected to 2 or more steps.
- **For a Condition step**, select the step and right-click **Condition**. Enter a valid condition.

Changing the Currency Unit

Problem You need to change the currency unit.

Description The default currency unit for iProcess is pounds and pence. For different currency units, you must change the system configuration. The currency unit information is stored in the *SWDIR/etc/ENGLISH.LNG/LONGNUM.FMT* file. To change the currency units, you must change all the references to pounds and pence in the *LONGNUM.FMT* file to the currency unit you require. You can, either:

- **Solution 1:** Amend the original *SWDIR/etc/LONGNUM.FMT* file, or
- **Solution 2:** Create a copy of the *SWDIR/etc/ENGLISH.LNG* directory, rename it to the language whose currency unit you require and amend the *LONGNUM.FMT* file in the new *SWDIR/etc/language.LNG* directory.

The sections below describe the steps to perform for each solution:

Solution 1 Complete the following steps:

1. Create a back up copy of the *SWDIR/etc/ENGLISH.LNG* directory.
2. Open the *SWDIR/etc/ENGLISH.LNG/LONGNUM.FMT* file in a text editor, for example, Textpad.
3. Change all references to pounds and pence in the *SWDIR/etc/ENGLISH.LNG/LONGNUM.FMT* file to the currency units you require.

Solution 2 Complete the following steps:

1. In *SWDIR/etc*, create a directory named *language.LNG* where *language* is the name of the language whose currency unit you want to use. For example, if you wanted to change the currency unit to dollars, you could create a directory called **USA.LNG**.
2. Copy all the files from the *SWDIR/etc/ENGLISH.LNG* directory to the *language.LNG* directory you have just created.
3. Open the *SWDIR/etc/language.LNG/LONGNUM.FMT* file in a text editor, for example, Textpad.
4. Change all references to pounds and pence in the *LONGNUM.FMT* file to the currency units you require.
5. Open **User Manager** in the iProcess Administrator.

6. For each user whose currency unit needs to change, change the user LANGUAGE attribute to the value of the *language.lng* directory that you created in [step 1](#). For example, if you created a directory called **usa.lng**, then change the LANGUAGE attribute from **english** to **usa**.
7. Open **Move SysInfo** in the iProcess Administrator.
8. Click **Move SysInfo** to apply the changes.

Index

A

- Action
 - conditional
 - defining 60
 - deadline withdraw, drawing 66
 - wait, defining 71
 - withdraw
 - drawing 65
 - using on sub-procedure 67
- Addressee
 - group 10
 - multiple user 10
 - single user 9
- Amperсандed field 24
- Annotation 98
- Application field, marking 27
- Array fields
 - about 16
 - defining 18
- Assigning addressee 9

C

- Calculating
 - case description 44
 - deadlines dynamically 56
 - text field 44
- Calculation
 - conditional 43
 - field 41
- Case
 - description
 - calculating 44
 - ignore suspend flag 14
 - suspend 14

- Changing
 - colors
 - in Step Definer 116
 - dynamic scroll setting, in Step Definer 118
 - font, in Step Definer 117
 - step
 - icons 97
 - tabs
 - in Step Definer 121
- Colors, changing
 - in Step Definer 116
- Command, form 34
- Complex router 107
- Condition
 - using waits with 79
- Conditional
 - action
 - defining 60
 - calculations 43
 - text 31
- Conditional actions
 - using 59
- Configuring
 - Process Step Definer 115
- Copy objects 6
- Copying field values 40
- Copying form contents 13
- Creating
 - form 21
 - procedure 1
- customer support xiii
- Cut objects 6

D

Deadline

- defining 52
- dynamic recalculation 56
- link 8
 - drawing 54
- on sub-procedure 55
- withdraw
 - action, drawing 66
 - link 8

Defining

- conditional action 60
- deadline 52
- field help 24
- fields 18
- roles 11
- step status options 13
- table field 48
- wait action 71
- waits 69

Deleting

- field 31
- swim lanes 96

Delimiters 46

Drawing

- deadline
 - link 54
- withdraw action 66
- withdraw action 65

Dynamic

- deadline recalculation 56
- scroll setting 118

E

Editing

- form 35
- marked fields 23

Editing form

- edit permission 13

Embedded field 24

F

Field

- ampersanded 24
- application 27
- array 16
- calculations 41
- defining 18
- deleting 31
- embedded 24
- names, showing in Step Definer 119
- table 49
 - defining 48
- text 44
- using for dynamic routing 12
- validations 38
- values, copying 40

Field Definition dialog

- Field Length 18
- Field Name 18
- Field type 18
- Tables 18

Font, changing in Step Definer 117

Form

- commands 34
- creating 21
- editing 35
- using conditional text to change dynamically 31

Forwarding steps 13

G

GOTOSTEP expression 110

Group addressee 10

H

Horizontal swim lanes 85

I

- Icons, step, changing [97](#)
- Ignore suspend flag [14](#)
- iProcess
 - iProcess Engine, viewing step status on [71](#)
 - tables, using [48](#)

L

- Line length [120](#)
- Link
 - deadline [8](#)
 - withdraw [8](#), [54](#)
 - normal [7](#)
 - withdraw [8](#)
- Link icons, setting [102](#)
- Link labels, setting [102](#)
- Linking
 - objects [7](#)
- Loops
 - using waits in [72](#)

M

- Mark Field - Field Help dialog
 - Field Help [24](#)
- Mark Field dialog
 - Command [22](#)
 - Field [22](#)
 - Origin [22](#)
- Marked field
 - editing [23](#)
- Marking [49](#)
 - application fields [27](#)
 - table fields [49](#)
- Multiple user addressee [10](#)

N

- Nesting
 - levels [122](#)
- Normal link [7](#)

O

- Object label position, changing [100](#)
- Objects
 - linking [7](#)
 - types of [2](#)
- Options, Process Definer [84](#)

P

- Paste objects [6](#)
- Placing procedure objects [2](#)
- Preventing deletion on withdraw [14](#)
- Procedure
 - creating [1](#)
 - logic, simplifying with complex routers [107](#)
 - zooming in and out of [111](#)
- Process Definer
 - options [84](#)
- Process routing [9](#)
- Process Step Definer
 - configuring [115](#)

R

- Recalculating deadlines dynamically [56](#)
- Roles [11](#)
- Router
 - using [106](#)
- Routing, process [9](#)

S

- Showing field names in the Step Definer 119
- Single instance fields
 - about 16
- Single user addressee 9
- Status
 - step 71
- Step
 - icons 97
 - status 71
- Step Definer 119
 - colors 116
 - dynamic scroll setting 118
 - font 117
 - tabs 121
 - using 15, 37
- Steps
 - forwarding 13
 - withdrawing 63
- Sub-procedure
 - using
 - deadlines on 55
 - withdraw action on 67
- support, contacting xiii
- Suspend case 14, 14
- Swim lanes
 - adding 90
 - configuring 90
 - deleting 96
 - displaying 96
 - editing properties of 96
 - enabling 85
 - nesting 93
 - resizing 95
 - selecting 95
 - swapping between types 89
 - types of 85
 - using 85

T

- Table
 - field
 - defining 48
 - marking 49
 - iProcess 48
- Tabs
 - changing in Step Definer 121
- technical support xiii
- Text
 - condition, entering 31
 - field, calculating 44
- Troubleshooting procedures 125

U

- Under Construction symbol
 - troubleshooting 126
- Use files, inserting 33
- Using
 - ampersanded fields 24
 - complex routers
 - to simplify procedure logic 107
 - conditional actions 59
 - conditional text 31
 - deadlines
 - on a sub-procedure 55
 - deadlines in a procedure 51
 - embedded fields 24
 - field
 - calculations 41
 - validations 38
 - form commands 34
 - iProcess tables 48
 - routers 106
 - Step Definer 15, 37
 - swim lanes 85
 - waits
 - in loops 72
 - with conditions 79
 - with withdraws 80
 - withdraw action on a sub-procedure 67

Using GOTOSTEP [110](#)

V

Validation, field [38](#)

Vertical swim lanes [85](#)

Viewing

- step status
on the TIBCO iProcess Engine [71](#)

W

Wait

- action [71](#)
- how iProcess processes [71](#)
- using
 - in loop [72](#)
 - with conditions [79](#)
 - with withdraws [80](#)

Waits

- defining [69](#)

Withdraw

- action
 - drawing [65](#)
 - using on sub-procedure [67](#)
- link [8](#)
- using waits with [80](#)

Withdrawing steps [63](#)

Work items [14](#)

Z

Zooming in and out of a procedure [111](#)