



Using eClarus BPMN Modeler to Model an Ensemble BPL Process

Hong-Lee Yu

12/01/08

The Business Process Modeling Notation (BPMN) has been developed to enable business users to develop readily understandable graphical representations of business processes. BPMN is also supported with appropriate graphical object properties that will enable the generation of implementation such as Ensemble BPL. Intersystems Ensemble is a rapid integration and development platform that enables fast development, deployment and integration of enterprise application and Ensemble BPL is the business process language with syntax and functionality very similar to WS-BPEL. eClarus BPMN-based business process modeler enables a team of business users and technology users to model, simulate, document and implement business process using model-driven architecture (MDA). This paper presents a comprehensive example of how a BPMN diagram can be used to generate an Ensemble BPL process, using eClarus Business Process Modeler.

Contents

Introduction 3

The Example – A loan request process 3

 BPL document name 4

 Modeling BPL process properties 5

 Modeling a BPL call operation 6

 Modeling a BPL Sync operation 8

 Looping through responses 11

 Sending out the response through right channel 14

 Generating BPL from the BPMN model 16

Conclusion 16

References 17

Appendix BPL – BPMN Mapping 17

Introduction

Business Process Modeling Notation (BPMN) is the new standard for modeling business process flows. Created initially by the Business Process Management Initiative (BPMI) and later adopted by the Object Management Group (OMG), it is a rich notation to enable business users to develop readily understandable graphical representations of business processes. Further, BPMN is also supported with appropriate graphical object properties that will enable the code generation of process implementation by technology users. Thus BPMN creates a standardized bridge for the gap between the business process design and process implementation.

Intersystems Ensemble is a rapid integration and development platform that enables fast development, deployment and integration of enterprise application. One of the Ensemble key components, Business Process Language (BPL), is the executable business process language used to automate business processes. Ensemble BPL is very similar to WS-BPEL in terms of syntax and functionality.

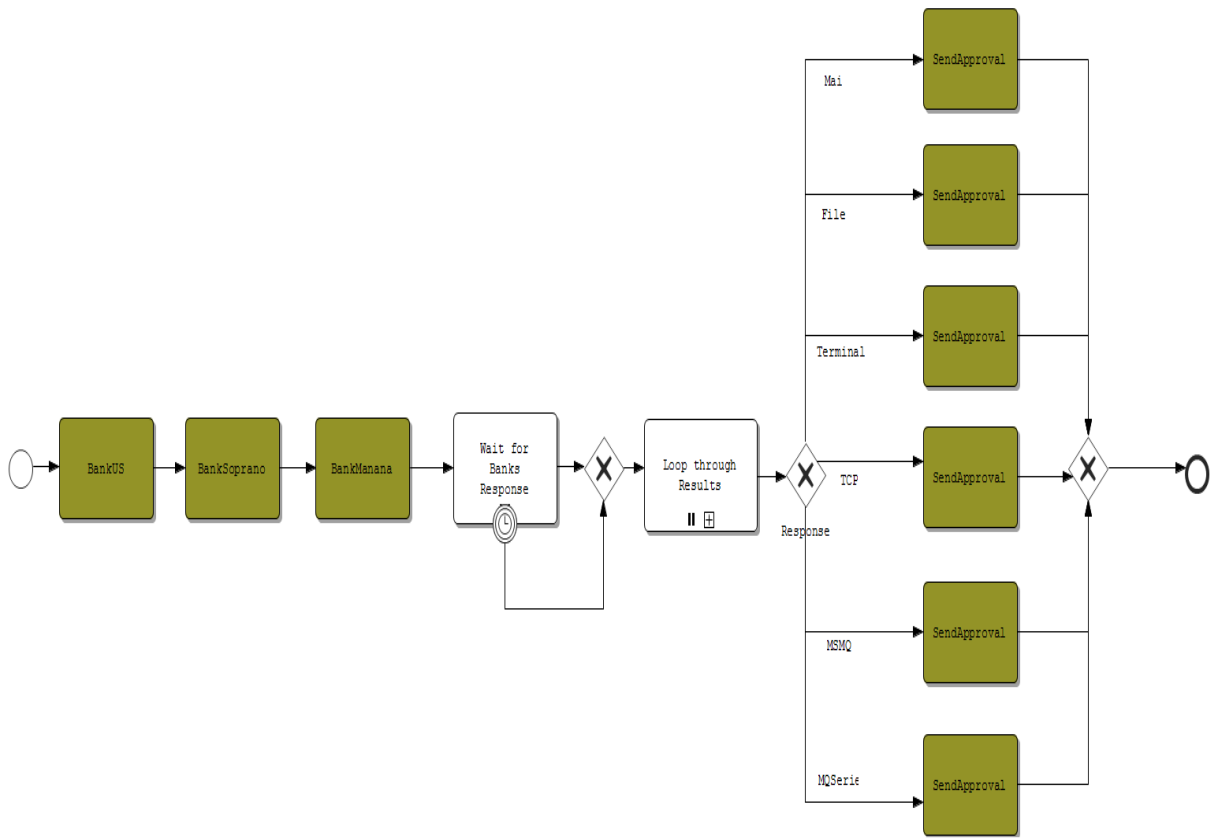
eClarus BPMN-based business process modeler enables a team of business users and technology users to model, simulate, document and implement business process using model-driven architecture (MDA). It supports multiple business process languages, including WS-BPEL 1.1, WS-BPEL 2.0 with BPEL4People and now Ensemble BPL from Intersystems.

The Example – A loan request process

This paper presents a loan request process example that shows how a BPMN diagram can be used to generate an Ensemble BPL process. The loan request process consists of the following steps:

1. Send loan requests to bank US, bank Soprano and bank Manana.
2. Wait for their responses in 5 seconds.
3. Loop through responses to find an approved loan with the best rate
4. Send out loan approval response through right channel.

The overall BPMN diagram is as follows:



BPL document name

The BPL document name maps to the process name in BPMN.

?? xml	version="1.0" encoding="UTF-8"
Export	
generator	Cache
version	20
zv	Cache for Windows (Intel) 5.2.1 (Build 509U)
ts	2007-09-03 14:08:58
Document	
name	Demo.Loan.FindRateDecisionProcessBPL.bpl
process	
request	Demo.Loan.Msg.Application
context	
sequence	

Properties | Reviews | Problems | Tasks

General | Properties | Ad Hoc

<Process> Demo.Loan.FindRateDecisionProcess

Name: Demo.Loan.FindRateDecisionProcess

Target Namespace:

Type: None

Suppress Join Failure

Enable Compensation

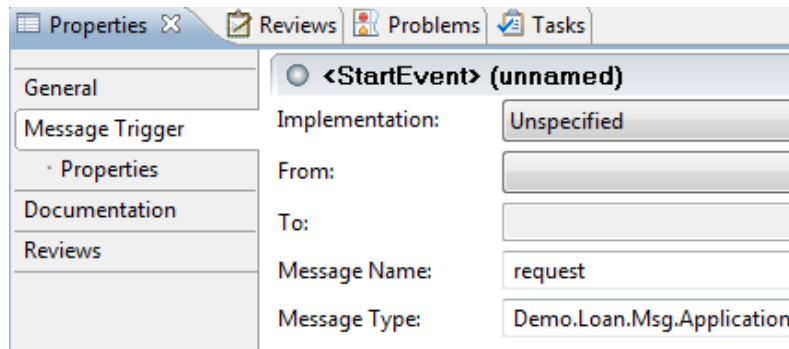
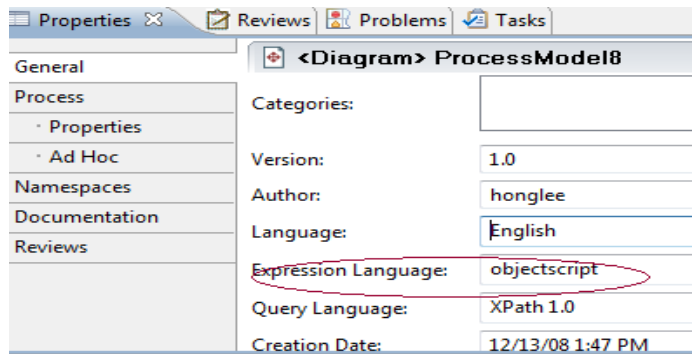
Exit On Standard Fault

Modeling BPL process properties

A BPL process defines a business process. It has language, request and response attributes that can be modeled in eClarus Business Process Modeler.

Use the message trigger in the start event to model the request attribute. Similarly, if a BPL process has response, then use end event with message trigger to model the response attribute.

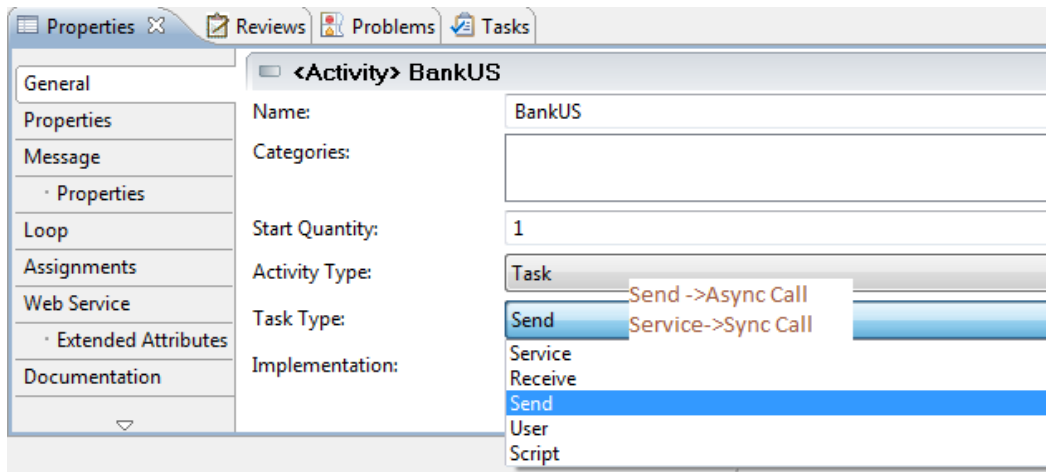
The process language in BPL can be specified using **expression language** property of the BPMN model.



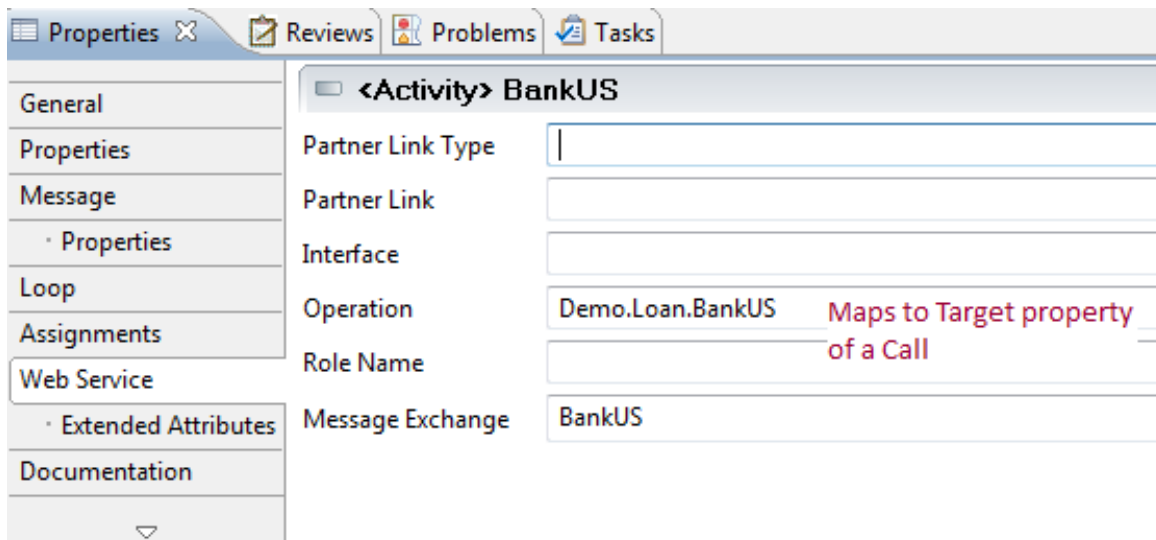
Document	
name	Demo.Loan.FindRateDecisionProcessBPL.bpl
process	
request	Demo.Loan.Msg.Application
context	
sequence	

Modeling a BPL call operation

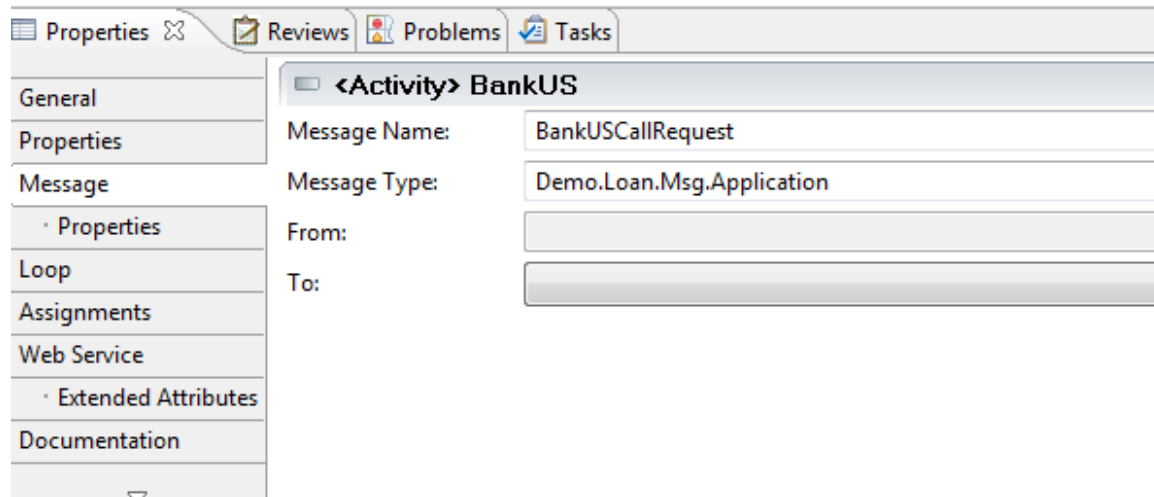
BPL “Call” is used to send a request to a business operation or to another business process synchronously or asynchronously. A synchronous call in BPL maps to a service task in BPMN while an asynchronous call maps to a send task in BPMN. In our example, BankUS, BankSorano and BankManana are three asynchronously calls to respective banks to get a loan approval.



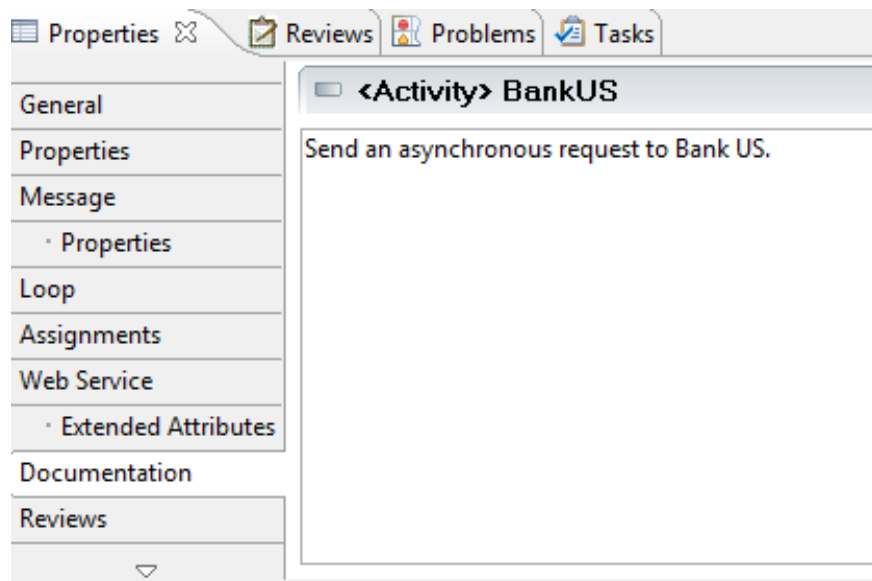
A BPL call requires the specification of target attribute, which is the configured name of the business operation or business process to which the request is being sent. eClarus uses the operation name in Web Service to specify the target attribute.



Use in-message and out-message in BPMN to model request and response message respectively. In BPL, the names of request message and response message are always named as “request” and “response”. In BPMN, you can name message variables anyway you want.



To make business process understandable, business users often document each business process diagram and diagram objects with additional annotations. In BPMN, each diagram object has documentation property that maps to the annotation attribute in BPL



The following is the snippet for the generated BPL.

▲ e call	
ⓐ async	1
ⓐ name	BankUS
ⓐ target	Demo.Loan.BankUS
e annotation	Send an asynchronous request to Bank US.
▶ e request	

Modeling a BPL Sync operation

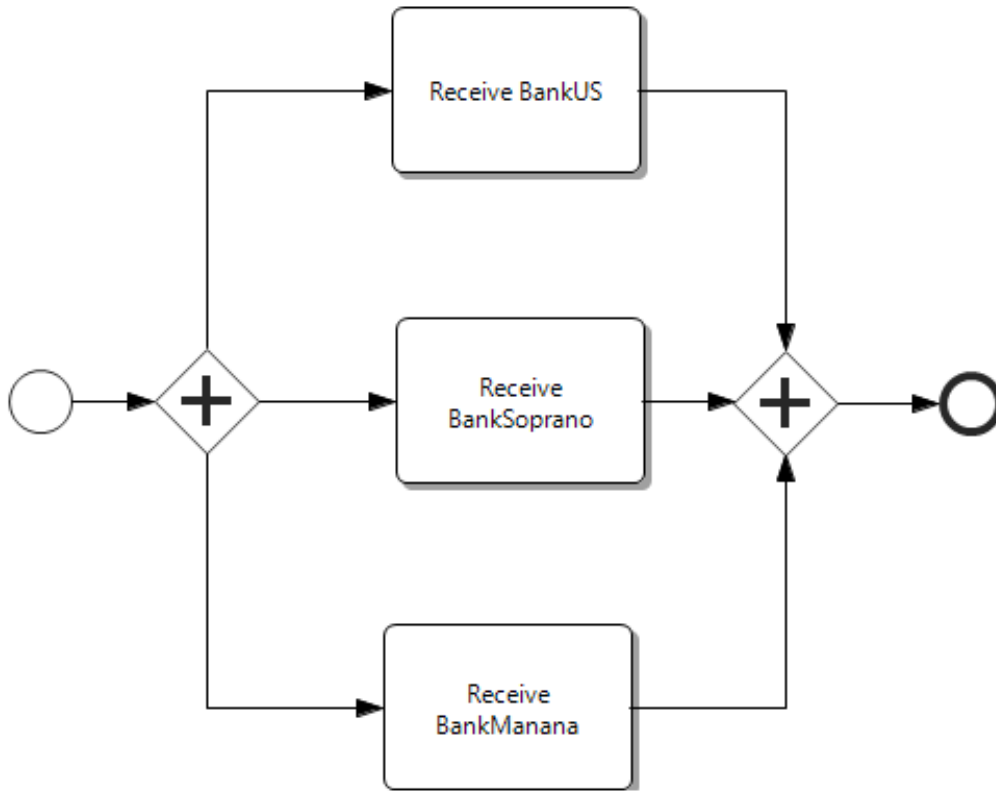
The sync operation is used to wait for responses from one or more asynchronous calls. It has several required attributes.

The calls attribute includes a list of asynchronous call.

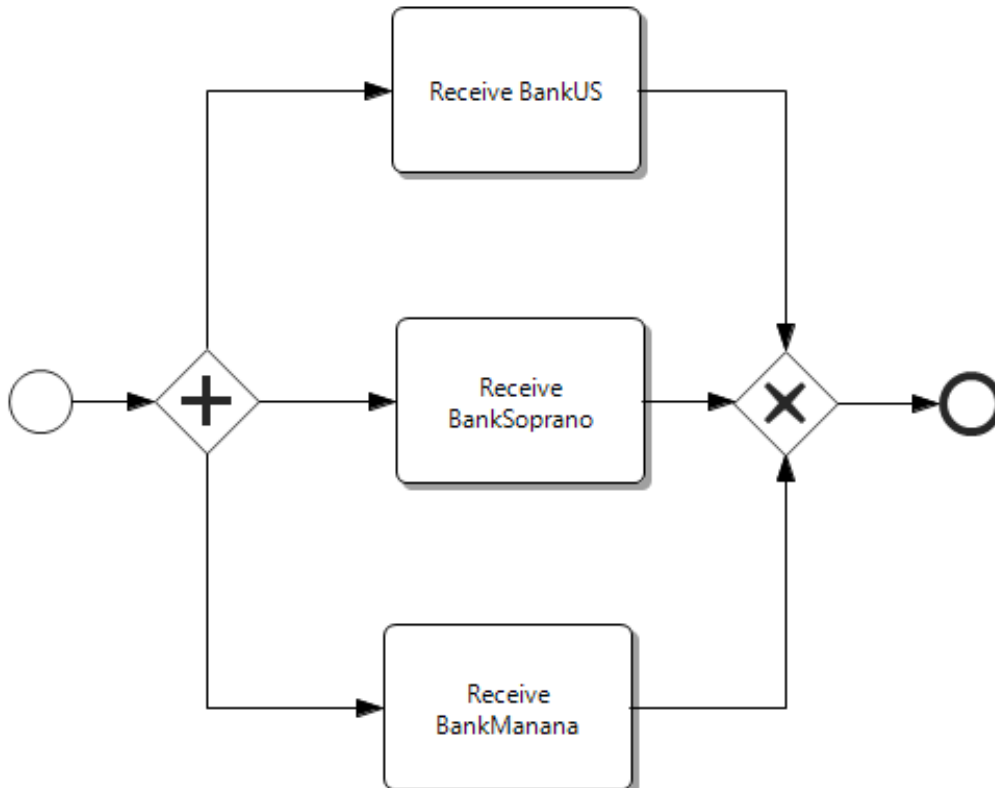
The type attribute either has the value “all” , which specifies that the <sync> should wait for a response from all calls, or “any” , which specifies that it will only wait for the first response it receives (in this case, the remaining responses are discarded in the same manner as an expired timeout).

The sync operation can optionally contain a timer.

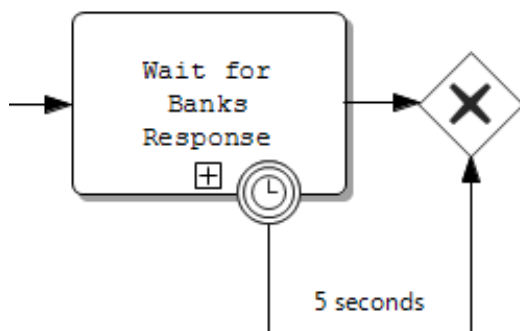
Sync Type = "All"



Sync Type = "any"



If the sync operation has time-out attribute, you can model a sub-process with a timer intermediate event.

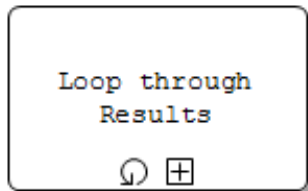


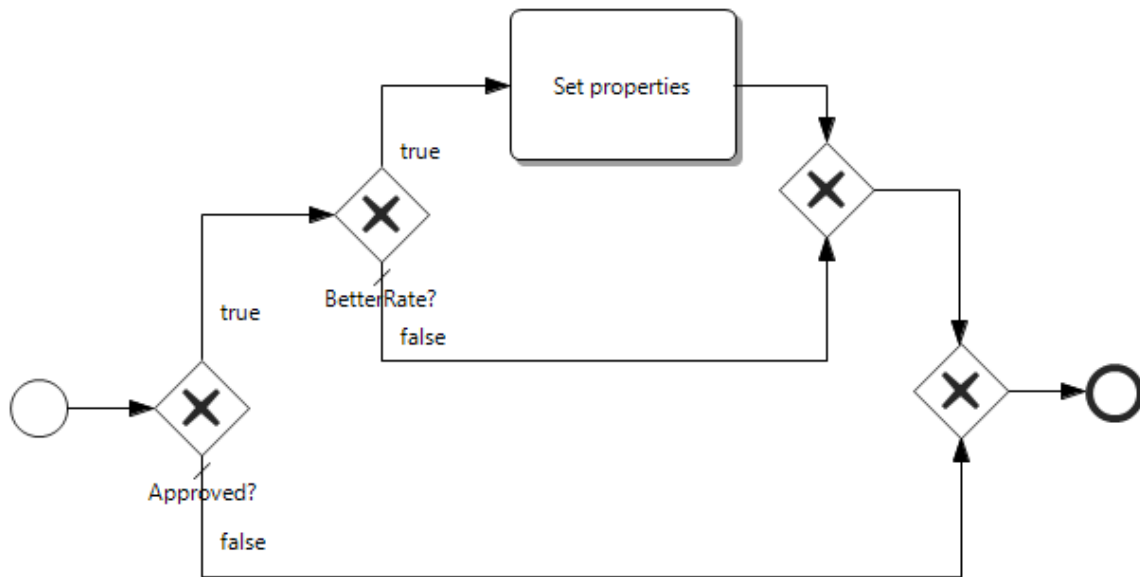
Here is the resulted generated BPL code for “Wait for Banks Response”.

▲ [e] sync	
ⓐ name	Wait for Banks Response
ⓐ calls	BankUS, BankSoprano, BankManana
ⓐ type	all
ⓐ timeout	5
▶ [e] annotation	

Looping through responses

You can use a BPMN sub-process with loop marker to model the BPL “foreach” operation that defines a sequence of activities to be executed iteratively.



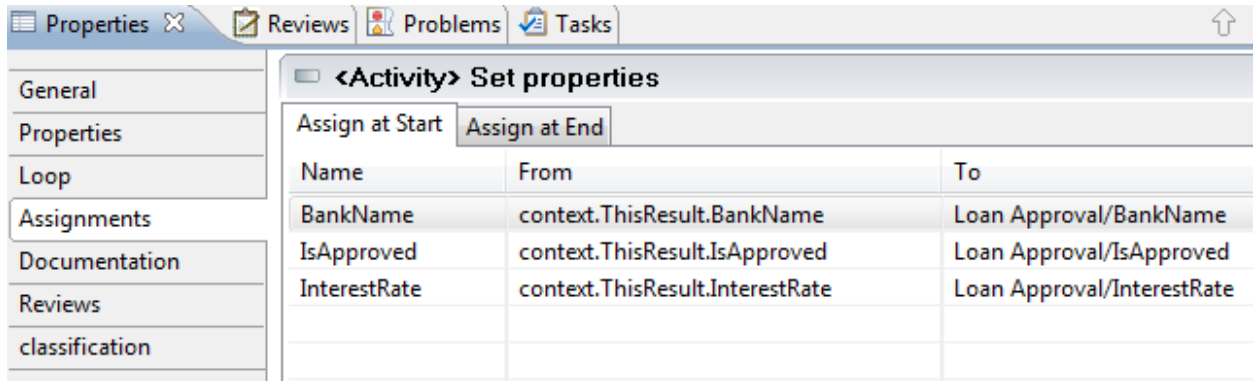


The sub-process first checks whether the loan request is approved. If it is true, then it further checks whether the response has the better rate. In order to implement that, this business process needs to maintain a list of context variables. Use “assign” to set the value of a variable.

BPL context variables map to BPMN process properties.

<Diagram> Loan Approval			
Name	Type	Type	Type
BankName	%String	Type	Type
IsApproved	%Boolean	Type	Type
InterestRate	%Numeric	Type	Type
TheResults	Demo.Loan.Msg.Approval	Type	Type
Iterator	%String	Type	Type
ThisResult	Demo.Loan.Msg.Approval	Type	Type

Both BPL and BPMN have “assign” object that set properties or messages value. The BPMN activity “Set properties” contains three assignments



Here is the generated BPL for context variables.

context	
property	
name	BankName
type	%String
property	
name	IsApproved
type	%Boolean
property	
name	InterestRate
type	%Numeric
property	
name	TheResults
type	Demo.Loan.Msg.Approval
collection	list
property	
name	Iterator
type	%String
property	
name	ThisResult
type	Demo.Loan.Msg.Approval

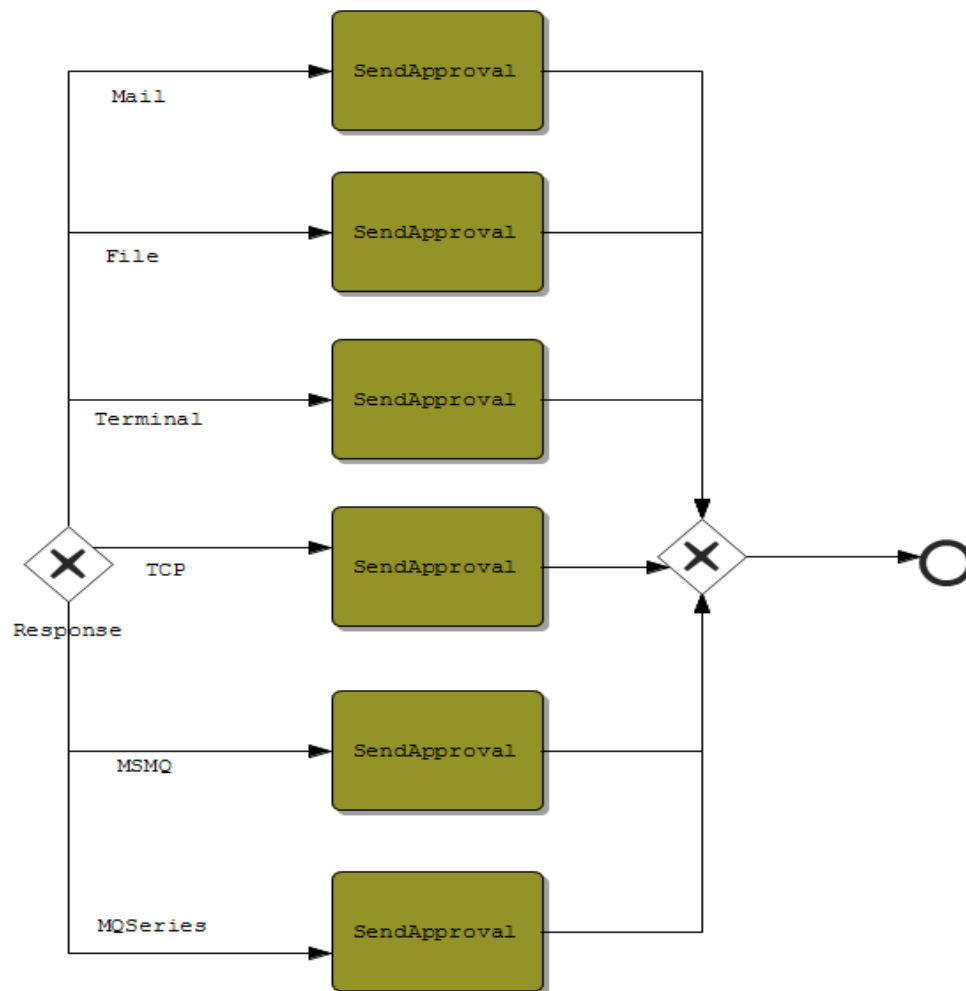
Here is the generated BPL for looping through results:

<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> ⓐ name ⓐ property ⓐ key ▷ [e] annotation ▷ [e] assign ▷ [e] trace <ul style="list-style-type: none"> <ul style="list-style-type: none"> ⓐ name ⓐ condition ▷ [e] annotation <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> ⓐ name ⓐ condition ▷ [e] annotation <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> ⓐ name ⓐ property ⓐ value ▷ [e] annotation <ul style="list-style-type: none"> <ul style="list-style-type: none"> ⓐ name ⓐ property ⓐ value <ul style="list-style-type: none"> ⓐ name ⓐ property ⓐ value 	<ul style="list-style-type: none"> Loop through Results context.TheResults context.Iterator Approved? context.ThisResult.IsApproved BetterRate? ('context.IsApproved') (context.InterestRate> context.ThisResult.InterestRate) BankName context.BankName context.ThisResult.BankName IsApproved context.IsApproved context.ThisResult.IsApproved InterestRate context.InterestRate context.ThisResult.InterestRate
--	--

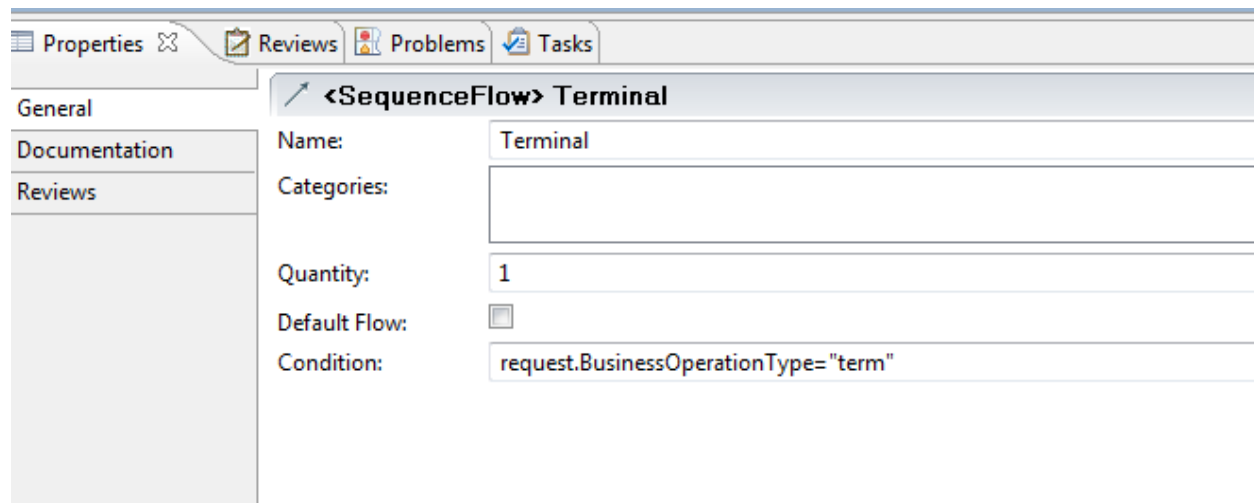
Sending out the response through right channel

After finding the best loan rate from banks, the business process sends out the loan request response through right channel, which is specified in the original request (request.BusinessOperationType).

Use BPMN data-based exclusive gateway to implement a “switch” predicate in BPL.



The branch condition is specified in each sequence flow.

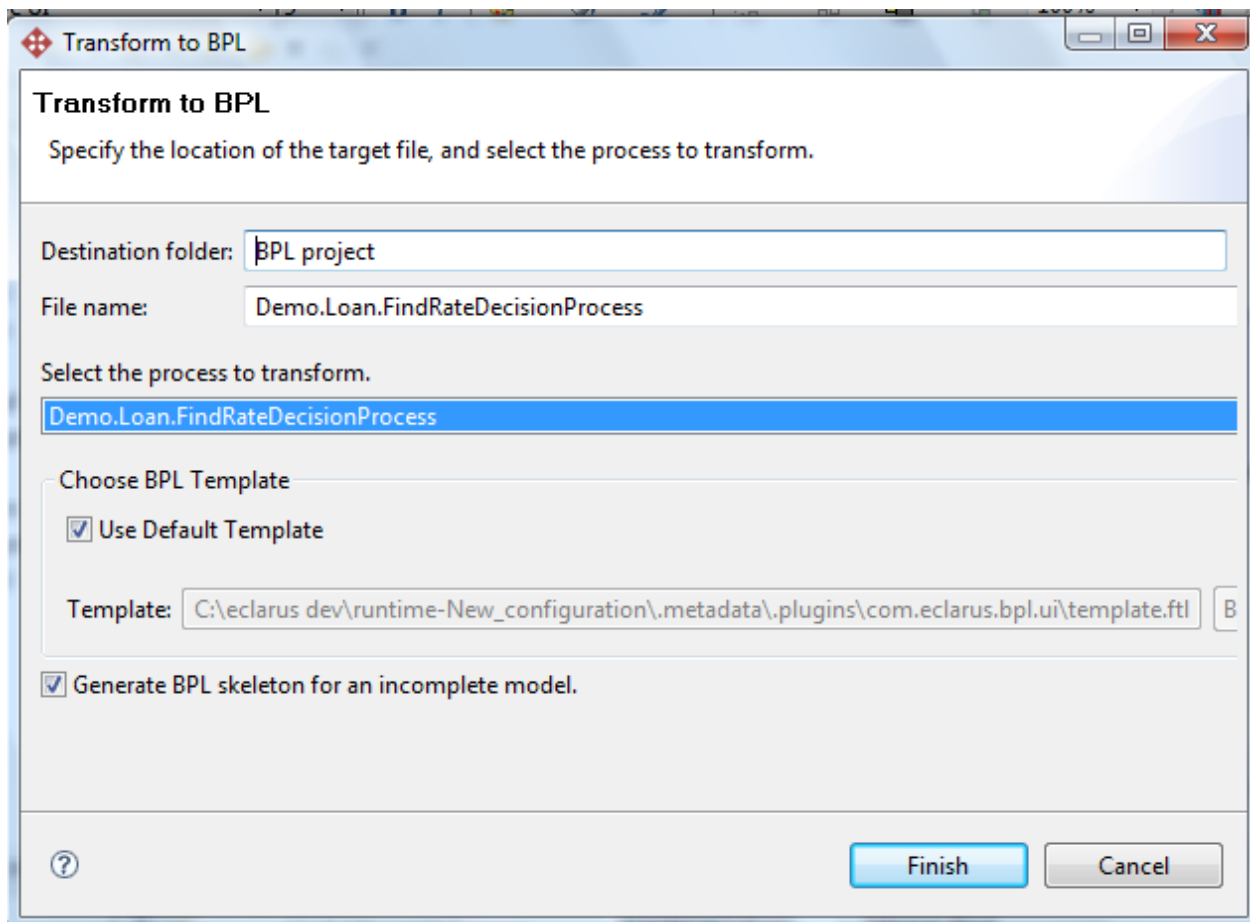


And the generated BPL is as follows:

▲ [e] switch	
ⓐ name	Response
▷ [e] annotation	
▲ [e] case	
ⓐ name	Mail
ⓐ condition	request.BusinessOperationType= "mail"
▷ [e] call	
▲ [e] case	
ⓐ name	File
ⓐ condition	request.BusinessOperationType= "file"
▷ [e] call	
▲ [e] case	
ⓐ name	Terminal
ⓐ condition	request.BusinessOperationType= "term"
▷ [e] call	
▲ [e] case	
ⓐ name	TCP
ⓐ condition	request.BusinessOperationType= "tcp"
▷ [e] call	
▲ [e] case	
ⓐ name	MSMQ
ⓐ condition	request.BusinessOperationType= "msmq"
▷ [e] call	
▲ [e] case	
ⓐ name	MQSeries
ⓐ condition	request.BusinessOperationType= "mqseries"
▷ [e] call	

Generating BPL from the BPMN model

Use the “Transform to BPL” wizard to generate BPL from the model. You can specify the project name and file name in the wizard. The wizard has two additional options; an option to allow you to generate an incomplete BPL that must be edited before deploying to Ensemble server and an option to select a BPL template for its BPL header and footer – a default template is provided.



The screenshot shows a Windows-style dialog box titled "Transform to BPL". The dialog has a title bar with standard minimize, maximize, and close buttons. The main content area is divided into several sections:

- Transform to BPL**: The title of the dialog, followed by the instruction: "Specify the location of the target file, and select the process to transform."
- Destination folder:** A text input field containing "BPL project".
- File name:** A text input field containing "Demo.Loan.FindRateDecisionProcess".
- Select the process to transform.**: A list box containing one item, "Demo.Loan.FindRateDecisionProcess", which is highlighted with a blue selection bar.
- Choose BPL Template**: A section containing a checked checkbox labeled "Use Default Template".
- Template:** A text input field containing the path "C:\eclarus dev\runtime-New_configuration\metadata\plugins\com.eclarus.bpl.ui\template.ftl".
- Generate BPL skeleton for an incomplete model.**: A checked checkbox.

At the bottom of the dialog, there is a help icon (a question mark in a circle) on the left, and two buttons: "Finish" and "Cancel" on the right.

Conclusion


This paper provides an example of how a BPMN Business Process Diagram can be used to model an Ensemble BPL process. To create the executable BPL process, the diagram objects and their properties are dissected and then mapped into the appropriate Ensemble BPL elements.



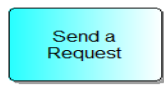


Together, eClarus business process modeler and Ensemble accelerate the implementation of business process solutions, allowing customers to introduce new products and services more quickly, and respond more nimbly to continually changing business demands. It is agile because business processes now can be specified using the standard notation - BPMN and IT users can use the same model to generate 100% integration code code-free. In addition, business users and IT users can identify business services needed that will facilitate the component reuse.

References

1. [Business Process Modeling Notation Specification 1.0, OMG Final Adopted Specification](#), February, 2006.
2. [Using BPMN to Model a BPEL Process](#), Stephen White - February, 2005.
3. [eClarus Business Process Modeler Reviewer's Guide](#), August 2008
4. [Ensemble Business Process Language Reference](#), Version 4.0, July 2006







Appendix BPL – BPMN Mapping

BPL Object	BPMN Object	Comments
request	message 	Referenced by start event

response	<p>Message</p>  <p>Response</p>	Referenced by end event
context (list of context variables)	<p>Process.properties</p>	
call (sync)	<p>task</p> 	Task type = "Service"
call (async)	<p>task</p> 	Tasktype = "Send"
foreach	<p>sub-process</p> 	With Loop marker
if	<p>Data-based exclusive gateway</p> 	
switch	<p>Data-based exclusive gateway</p>	A pair of data-based exclusive gateways with one of the activities to be selected based on the condition

flow	<p>Parallel gateway</p>	<p>A pair of parallel gateways with list of activities to be executed in parallel</p>
sequence	<p>None</p>	<p>A list of activities to be executed in sequence (connected by sequence flow)</p>
delay	<p>Intermediate event</p>	<p>With timer trigger</p>
branch	<p>Sequence flow</p>	<p>Optionally conditional sequence flow to an element identified by its LABEL (name)</p>

	<pre> graph LR A[Receive a loan] -- "If loan request > 20K" --> B[Assess] </pre>	
SQL	<p>Task</p>	<p>Task type = "script"</p> <p>Language = "SQL"</p>
transform	<p>Task</p>	<p>Task type = "transform"</p>
rule	<p>Task</p>	<p>Task type = "rule"</p>
code	<p>Task</p>	<p>Task type = "script"</p> <p>Language = "objectscript", or "basic"</p>
until	<p>Sub-process</p>	<p>With Loop marker</p>

while	<p>Sub-process</p> 	With Loop marker
break	<p>End event</p> 	
continue	<p>End event</p> 	
reply	<p>End event</p> 	
alert	<p>Intermediate event</p> 	With message trigger
empty	<p>Task</p> 	Task type "none"
sync	pattern	See example above
assign	assignment	

