Business Object Process Modeling Specification

Version of the document: 0.9

Date: 21.4.2013-15.5.2013

Standard document URL: www.bopm.org/specifications-v-1-0.pdf

Associated Normative Machine-Readable Files:

notice if you breach any of these terms or conditions. BOPMG may change the license in the future to require a fee from the use of the specifications. In such case all products that implemented this specification or were based on the specifications deducted from this are subject of this fee. Upon termination, you will destroy immediately any copies of the specifications in your possession or control.

GENERAL USE RESTRICTIONS

Any unauthorized use of this specification may violate copyright laws, trademark laws, and communications regulations and statutes. This document contains information which is protected by copyright. All Rights Reserved. No part of this work covered by copyright herein may be reproduced or used in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems; without permission of the copyright owner.

DISCLAIMER OF WARRANTY

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE BUSINESS OBJECT PROCESS MODELING GROUP AND THE COPYRIGHT HOLDERS ABOVE MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE. IN NO EVENT SHALL THE BUSINESS OBJECT PROCESS MODELING GROUP OR ANY OF THE COPYRIGHT HOLDERS ABOVE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The entire risk as to the quality and performance of software developed using this specification is borne by you. This disclaimer of warranty constitutes an essential part of the license granted to you to use this specification.

RESTRICTED RIGHTS LEGEND

The specification copyright owners are as indicated above and may be contacted through the BOPMG - Business Object Process Modeling Group, info@bopm.org.

COMPLIANCE

The copyright holders listed above acknowledge that the Business Object Process Modeling Group (acting itself or through its designees) is and shall at all times be the sole entity that may authorize developers, suppliers and sellers of computer software to use certification marks, trademarks or other special designations to indicate compliance with these materials.

Software developed under the terms of this license may claim compliance or conformance with this specification if and only if the software compliance is of a nature fully matching the applicable compliance points as stated in the specification. Software developed only partially matching the applicable compliance points may claim only that the software was based on this specification, but may not claim compliance or conformance with this specification. In the event that testing suites are implemented or approved by BOPMG, software developed using this specification may claim compliance or conformance with the specification only if the software satisfactorily completes the testing suites.

BOPMG's Issue Reporting Procedure

All BOPMG specifications are subject to continuous review and improvement. As part of this process we encourage readers to report any ambiguities, inconsistencies, or inaccuracies they may find by completing the Issue Reporting Form listed on the main web page http://www.bopm.org, under Documents, Report a Bug/Issue.

Table of contents

1 Scope

This specification defines Business Object Process Modeling Language (BOPM), revision 1.0. The objective of BOPM is to provide any interested person in system modeling with tools for analysis, design, and implementation of software based systems as well as for modeling business and similar processes.

One of the primary goals of BOPM is to advance the state of the industry by enabling object visual modeling tool interoperability. However, to enable meaningful exchange of model information between tools, agreement on semantics and notation is required. BOPM meets the following requirements:

- A formal definition of metamodel
- A specification for the human-readable notation
- XML-based specification

The **purpose** of BOPM is to create the **executable** application from the **easy to model** concept. The BOPM should be understandable by every participant – from management, through accountant to the system analysts.

1.1 RFC-2119

The upper case keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119. [1]

[1] http://www.ietf.org/rfc/rfc2119.txt

2 BOPM

2.1 Definitions

BOPM	Business Object Process Model is notation, set of rules, set of best practices, and schema for process modeling in object environment. In some cases means Business Object Process Model Language. In some cases means Business Object Process Modeling Group.
Object	Class as defined in UML. Object under BOPM MUST have defined set of variables and set of functions. BOPM does not rely on specific object definition. If object is defined as Entity with possibility to define set of functions and set of variables, this definition should be usable for use of the BOPM, however with already existing tools, the BOPM should be easy to manage.
Process object	Object where it is possible to track changes in the states of Object.
Process state	Process state is a state of a Process object. It express a situation of the Process object that takes some time while a Process object does not change its meaningful situation.
Process action	Process action is an action that changes the Process state of a Process object. Process action expresses an activity of authorized Process role who is allowed to change the Process state of the Process object. Action can be performed by a person, script, or event on behalf of the Process role.
Timeout action	Action executed in time out after no other Process action was performed.
Process role	Role who is allowed to execute the Process action and/or can receive Process messages.
Process message	Message to the user of the process role who should be informed about the development of the Process object
Process	Oriented graph of states of the Process object.
Process overview	Graphical representation of one Process for selected Process role or Timeout.

2.2 Process object

Process object is *Object* with defined attribute "state" (*Process state*). *Process* is subset of *Process states* of the *Process object*. One *Process object* can have several *Processes* defined, but one instance of the *Process object* cannot be in one time in two different *Process states*. Each *Process state* must have the *Process object* defined and cannot be in two process objects in one time. This however does not exclude possibility of naming two *Process states* of two different *Process objects* the same.

2.3 Conformance

BOPM is a language with strict rules. It should be possible to interchange the models between BOPM, and other process business driven languages.

2.3.1 Compliance levels

BOPM can be used for business modeling as well as for implementation of information systems. To differentiate between these two approaches, the compliance levels were set.

2.3.1.1 BOPM Compliance Level 1

BOPM compliance level 1 is for **analysis** purposes. Process overview must be shown using these methods:

1. Table overview: For selected *Process role* of *Process states* and *Process actions* with arrows indicating the ability of *Process action* to change the *Process state* of the *Process object* to the different *Process state*;
2. Text table overview: List of process states modifiers with textual representation for < state: (TEXT) -> action [optional (Process role)]: (TEXT) -> new state: (Text) >;
3. or as oriented graph between *Process states* with *Process actions* in between.

Process overview SHOULD display the *Compliance level 1 Icons* defined in this specification.

2.3.1.2 BOPM Compliance Level 2

BOPM compliance level 2 is executable application with defined *Process messages* for every *Process role* for every *Process object state*.

Process overview MUST be displayed in the table overview with defined Process role as defined in Compliance level 1. Process state must be active link and *Current process state overview* must be displayed on activation. Process overview MUST display Compliance level 1 and Compliance level 2 icons.

2.3.1.3 BOPM Compliance Level 3

BOPM compliance level 3 is executable application with multilingual support.

Process overview MUST be displayed the same way as for Compliance level 2. Process overview MUST display Compliance level 3 icons.

2.4 Texts

BOPM is object state process system. Texts MUST be defined in exprehensive language. Process states represent the time between the state is initialized and time the state ends. Therefore the state should be described in present perfect tense. For example “User is submitting application”, “Contract is waiting for time of the settlement”, “Invoice is waiting for invoice number”,… Process states marked with the EndPoint or Terminate flags should be described in past tense. For example: “Client has received full Membership”, “Invoice has been paid”…

Process actions represent the action of the Process role that can change the Process state of the Process object to the different Process state. Therefore the text should contain the process role and action that it can perform. For example: “User can submit application”, “Client can register new company”, “Buyer is allowed to choose guarantee”…

Timeout actions texts SHOULD inform about the elapsing time. For example: “When the time is up, the invoice will be marked as unpaid”, “When the time is up, client’s registration will be canceled”…

2.5 Message central

Information system under BOPM MUST have message central, where user can find his or her messages from information system and can interact with the running processes. In the message central user SHOLD be informed which processes he should suppose to influence currently so that they can be executed in the standard non error flow.

2.6 Process central

Information system under BOPM MUST have process central, where user can find all processes he or she can initiate in the moment. If user can not initiate any processes, process central can be hidden from him or her.

2.7 XML Model

In compliant information system, the processes MUST be exportable and importable. This specification defines exact XML Schema the information system must be able to import and export. Namespace in the XML files for standard process structure MUST be “http://bopm.org/xml/process”. Schema is available at http://bopm.org/xml/process/schema.xsd.

2.7.1 Root element

Root element is Process. It gathers all information about one process. Please note that one Process object can consists of several Processes, and the imports MUST be able to handle multiple imports of Processes for one Process object.

Root element CAN contain any of following elements in any order:

- pr:ProcessRole
- pr:ProcessState
- pr:Action
- pr:ProcessState2Action
- pr:Timeout
- pr:ProcessState2Timeout

2.7.2 ProcessRole

Process role represents the Role who is allowed either to execute Process actions or to receive system message for the current selected process. Process role may be optional which means that user MAY receive the system message for some process states. When optionality is set to “always”, the role MUST receive the system message for every state of the initialized Process object.

ProcessRole MUST define id3 attribute, what is the function of the Process object that selects users. Function MUST return array of Users. Name of the function defined in the Process object usable in the ProcessRole must start with “role”. The starting text “role” MUST not be included in the id3 attribute.

ProcessRole MUST contain element name what is type of pr:text, and must contain at least one pr:dictionary element. For example: <pr:name> <pr:dictionary xml:lang="sk">Text</pr:dictionary></p:name>

Required attributes:

- pr:id3 – Defines Process object function that selects users

Optional elements:

- pr:optionality – ENUM: always|sometimes – default always – Always means the message is required for the
- pr:processid – char(32) – process identifier

Required elements:

- pr:name – pr:TypeText – Name of the process role in human readable form.

2.7.3 ProcessState

Process state represents one state of the Process object.

Required text parameter is name, what should be human readable name of the Process state. Element must contain at least one pr:dictionary entry with text in it.

Required function parameters are: beforestart, afterstart, onactive, beforeend, and afterend. Usable functions of the Process object must start with "run". Element MUST NOT contain this starting string "run" but the rest of the function name.

Process state may contain this flags: flagentrypoint, flagendpoint, check1, check2, flagnewprocess, flagnewprocesses, flagsynchronisation, flagthrowevent. The values of these elements CAN be either 0 or 1. These elements are optional and default value for each of them is 0 as FLAG NOT SET.

Process state MAY contain element processid what represents identifier of the Process.

Optional attributes:

- pr:id – char(32) - identifier of the ProcessState
- pr:id3 – varchar(250) - Human readable identifier of the Process state that can be used in the functions. When id3 is defined, in whole document the reference to this Process state should be used with using the id3.

Required elements:

- pr:name - pr:TypeText – Name of the process state in human readable form
- pr:Message – pr:MessageType – Messages to be sent to the process role users when a process state is activated
- pr:Attachment - pr:Attachment – Attachment to be sent to the process role users when a process state is activated

Optional elements:

- pr:beforestart – FCT – "run" – Function of the Process Object that is lunched before the Process state can change to this Process state.
- pr:afterstart – FCT – "run" – Function of the Process Object that is lunched after the Process state changes to this Process state.
- pr:onactive – FCT – "run" – Function of the Process Object that can catch the Events.
- pr:beforeend – FCT – "run" – Function of the Process Object that is lunched before the current state is changed.

- pr:afterend – FCT – "run" – Function of the Process Object that is lunched after the current state is changed.
- pr:flagentrypoint – ENUM: 0|1 – default 0 – Compliance level 1 flag EntryPoint
- pr:flagendpoint – ENUM: 0|1 – default 0 – Compliance level 1 flag EndPoint or Terminate
- pr:check1 – ENUM: 0|1 – default 0 - Error3 flag for Compliance level 3
- pr:check2 – ENUM: 0|1 – default 0 - Error4 flag for Compliance level 3
- pr:flagnewprocess – ENUM: 0|1 – default 0 – Compliance level 1 flag NewProcess
- pr:flagnewprocesses – ENUM: 0|1 – default 0 – Compliance level 1 flag NewProcesses
- pr:flagsynchronisation – ENUM: 0|1 – default 0 – Compliance level 1 flag Syncrhonisation
- pr:flagthrowevent – ENUM: 0|1 – default 0 – Compliance level 1 flag ThrowEvent
- pr:processid – char(32) – process identifier

If flagnewprocess or flagnewprocesses is set to true, required element:

- NewProcessesList – List of processes that can be launched by the Processes role

2.7.4 ProcessAction

ProcessAction element represents the Process action but not the Process timeout.

Required attributes:

- pr:id – char(32) - identifier of the ProcessAction

Optional attributes:

- pr:id3 – varchar(250) - Human readable identifier of the Process action that can be used in the functions. When id3 is defined, in whole document the reference to this Process action should be used with using the id3.

Required elements:

- pr:party – varchar(250) – identifier of the *Process role*
- pr:text – pr:TypeText - Text of the Process Action
- pr:buttontext - pr:TypeText - Text of the Button
- pr:weight – number – The weight of the default flow through this Process action
- pr:next – varchar(250) – id or id3 of the following Process state

Optional elements:

- pr:flagerror - ENUM: 0|1 – default 0 – Compliance level 1 flag Error
- pr:flagcatchevent - ENUM: 0|1 – default 0 – Compliance level 1 flag CatchEvent

- pr:flagparameters- ENUM: 0|1 – default 0 – Compliance level 1 flag Parameters
- pr:flagerror3 - ENUM: 0|1 – default 0 – Compliance level 3 flag Error3
- pr:flagerror4 - ENUM: 0|1 – default 0 – Compliance level 3 flag Error4

If flagparameters == 1, Required elements:

- pr:parameters – pr:ParamentersListType – List of parameters

2.7.5 Message

Message is a message that is sent to the Process role when the current Process state is activated. The process state is defined by the parent element of Process state. Message MUST be customized for every process state for every relevant Process role.

Optional attributes:

- pr:id – char(32) - identifier of the Message

Required elements:

- pr:party – varchar(250) – identifier of the Process role
- pr:message – typeText – the text that will be send to the users

2.7.6 ProcessState2ProcessAction

Because the relation between ProcessState objects and ProcessActions objects is NxM, in the ProcessState2ProcessAction is all possible combinations of ProcessActions from the ProcessStates.

Optional attributes:

- pr:id – char(32) - identifier of the ProcessState2ProcessAction

Required elements:

- pr:ProcessState - varchar(250) – identifier of the Process state. id3 if set, or id if id3 not set
- pr:ProcessAction - varchar(250) – identifier of the Process action. id3 if set, or id if id3 not set

2.7.7 Timeout

Timeout is Process action that is executed when the exact time is passed. The compliant information system MUST show the countdown of the time elapsed for the active ProcessState.

Optional attributes:

- pr:id – char(32) - identifier of the ProcessState2ProcessAction
- pr:id3 – varchar(250) – human readable identifier of the Timeout for use in function

Required elements:

- pr:text – pr:TypeText - Text of the Timeout action
- pr:weight – number – The weight of the default flow through this Process action
- pr:mintime – type:TimeValue – Minimum time value as defined in 2.7.10
- pr:stdtime – type:TimeValue – Standard time value as defined in 2.7.10
- pr:maxtime – type:TimeValue – Maximum time value as defined in 2.7.10
- pr:next – varchar(250) – id or id3 of the following Process state

Optional elements:

- pr:processid – char(32) – process identifier

2.7.8 ProcessState2Timeout

Because the relation between ProcessState objects and Timeout objects is NxM, in the ProcessState2Timeout is all possible combinations of Timeouts from the ProcessStates.

Optional attributes:

- pr:id – char(32) - identifier of the ProcessState2Timeout

Required elements:

- pr:ProcessState - varchar(250) – identifier of the Process state. id3 if set, or id if id3 not set
- pr:Timeout - varchar(250) – identifier of the Timeout. id3 if set, or id if id3 not set

2.7.9 Attachment

Process analysts have option to send a default file for a process role when the Process state is activated. This attachment MUST be displayed in the Messages central, and in the case of sending emails to the users, it MUST be attached in the email.

The attachment however does not influence the ability to display dynamic files in the Messages.

Optional attributes:

- pr:id – char(32) - identifier of the ProcessState2Timeout

Required elements:

- pr:party – varchar(250) – identifier of the Process role
- pr:file – fileType – File to be attached with the Process state.

2.7.10 Parsing timeout's time value

Timeout has three different times definitions. Minimum time, standard time, and maximum time. First priority in time evaluation has the minimum time, second priority has the maximum time, and third priority has the standard time. The highest priority sets the time if it is after the maximum time or before the minimum time.

The time in the compliant information system must be able to parse the time value.

First item in the time value is positive if sign minus is not first character. Plus sign means add to the current time the value that follows, minus sign means to subtract the value that follows.

The value of time can be either:

- T represents time when the Process state has been activated
- H represents the holidays between the current time and already evaluated time
- Function calculation – FCT of the Process object must start with "date", and this function name MUST be present in the time value without the "date" string, and encapsulated with % marks.
- Time expression

Time expression must start with a digit [0-9] and must be numeric. Time expression must end with one of the following:

- s – for seconds,
- m – for minutes (60s),
- h – for hours (60m),
- d – for days (24h),
- w – for weeks (7d).

2.7.11 Functions

Functions in BOPM are specific designed algorithms to lunch new objects, wait for events, modify object's variables, oversight decision points, and similar. Compliant process model system SHOULD NOT rely on the functions, and MUST allow human interaction in every Process state. Functions are add on which speeds up the change of Process states and provide more simplistic user interaction.

These types of functions can be used in BOPM

Function type	Identifier
Algorithmic functions	runal
Conditional functions	runif
Throwable conditional functions	runtif

Function type	Identifier
Set functions	set
Variable functions	get
Parameters functions	paramget
Date functions	date
Role functions	role

2.7.11.1 Algorithmic functions

Algorithmic functions usually perform computing tasks, modification of the object variables, or lunching new Process objects.

Algorithmic functions SHOULD return true on successful execution.

Algorithmic functions name MUST start with string “runal”.

Algorithmic functions MUST NOT throw exceptions. All exceptions must be taken care of in the function.

Algorithmic functions are: ProcessObject.afterstart, ProcessObject.onactive, ProcessObject.afterend, ProcessAction.execute_before

2.7.11.2 Conditional functions

Conditional functions are functions that checks if the action CAN or CAN NOT be executed or visible.

Conditional functions return TRUE or FALSE. Function returns TRUE, if the action CAN be executed, or FALSE if action CAN NOT be executed. For example function that checks if the user has enough money on the account must return true if he has.

Conditional functions MUST NOT throw exceptions. All exceptions must be taken care of in the function.

Algorithmic functions name MUST start with string “runif”.

Conditional functions are: ProcessAction.execute_if

2.7.11.3 Throwable conditional functions

Throwable conditional functions are functions **that** can prevent the *Process state* to become active, or leave the *Process state*.

Throwable conditional functions are: ProcessObject.beforestart, ProcessObject.beforeend

Throwable algorithmic functions name MUST start with string “runtif”.

When beforestart function returns TRUE, the system expects that the function has activated different *Process state*, and currently selected *Process state* MUST NOT be activated. When beforeend function returns TRUE, the system expects that the function has taken care of finalizing the active *Process state*, and lunched new *Process state*.

Conditional functions MAY throw exceptions. Exception MUST prevent change of the active *Process state*. Exception text MUST be shown to the user so that he is aware why the *Process state* did not change.

2.7.11.4 Set functions

Set functions save the value of the parameters.

Set function can throw an Exception. If the function throws an exception, the change of state MUST NOT occur, and this exception must be shown to the user as the reason why the state has not changed.

Example of using the set function is filling a date by the process role while it wants to change the process state. If the date is lower than for example current date, the function may throw an exception saying that the date may be only future day.

Set function MUST start with string “set”.

2.7.11.5 Variable functions

Variable functions return the variable that is shown in the Message. The Variable function MUST return value with type of string.

Example of using the variable function: The function returns anchor reference to the invoice file issued for the buyer.

Variable function MUST start with string “get”.

Variable functions MUST not throw exceptions, and must take care of all exception inside it.

2.7.11.6 Parameters functions

Parameters functions return the choice for parameters values when changing an active process state. The Parameters function MUST return array of arrays with string key and string value.

Variable function MUST start with string “paramget”.

Variable functions MUST not throw exceptions, and must take care of all exception inside it.

2.7.11.7 Date functions

Date functions calculate the time. Date functions can be used either in the other functions or formulas that calculate time for timeout.

Date function MUST start with string “date”.

Date function returns the UNIX timestamp. If the returned string length is 16 characters long, the return is count of microseconds from UNIX epoch – 1.1.1970. If the returned string is 10 characters long, the return is count of seconds from UNIX epoch. The return can be also a negative number, and should be stored with at least 64 bit integer value.

2.7.11.8 Role functions

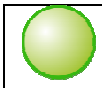
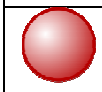
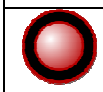
Role functions define the users who are attached to the process.

Role function MUST start with string “role”.

Role function returns the array of user identifiers.

3 Icons

3.1 Compliance 1 icons and use

	EntryPoint flag. *Process state* MUST be marked with this icon if the *Process object* can be either instancionalized with this state, or if is a Process milestone.
	EndPoint flag. *Process state* MUST be market with this icon if it does not have the *Timeout action* but have the other Process action defined.
	Terminate flag. *Process state* MUST be marked with this icon if it does not have any *Process action defined*.
	Synchronize flag. *Process state* SHOULD be marked with this icon if default flow of the process is waiting for other process to finish.
	NewProcess flag. *Process state* MUST be marked with this flag if one other process is being initialized in this state.
	NewProcesses flag. *Process state* MUST be marked with this flag if at least one other process is being initialized in this state.
	ThrowEvent flag. *Process state* MUST be marked with this flag if it creates the System events.
	Timeout flag. *Process state* MUST be marked with this icon if the default *Process action* is timeout. *Process action* MAY be marked with this icon if it is timeout.
	Function flag. *Process state* MUST be marked with this flag if it has function defined. *Process action* MUST be marked with this flag if it has function defined.
	ConditionalProcessAction flag. *Process action* MUST be marked with this icon if the condition is required for the *Process role* to execute this *Process action*.
	Parameters flag. *Process action* MUST be marked with this icon if the parameters are optional or required for the *Process action* to be executed.
	Error flag. *Process action* MUST be marked with this icon if the process flow is not expected.
	CatchEvent flag. *Process action* MUST be marked with this icon if it can catch System events. *Process state* MUST be marked with this icon if it can catch System events.

3.2 Compliance 2 icons and use

	Error1 flag. *Process state* MUST be marked with this icon if it does not have EndPoint flag on and does not have *Timeout action* defined.
	Error2 flag. *Process state* MUST be marked with this icon if it does not have *Process message* filled in for required *Process role*.

3.3 Compliance 3 icons and use

	Error3 flag. *Process state* MAY be marked for the language check.
	Error4 flag. *Process state* MAY be marked when all languages are not yet translated.