Business Process Simulation Specification

Document Number WFMC -BPSWG-2012-1

Document Status – Beta

July 13, 2016
Version 2.0

Table of Contents

1. Acknowledgement

1.1 BPSim 2.0

BPSim 2.0 was a collaborative effort coordinated by Denis Gagne.

BPSim 2.0 required many hours of work by individuals who had to find time to contribute while carrying out their normal duties for the company that employs them. We acknowledge the valuable contribution of the following individuals:

Lloyd Dugan (BPM.com), Denis Gagne (Trisotech), Geoff Hook (Lanner), Jeremy Horgan (Lanner), Benjamin Michel (W4), Simon Ringuette (Trisotech)

1.2 BPSim 1.0

BPSim 1.0 was a collaborative effort coordinated by Denis Gagne and Robert Shapiro.

BPSim 1.0 required many hours of work by individuals who had to find time to contribute while carrying out their normal duties for the company that employs them. We acknowledge the valuable contribution of the following individuals:

Andy Adler (Process Analytica), Francois Bonnet (W4), Justin Brunt (Tibco), Mike Carpenter (CACI), Peter Denno (NIST), Lloyd Dugan (DCMO), Denis Gagne (Trisotech), Torben Haag (Open Text), Hanaa Hammad (IBM), Charles Harrell (CACI), Geoff Hook (Lanner), Jeremy Horgan (Lanner), John Januszczak (SIM4BPM), Alberto Manuel (Process Sphere), Razvan Radulian (Why What How), Simon Ringuette (Trisotech), William Rivera (BizAgi), Jesus Sanchez (BizAgi), Redirley Santos (FedEx), Robert Shapiro (Process Analytica), Frances Sneddon (Simul8), Tim Stephenson (KnowProcess), Tihomir Surdilovic (Red Hat).

We also would like to acknowledge Sparx Systems for graciously providing licences to Enterprise Architect (EA) in support of this effort.

2. Introduction

As organizations face increasing levels of pressure to deliver more efficient and effective operations, business process simulation and analysis is being recognized as an integral part of optimizing performance. Inadequate or poorly designed business processes lead to customer expectations not being met and to undesired organizational behavior that may in turn lead to loss of revenues, goodwill or worse. This is why it is important to thoroughly analyse business processes in a safe isolated environment before they are deployed. The advantages of business process simulation and analysis over testing new business process in the real world include: no disturbance to current operations, the speed of validation of potential scenarios and a lower relative cost of business transformation exploration/experimentation.

Although recognized as desired and relevant within the practice of Business Process Management (BPM), simulation and analysis of business processes is still not systematically used in most business process improvement projects. The reasons for this may be many (availability, tooling, training, etc.) but one certain factor is the lack of existence of standards. While mature standards exist for the definition of business process models (e.g. BPMN and XPDL) there is no generally accepted standard for business process simulation and analysis.

In analysing business processes many different possibilities to improve the process are at hand. Structural analysis will concentrate on the structural aspects (e.g. configuration) of a business process model. These will usually consist of statistical analysis often using static methods. Capacity Analysis will on the other hand concentrate on the capacity aspects of a business process model (e.g. limitations) usually based on dynamic analysis often using discrete simulation methods.

To carry out all these analyses, business process models often need to be augmented with process analysis data. Both estimated values and historical execution values are often used as parameterization of the business process model in support of pre-execution or post-execution optimization. Pre-execution optimization will concentrate on “what if” analysis of estimated values as input parameters. While post-execution optimization will concentrate on “what if” analysis of historical values as input parameters, either provided as data for distributions or as “actuals”.

3. Scope

This document introduces the Business Process Simulation (BPSim) framework, a standardized specification that allows business process models captured in either BPMN or XPDL to be augmented with information in support of rigorous methods of analysis.

This specification defines the parameterization and interchange of process analysis data allowing structural and capacity analysis of process models. This specification is meant to support both pre-execution and post-execution optimization of said process models.

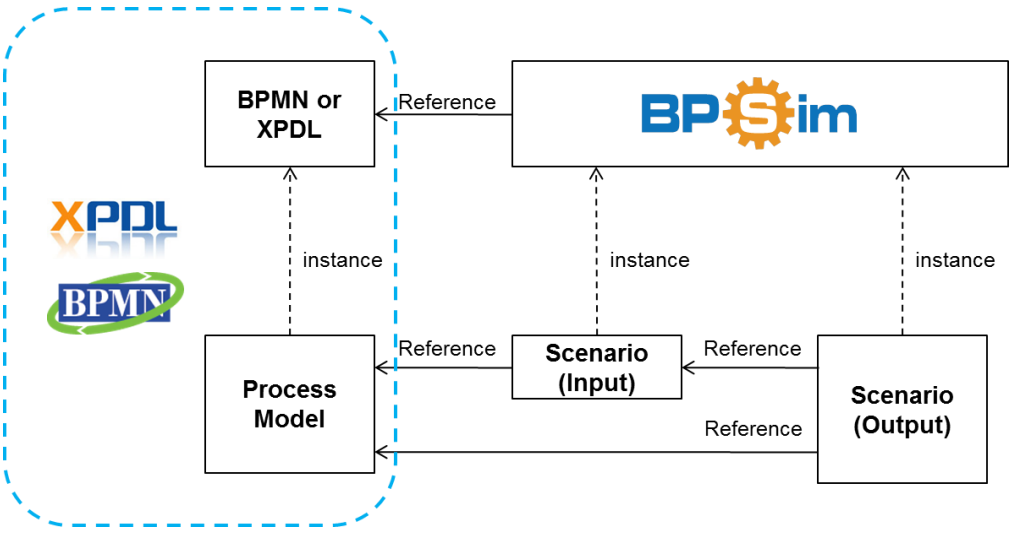
This specification consists of an underlying computer-interpretable representation (meta-model) and an accompanying electronic file format to ease the safeguard and transfer of this data between different tools (interchange format).

The BPSim meta-model is captured using the Unified Modeling Language (UML) and the interchange format is defined using an XML Schema Definition (XSD). Note that the BPSim meta-model and the interchange format represent the core of the normative material of this specification.

In defining the meta-model and the interchange format, priority was given to ensure a resulting interchange format (XML file) that is more human consumable. Conversely, this decision of favoring a more human consumable interchange format has the known side effect that the resulting meta-model does not adhere to all best practices of the object orientation.

In order to support both pre-execution and post-execution optimization, the meta-model and interchange format allow for the capture of both inputs and outputs of the process analysis. Both estimated values and historical execution values are supported as parameterization of the business process model.

One of the goals of this specification is to be complementary to already existing standards related to business process modeling. This version of the BPSim specification is scoped based on the Business Process Model and Notation (BPMN) version 2.0 from the Object Management Group (OMG) [1] and the XML Process Definition Language (XPDL) version 2.2 from the Workflow Management Coalition (WfMC) [2]. The BPSim conceptual model is presented below.



In realizing this version, attention was given as to not duplicate any process model information already provided by these two process modeling standards whenever possible. Great care was also taken to ensure that proper extension mechanism of both BPMN and XPDL were respected as to ensure that interchange could take place within a proper BPMN file, a proper XPDL file or as a standalone XML file.

4. References

[1] Object Management Group (OMG): Business Process Model and Notation (BPMN Version 2.0), OMG report: dtc/2010-06-05, OMG, (2010). http://www.bpmn.org

[2] Workflow Management Coalition (WfMC): Process Definition Interface- XML Process Definition Language (XPDL Version 2.2), WfMc Document Number WFMC-TC-1025, WfMC, (2012). http://www.xpdl.org

[3] ISO 4217 defined at http://www.iso.org/iso/catalogue_detail/?csnumber=46121

[4] ISO 8601 defined at http://www.iso.org/iso/catalogue_detail?csnumber=40874

[5] XPATH 1.0 language defined at http://www.w3.org/TR/1999/REC-xpath-19991116/

[6] iCalendar (RFC 5545) defined at http://tools.ietf.org/html/rfc5545

[7] XES 2.0 defined at http://www.xes-standard.org/_media/xes/xesstandarddefinition-2.0.pdf

5. Conformance

The meta-model and the interchange format represent the core of the normative material of this specification.

This rest of this specification is organized into sections. All sections of this document are normative.

An individual or organization (vendor or otherwise) cannot claim conformance to this specification unless addressing all normative sections of this specification.

6. Elements

6.1 Scenario

In BPSim process analysis data is used to provide complementary information to a BPMN or XPDL business process model in the context of process analysis, simulation and optimization. Business process models and their business process elements are external sources. Scenarios within the process analysis data are always in reference to a single business process model (note that many processes can be captured into a single BPMN or XPDL business process model). Thus the business process model is a separately defined fixed point with variations possible on scenarios.

Scenarios can be used to capture:
a) input parameter specification for analysis, simulation and optimization;
b) results from analysis, simulation and optimization;
c) historical data from past real world execution of the business process model.

Scenarios of results will often reference a scenario of input specification (i.e. the results for the referenced input set).

It is possible for a scenario to overload or augment (inherit from) another scenario. In such case, only the changes to the element parameters values, or the added element parameters and their values, need to be specified in the inheriting scenario.

A scenario is composed of a collection of element parameters. Each element parameter of a scenario references a specific element of a process within the business process model.

Each scenario may possess scenario parameters.

An extension capability is provided to the BPSim data. Both scenarios and element parameters can be extended with proprietary vendor extensions.

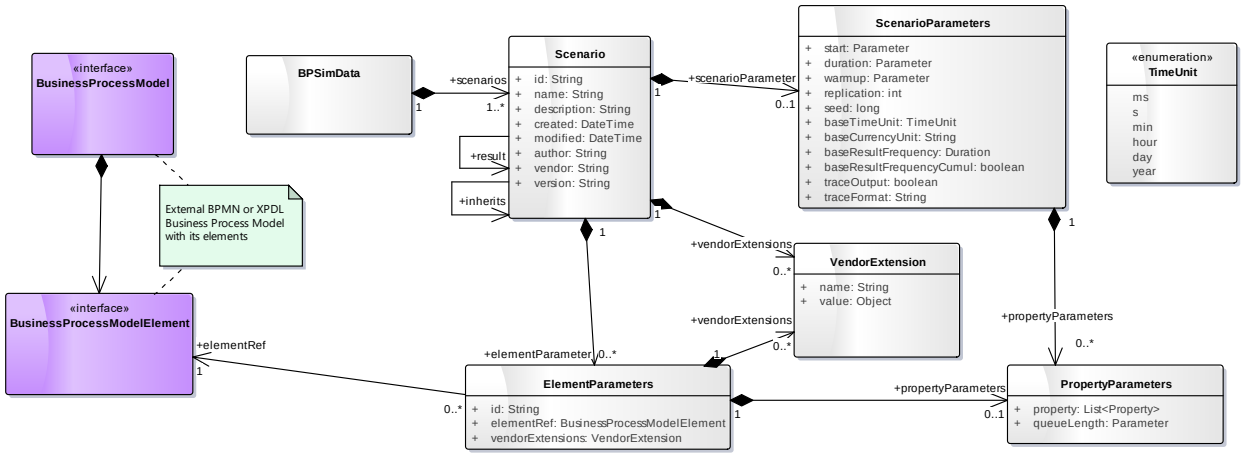


Figure 1

6.1.1 BPSimData

The BPSimData class is the root class where all scenarios are defined.

Attribute	Description
Scenario : scenarios	A collection of scenarios

6.1.2 Scenario

The Scenario class regroups all ElementParameter for a given scenario.

Attribute	Description
String : id	Unique scenario identifier
String : name	A name for this scenario
String : description	A description of this scenario
DateTime : created	When this scenario was created
DateTime : modified	When this scenario was last modified
String : author	The scenario author name
String : vendor	The name of the software tool that was used to create this scenario
String : version	The version of this scenario
Scenario : result	In the case that this scenario is the output of an analysis and that the source of the analysis is also provided in the model, this field references the source scenario.
Scenario : inherits	Reference to the scenario that this scenario inherits from. When inheriting from scenario, only overload values and added ElementParameter with values are provided.
ElementParameter : elementParameter	Collection that compose this scenario
ScenarioParameter : scenarioParameter	Parameters about this scenario
VendorExtension : vendorExtension	Proprietary vendor extensions for this Scenario

6.1.3 ScenarioParameters

The ScenarioParameter class defines the parameters about the scenario.

Attribute	Description
Parameter : start	Start time of the scenario
Parameter : duration	Duration of the scenario
Parameter : warmup	Subset of the duration period during which the statistics are not collected.
int : replication	Number of replications of that scenario that needs to be executed. Defaults to 1.
long : seed	A random seed to be used to initialize a pseudo random number generator. Given the exact same model (Business Process Model and BPSim data) and a given seed, the results should be the same across executions.

	Using replication, giving a seed does not mean that each replication will return the same result but that for a given seed and a given number of replications the exact same results are generated.
TimeUnit : baseTimeUnit	Base time unit of this scenario. All numeric and floating values representing time should be considered as being expressed in that unit unless overridden locally. The default value of this parameter is minutes (min).
String : baseCurrencyUnit	Base international currency code of this scenario expressed using the ISO 4217 [3] (three letter codes). All numeric and floating values representing a cost should be considered as being expressed in that currency code. The default value of this parameter is USD.
Duration : baseResultFrequency	Base Result Frequency of this scenario. If set, all ResultRequests of this scenario should be considered as being requested at that frequency. The default value of this parameter is undefined.
boolean : baseResultFrequencyCumul	If set to true then the ResultRequests for this scenario are cumulative based on the baseResultFrequency. If set to false then the ResultRequests will reset at each baseResultFrequency throughout this scenario. The default value of this parameter is false.
boolean : traceOutput	If set to true then a trace is generated as a separated output according to the language defined in the traceFormat parameter. The default value of this parameter is false.
String : traceFormat	Specifies the format of a simulation trace generated as a separate output. The default value of this parameter is XES [7]

6.1.4 TimeUnit

The TimeUnit enumeration represents all the possible time units.

Attribute	Description
: ms	milliseconds
: s	seconds
: min	minutes
: hour	hours
: day	days
: year	year

6.1.5 VendorExtension

The VendorExtension class is a proprietary vendor extension holder. Vendors can add non normative extensions.

Although BPSim contains most of the constructs which are likely to be required in the exchange of business process simulation, there may be circumstances under which additional information will need to be included within a process definition. Users and vendors are encouraged to work as far as possible within the standard entity / attribute sets; however, when extensions are needed BPSim provides a standard way to extend it with vendor specific extensions.

Possible extensions are structured and controlled within BPSim in order to ensure (protect) interchange capability.

When transporting a BPSim model, tools must carry all vendor extensions and not drop unknown extensions in its output.

Attribute	Description
String : name	The name of the vendor extension. Use an appropriate prefix to your extension names to prevent collision
Object : value	The value of the Vendor Extension

6.2 Parameters

Each element parameter of a scenario references a specific element of a process within the business process model.

To address separation of concerns, element parameters are divided into different perspectives. Each perspective regroups a collection of parameters from a common concern.

The values of parameters may be attached to a specific calendar to define applicability period (See section 6.4). For example a certain parameter may have a value of X on weekdays and a value of Y on weekends.

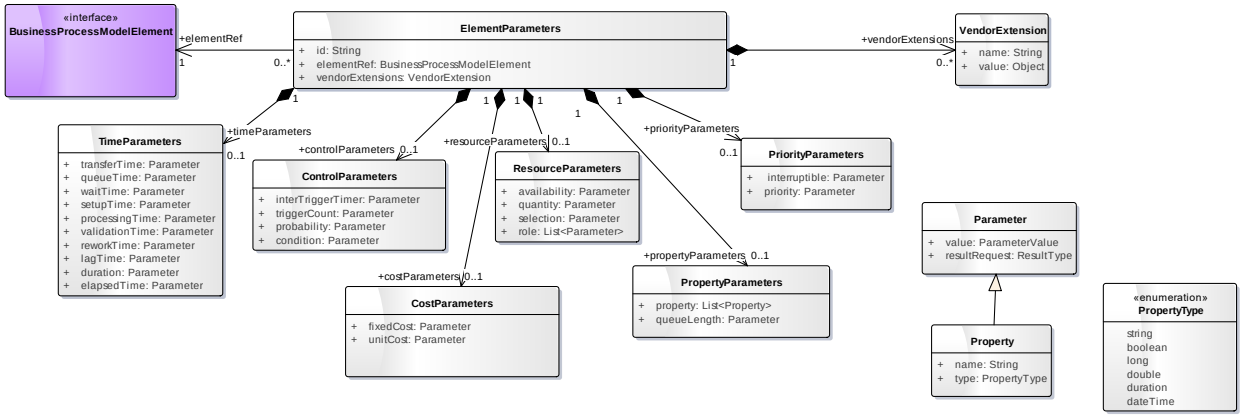


Figure 2

6.2.1 Property

Attribute	Description
String : name	The name of the property. This is a required attribute.
PropertyType : type	The type of a property. This is optional and by default the type is defined by the value type evaluated.

6.2.2 PropertyType

Attribute	Description
: string	
: boolean	
: long	
: double	
: duration	
: dateTime	

6.2.3 ElementParameters

The ElementParameter class is the concrete class definition of all parameter perspectives.

Element Parameter instances reference (through the elementRef) a business process model element.

In the case that two Element Parameters referencing the same business process element, the parameters are applied in the order they appear in the model. A later definition would overwrite an earlier definition.

Attribute	Description
String : id	A unique identifier for this parameter
BusinessProcessModelElement : elementRef	A reference to the business process element identifier for which we are defining a parameter
VendorExtension : vendorExtensions	Proprietary vendor extensions for this ElementParameter

6.2.4 TimeParameters

The TimeParameter class groups all parameters that specify time related parameters for a business process element.

All TimeParameters capture time intervals and are defined from an external observer point of view.

The time it takes from the completion of a predecessor process model element to the completion of the current process model element is known as the elapsed time.

Elapsed time is broken into two main time interval of interest that are duration and lagTime. The duration time interval captures the time from the start of a work effort to its completion, while the lagTime interval captures the time from the completion of a predecessor process model element until the start of the successor process model element.

In order to support temporal analysis experiments with lower levels of granularity as often desirable in Lean and Six Sigma projects, BPSim specifies elapsed time, duration and lagTime as elements of the meta-model and interchange format only for ResultRequests in order to prevent conflicting parameterization of the summation value with respects to its composing elements.

Elapsed time is in effect lagTime + duration.

Duration is in effect the sum of setupTime, processingTime, validationTime and reworkTime.
duration=setupTime+processingTime+validationTime+reworkTime.

In situations where this granularity is not desired, processingTime should be used to specify the temporal interval.

Lag time is in effect the sum of transferTime, queueTime and waitTime.
lagTime=transferTime+queueTime+waitTime.

In situations where this granularity is not desired, waitTime should be used to specify the temporal interval.

TimeParameters values must resolve to either: a NumericParameter, a FloatingParameter or a DurationParameter.

The default value of all unspecified TimeParameters is considered to be 0, except for elapsedTime, duration and lagTime which can only be used for ResutRequests and cannot be set.

Parameters from the temporal perspective only apply to types of business process elements that take place over an interval of time (see section 7.1).

Attribute	Description
Parameter : transferTime	The time spent traveling from the previous processing step
Parameter : queueTime	The delay between the Successor being offered and the successor being allocated
Parameter : waitTime	The time between the Successor being allocated and it being actually started.
Parameter : setupTime	The time expended prior to performing the actual work.
Parameter : processingTime	The time actually spent doing the work at hand.

Parameter : validationTime	The time spent reviewing or inspecting the work done.
Parameter : reworkTime	The time spent correcting or redoing the work done
Parameter : lagTime	The time from the completion of a predecessor process model element to the start of the successor process model element. lagTime parameter can only be used for ResultRequests and cannot be set with a value.
Parameter : duration	The time from the start of a work effort to its completion. duration parameter can only be used for ResultRequests and cannot be set with a value.
Parameter : elapsedTime	The time it takes from the completion of a predecessor process model element to the completion of the current process model element.

6.2.5 ControlParameters

The ControlParameter class groups all parameters that specify the control flow of a business process element.

Parameters from the control perspective only apply to certain types of business process elements (see section 7.2).

Attribute	Description
Parameter : interTriggerTimer	The time interval between occurrences. After the specified time interval, the event occurs and will keep occurring every time interval until the triggerCount is reached. Modeler should be careful when putting an inter trigger time to a boundary event that can be triggered otherwise (e.g. signal or timer). This parameter must resolve to either: a NumericParameter, a FloatingParameter or a DurationParameter. The default value of this parameter is that the event never occurs. Setting a duration of 0 would mean that the event occurs instantly.
Parameter : triggerCount	The maximum number of times to trigger this event. This parameter must resolve to a NumericParameter.

	The default value of this parameter is considered to be infinity so not defining it means that the event will not stop occurring after a maximum number of times.
Parameter : probability	The probability of the control being passed to this element. This parameter must resolve to either a NumericParameter or a FloatingParameter. The default value of this parameter varies depending on the element being referenced. For sequence flow, the default probability is distributed evenly between any outgoing sequence flows that do not have a probability defined. e.g. : 4 outgoing sequence flows, none with defined probability, they each have a probability of 0.25 e.g. : 3 outgoing sequence flows, one with a probability of 0.4 defined, the other two that don't have a probability defined will receive the probability of 0.3. For events, the default probability is 0. Only one of probability or condition can be defined for a given business process model element.
Parameter : condition	A condition that must be met before passing the control to this element. This parameter must resolve to a BooleanParameter. The default value of this parameter is false. Only one of probability or condition can be defined for a given business process model element.

6.2.6 ResourceParameters

The ResourceParameter class groups all parameters that specify the resources of a business process element.

Parameters from the resources perspective only apply to certain types of business process elements.

Attribute	Description
Parameter : availability	Determine whether a resource is available or not. This will often be varied according to a calendar to represent when a resource is available. This parameter must resolve to a BooleanParameter. The default value of this parameter is true (the resource is available)
Parameter : quantity	The quantity of resources. This parameter must resolve to a NumericParameter. The default value of this parameter is 1.
Parameter : selection	Criteria for selecting the desired resource. This is an override of the BPMN ResourceRole element behavior to assign roles to resources. This parameter can reference roles defined using the Role ResourceParameters to easily reference the resources associated with the activity. This parameter must resolve to either a StringParameter or NumericParameter. The default value of this parameter is to conserve the BPMN ResourceRole behavior and not override it.
List<Parameter> : role	The roles (may be more than one) of a resource. These roles can be reused in the Selection ResourceParameter. Each role must resolve to a StringParameter. The default value of this parameter is to not define roles for the resource.

6.2.7 CostParameters

The CostParameter class groups all parameters that specify the cost of a business process element

Parameters from the cost perspective only apply to certain types of business process elements (see section 7.4).

Attribute	Description
Parameter : fixedCost	The fixed cost that has to be paid after each occurence. The cost is expressed as the number of baseCurrencyUnit defined in the Scenario Parameters. This parameter must resolve to either a NumericParameter or a FloatingParameter. The default value of this parameter is 0.
Parameter : unitCost	The cost per unit of time that has to be paid. The cost is expressed as the number of baseCurrencyUnit per baseTimeUnit defined in the Scenario Parameters. This parameter must resolve to either a NumericParameter or a FloatingParameter. The default value of this parameter is 0.

6.2.8 PropertyParameters

The PropertyParameter class groups all parameters that specify the property parameters of a business process element.

Parameters from the property perspective only apply to certain types of business process elements (see section 7.5).

Attribute	Description
List<Property> : property	Specifes additional properties that are assigned to BPMN Elements. Those properties can be accessed in ExpressionParameters. Property parameters are evaluated and set as soon as a token enters the element and before everything else.

	This parameter can resolve to any type of parameter and does not have a default value.
Parameter : queueLength	This parameter can only be used to request results against a business process model element. The queue length represents the number of tokens waiting for available resources.

6.2.9 PriorityParameters

The PriorityParameters class groups all parameters that specify the priority parameters of a business process element.

Parameters from the priority perspective only apply to certain types of business process elements (see section 7.6).

Attribute	Description
Parameter : interruptible	Determines whether the execution of this element is interruptible. This parameter must resolve to a BooleanParameter. The default value of this parameter is false (the element is uninterruptible).
Parameter : priority	Determines the priority of a business process element and is used to influence the order in which scarce resources are allocated. The higher the value, the higher the priority. This parameter must resolve to either a FloatingParameter or a NumericParameter. The default value of this parameter is 0.

6.3 Parameter Type

A parameter must have a default value. The default value of a parameter can be modified for various intervals of time by enumerating additional values for the parameter. Each additional value must specify a calendar of applicability.

Each value provided for a parameter must be of a specific type.

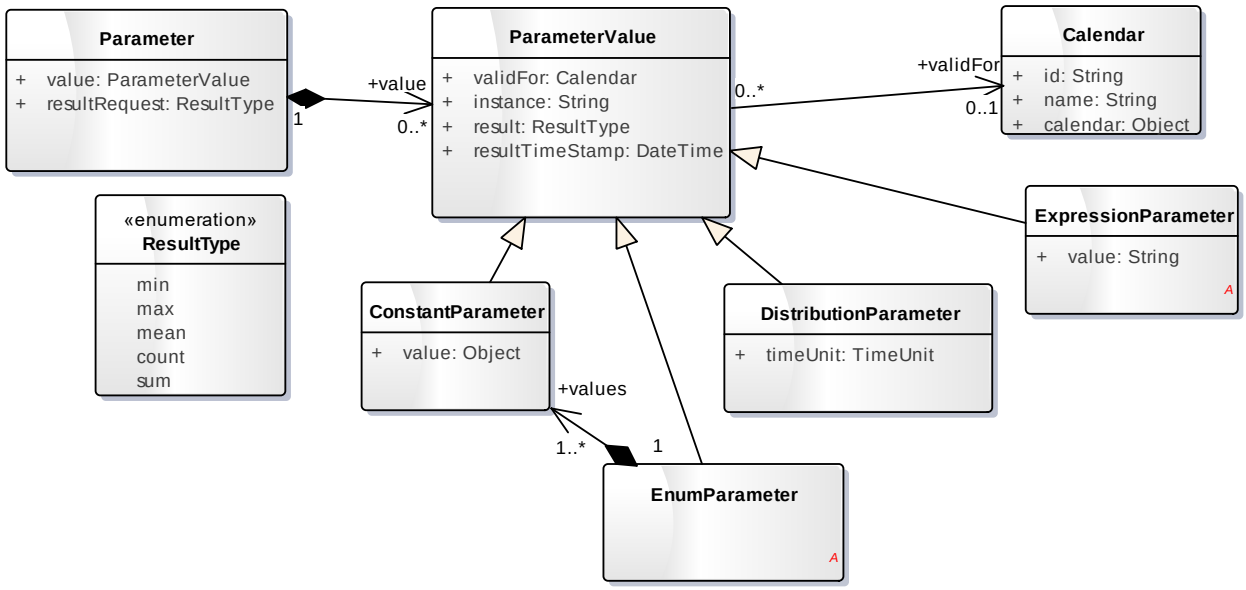


Figure 3

6.3.1 Parameter

The Parameter class groups the ParameterValues for a parameter.

Attribute	Description
ParameterValue : value	The value of the parameter.
ResultType : resultRequest	Used for input only, this indicates the result types that we expect to see in the result scenario for this parameter.

6.3.2 ParameterValue

The ParameterValue class is the abstract class definition of all ParameterValue types.

Attribute	Description
Calendar : validFor	References a calendar of applicability for this value. If unspecified, the ParameterValue is the default value of this parameter.

	Any parameter value changed by a calendar comes into effect the next time the parameter is evaluated. In the special case of an interTriggerTimer assigned to a process element that starts a process, the new value is applied immediately when the value is changed by the calendar (as opposed to the next time a token is generated).
String : instance	The unique identifier of the process instance that this value was obtained from. Used for representing an output value only.
ResultType : result	Used for result values only, this indicates the type of result that this parameter represents.
DateTime : resultTimeStamp	Used for result values only, this provides the simulation time at which the result was produced.

6.3.3 ResultType

The ResultType enumeration represents all the possible output that can be generated from the analysis.

Attribute	Description
: min	Output the minimum value as a result
: max	Output the max value as a result
: mean	Output the mean value as a result
: count	Output the number of occurrence as a result
: sum	Output the sum of all results

6.3.4 Constant Parameters

Constant parameters are parameters that will always resolve to the same value over time.

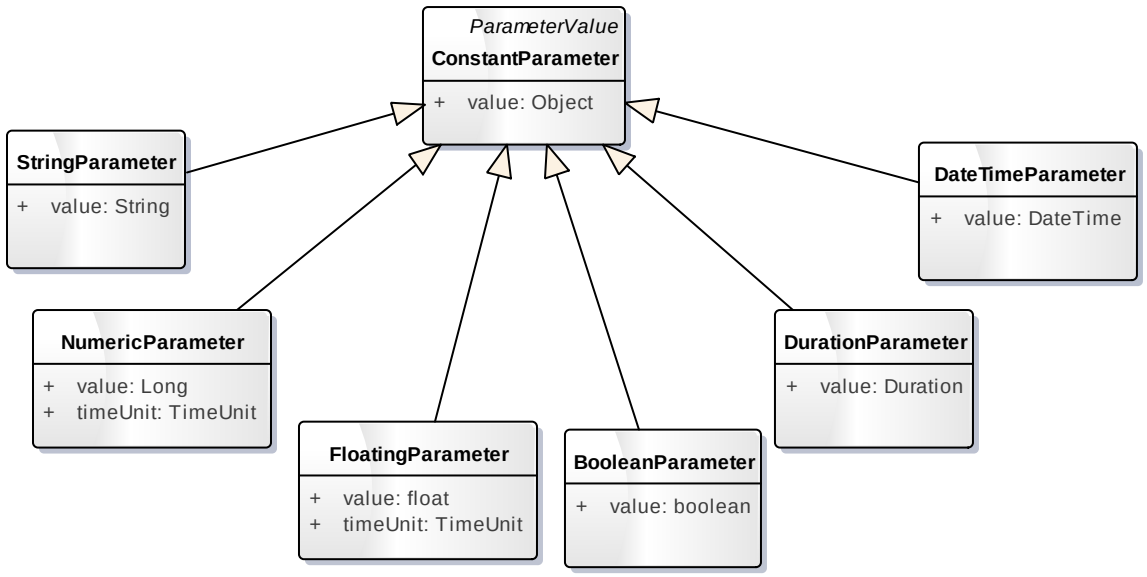


Figure 4

6.3.4.1 BooleanParameter

Attribute	Description
boolean : value	true or false.

6.3.4.2 ConstantParameter

Attribute	Description
Object : value	The constant value of the appropriate type.

6.3.4.3 DateTimeParameter

An instant in time defined using the ISO 8601 [4] format for dateTime that is expressed on Zulu time

Attribute	Description
DateTime : value	Using the YYYY-MM-DDThh:mm:ssZ format. All DateTime are assumed to be in Zulu time.

6.3.4.4 DurationParameter

A duration defined using the ISO 8601 [4] format for duration

Attribute	Description
Duration : value	Duration is specified using either the long format (PnnYnnMnnDTnnHnnMnnS) or the short format (e.g. PnnW).

6.3.4.5 FloatingParameter

Attribute	Description
float : value	a floating point value.
TimeUnit : timeUnit	Override the baseTimeUnit defined in the ScenarioParameters for this value.

6.3.4.6 NumericParameter

Attribute	Description
Long : value	Integer value.
TimeUnit : timeUnit	Override the baseTimeUnit defined in the ScenarioParameters for this value.

6.3.4.7 StringParameter

Attribute	Description
String : value	String value.

6.3.5 Distribution Parameters

Distribution parameters are parameters that have different values over time but statistically distributed according to a given distribution.

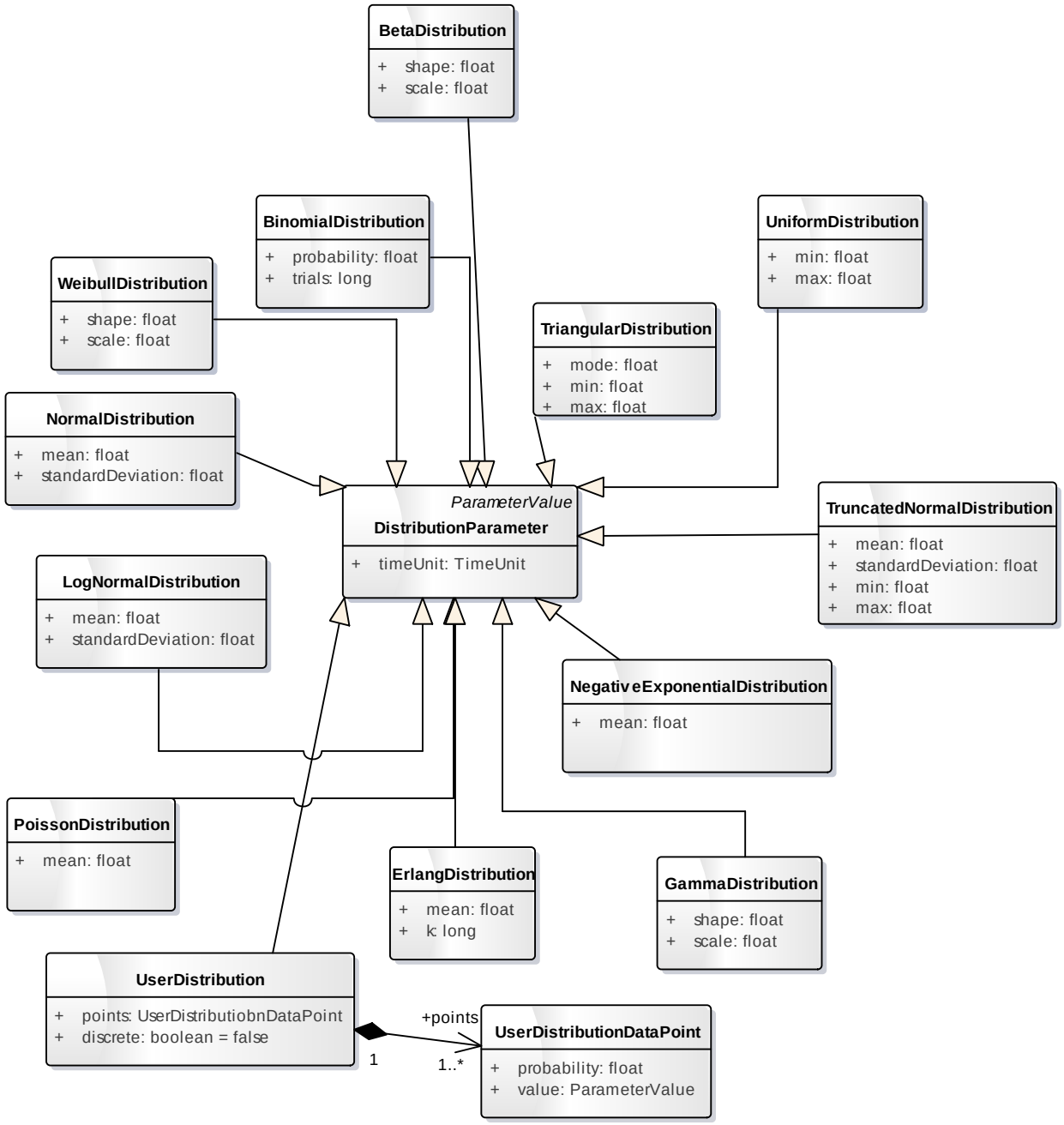


Figure 5

6.3.5.1 DistributionParameter

The various supported distributions each have a number of parameters to configure them. In the XML serialization, those attributes are not mandatory in the schema to support the transport of a work in progress model. However, to be consumed by a simulator, all parameters of a distribution should be provided.

Attribute	Description
TimeUnit : timeUn	Override the baseTimeUnit defined in the ScenarioParameters for this value.

6.3.5.2 BetaDistribution

The BetaDistribution class provides a sample from the beta distribution, which is a real distribution. The beta distribution can assume a wide variety of shapes and is often used as a rough model where real-life data is limited

Attribute	Description
float : shape	The shape value.
float : scale	The scale value.

6.3.5.3 BinomialDistribution

The BinomialDistribution class provides a sample from the binomial distribution, which is an integer distribution. It returns the expected number of successes, given a number of trials and a probability of success. For example, if light bulbs from a supplier are known to be 10% faulty, you could use the binomial distribution to estimate the number of faulty bulbs in a batch of five.

Attribute	Description
float : probability	The probability of success.
long : trials	The number of trials.

6.3.5.4 ErlangDistribution

The ErlangDistribution class provides a sample from an ERLANG K distribution, which is a real distribution. The Erlang is a family of distributions: it has a different curve depending on the value of the K parameter.

When K = 1, the Erlang distribution is identical to the **negative exponential** distribution (this is because it is based on the sum of K samples from a negative exponential distribution with the same mean).

When K = 2, the Erlang distribution is a bell-shaped distribution, strongly skewed to the left (similar in shape to the **Log Normal** distribution).

When K is larger than 2, the Erlang distribution starts to resemble the **normal** distribution.

However, unlike the **Normal** or **Log Normal** distributions, the **Erlang** distribution is characterized by its mean alone.

You can use the Erlang distribution for sensitivity analysis by changing the K parameter (for example, for testing the effect of stoppages). Low K values cause maximum chaos, while high K values reduce chaos.

Attribute	Description
float : mean	The mean value.
long : k	The K Value.

6.3.5.5 GammaDistribution

The GammaDistribution class provides a sample from the gamma distribution, which is a real distribution. It returns a sample from the distribution with a specified shape and scale.

Attribute	Description
float : shape	The shape value.
float : scale	The scale value.

6.3.5.6 LogNormalDistribution

The LogNormalDistribution class provides a sample from a Log Normal distribution, which is a real distribution. It is a bell-shaped distribution, strongly skewed to the right.

Data is said to come from a log normal distribution if the logarithms of the sample values follow a normal distribution.

Attribute	Description
float : mean	The mean value.
float : standardDeviation	The standard deviation value.

6.3.5.7 NegativeExponentialDistribution

The NegativeExponentialDistribution class provides a sample from the Negative Exponential distribution, which is a real distribution. It may be thought of as the complement of the Poisson distribution.

Attribute	Description
float : mean	The mean value.

6.3.5.8 NormalDistribution

The NormalDistribution class provides a sample from the Normal distribution, which is a real distribution. This is one of the most common distributions in nature, and has a symmetrical bell-shaped curve. It is useful for modeling situations where values are evenly distributed around a mean.

Attribute	Description
float : mean	The mean value.
float : standardDeviation	The standard deviation.

6.3.5.9 PoissonDistribution

The PoissonDistribution class provides a sample from the Poisson distribution, which is an integer distribution. Typically, it is used to estimate the number of arrivals within a given period (for example, size of batches for tokens). It may be thought of as the complement of the negative exponential distribution.

Attribute	Description
float : mean	The mean value.

6.3.5.10 TriangularDistribution

This provides a sample from the Triangular distribution which is a real value. As its name suggests, this distribution has a triangular 'curve'.

Attribute	Description
float : mode	The most likely value.
float : min	The lower bound of the generated numbers.
float : max	The upper bound of the generated numbers.

6.3.5.11 TruncatedNormalDistribution

The TruncatedNormalDistribution class provides a sample from the Truncated Normal distribution, which is a real distribution. This is similar to the normal distribution with the difference being that minimum and maximum values for sampling are specified.

Attribute	Description
float : mean	The mean value.
float : standardDeviation	The standard deviation value.
float : min	The lower bound of the generated numbers.
float : max	The upper bound of the generated numbers.

6.3.5.12 UniformDistribution

The UniformDistribution class provides a sample from the Uniform distribution, which is a real distribution. It may be used when there is equal probability of obtaining any real value in the specified range.

Attribute	Description
float : min	The lower bound of the generated numbers.
float : max	The upper bound of the generated numbers.

6.3.5.13 UserDistribution

The UserDistribution class provides a custom sampling of points with the likeliness of each one to occur. The discrete parameter (false) determines if a sample is extrapolated between data points or alternatively only actual data points (true) are returned.

Attribute	Description
UserDistributiobnDataPoint : points	A list of data points.
boolean : discrete	If set to true then the user distribution is discrete, if set to false than the distribution is continuous. The default value is set to false.

6.3.5.14 UserDistributionDataPoint

The UserDistributionDataPoint class represents a data point in the User Distribution

Attribute	Description
float : probability	The probability of this data point occurring expressed as a fraction from 0 to 1. The sum of all data point probabilities should add to 1.0.
ParameterValue : value	The value of the Data Point.

6.3.5.15 WeibullDistribution

The WeibullDistribution class provides a real sample from the Weibull Distribution. It returns a sample from the distribution with a specified shape and scale.

Attribute	Description
float : shape	The shape value.
float : scale	The scale value.

6.3.6 Enumeration parameters

Enumeration parameters are collections of constant parameters. Enumeration parameters provide a collection of data points resulting from analysis, simulation and optimization or from real world execution of the business process model (historical data).

Every time the parameter is evaluated, the next value in the collection is returned.

6.3.6.1 EnumParameter

The use of historical data can be supported by the specification in two ways, either by supplying the actual numbers as parameters, i.e. a sequence of processing times for a task. A more common way is to use historical data for an appropriate period of time to be used to generate a distribution. Curve fitting software can be used to suggest the appropriate distribution or alternatively a ‘user distribution’ constructed from the data depending on which approach is most valid for the circumstances.

Attribute	Description
List<ConstantParameter> : values	A collection of values for this enumeration.

6.3.7 Expression parameters

Expression parameters are parameters that are a combination of explicit values, operators and functions. Values are computed at runtime providing a result determined by the expression.

6.3.7.1 ExpressionParameter

Attribute	Description
String : value	The XPATH expression.

The expression has to be expressed using the XPATH 1.0 language [5]. For the purpose of the BPSim framework, XPATH is extended to provide these additional functions under the "bpsim" namespace:

XPath Extension Function	Description / Usage
bpsim:getProperty(*name*)	Returns the value of a property parameter. **Arguments** *name*: the name of the property parameter. **Return** Returns the value. **Remarks** If the property parameter does not exist, default to zero.
bpsim:getResource(*name*, *qty*)	Selects a collection of available resource(s) required for an Activity. **Arguments** *name*: the name of the resource required by the Activity. In the case of BPMN this is the attribute used to uniquely identify BPMN resource element. *qty*: the quantity of the resource required by the Activity, expressed as an integer. **Return** Collection of resource(s) or an empty collection if the

	resource requirements were not satisfied. **Remarks** Resources are defined in the BPMN interchange using the <resource> element.
bpsim:getResourceByRoles(*[role*, …], *qty*)	Selects a collection of available resource(s) that can satisfy the role(s) required for an Activity. Selected resource(s) will play all roles specified by the list of roles required. **Arguments** *[role …]:* the variable list of required role(s). *qty:* the quantity of a resource that satisfies the specified role(s), required by the Activity, expressed as an integer. **Return** Collection of resource(s) or an empty collection if the resource requirements were not satisfied **Remarks** A role can be applied to a resource using the BPSim *role* parameter from the *ResourceParameters* perspective.
bpsim:orResource(*[resources*, …])	Select the first collection of available resource(s) from the list of alternative resource(s) used for an Activity. **Arguments** A variable list of resources returned by the getResource() or getResourceByRoles() functions. **Return** Collection of resource(s). **Remarks** This allows alternative behaviour for resource selection. The evaluation order of resources is from left to right.

6.4 Calendar

Calendars are defined at the scenario level and Parameter Values references them.

A calendar is serialized using the iCalendar (RFC 5545) [6] format and it provides the time interval for which a parameter value should be used.

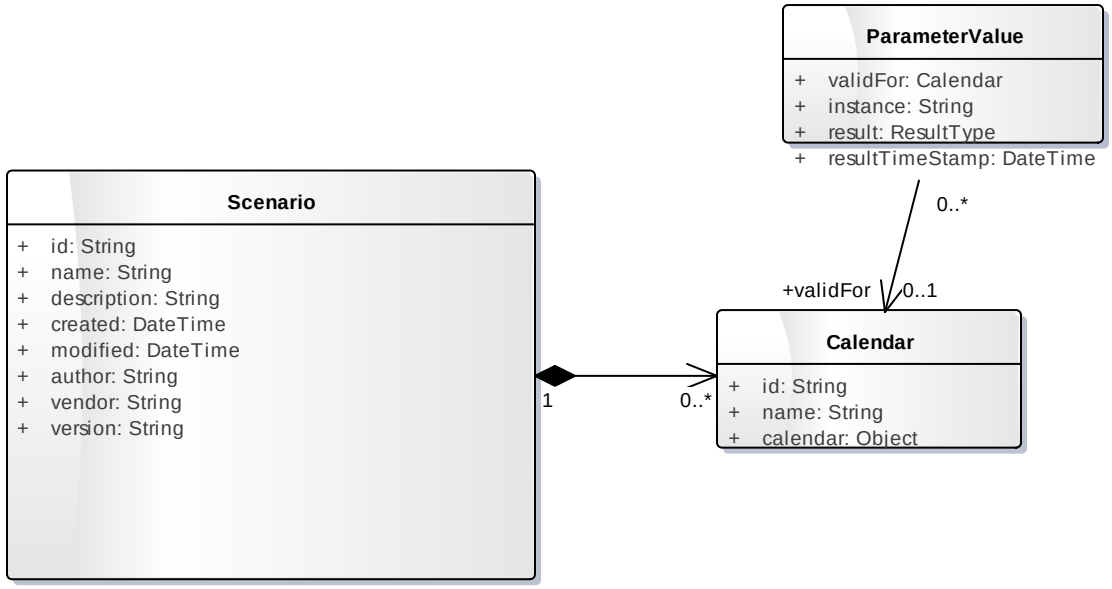


Figure 6

6.4.1 Calendar

The calendar class serializes the iCalendar format.

Attribute	Description
String : id	Calendar unique identifier.
String : name	Descriptive name for this calendar.
Object : calendar	iCalendar serialization of events (VEVENT) describing this calendar. In the XML serialization, the calendar is serialized in the text of the Calendar element.

7. BPSim Parameters Applicability

This section provides an overview of which BPSim parameters can be assigned to the various process model elements. Note that we only refer herein to the BPMN 2.0 [1] notational elements as both BPMN 2.0 [1] and XPDL 2.2 [2] uses the same notation for their process model definition.

Given the volume and complexity of these constraints, they are presented in a table format to improve readability. BPSim parameters are listed at the top of the table while BPMN notational elements are listed to the left of the table. Applicability constraints are provided within the cells. Each table is followed by a brief clarification text when necessary.

7.1 Time Parameters

		Time Parameters						
		transferTime	queueTime	waitTime	setupTime	processingTime	validationTime	reworkTime
Events	Start Event	No - BPMN Events map to time point and thus cannot have Time Parameters which are time intervals						
	Intermediate Event							
	End Event							
Activities	Task	**Yes**						
	Sub Process	**Yes - but only for activities without decomposition**						
	Transaction							
	Call Activity							
	Event Sub Process							
Gateways	Gateway	No - BPMN Gateways do not map to time intervals as they are only visualizations of branching logic						
Connecting Objects	Sequence Flow	No - BPMN Connecting Objects do not have time interval associated to them						
	Message Flow							
	Data Association							
	Association							
Data	Data Object	No - BPMN Data Elements have no impact on the Simulation or the Analysis of the Process						
	Data Store							
Swimlanes	Lane	No - BPMN Swimlanes have no impact on the Simulation or the Analysis of the Process						
	Pool							
Artifacts	Artifact	No - BPMN Artifacts have no impact on the Simulation or the Analysis of the Process						
Attributes	ResourceRole	No						
	Resource							

Generally speaking, Time parameters are applicable only to BPMN Activities that do not have decomposition in the business process model. If an activity has decomposition in the BPMN model, the Time parameters of the decomposition should be used and the abstraction can't have Time parameters defined. By “BPMN Activities that do not have decomposition” we mean abstractions that do not have its details defined in the business process model (e.g. BPMN collapsed sub process where the sub process elements are not defined in the process model).

7.2 Control Parameters

<table>
<tr><th colspan="2" rowspan="2"></th><th colspan="4">Control Parameters</th></tr>
<tr><th>interTriggerTimer</th><th>triggerCount</th><th>probability</th><th>condition</th></tr>
<tr><td rowspan="3">Events</td><td>Start Event</td><td>Yes</td><td>Yes</td><td colspan="2">Yes - but inside Event Sub Process only</td></tr>
<tr><td>Intermediate Event</td><td>Yes - but for Catch Event only</td><td>No</td><td colspan="2">Yes - but for Boundary Event only</td></tr>
<tr><td>End Event</td><td colspan="4">No</td></tr>
<tr><td rowspan="5">Activities</td><td>Task</td><td colspan="2" rowspan="4">Yes - but only for activities without decomposition without incoming sequence flow</td><td colspan="2" rowspan="4">No</td></tr>
<tr><td>Sub Process</td></tr>
<tr><td>Transaction</td></tr>
<tr><td>Call Activity</td></tr>
<tr><td>Event Sub Process</td><td colspan="4">Yes - but only for Event Sub Process without decomposition</td></tr>
<tr><td>Gateways</td><td>Gateway</td><td colspan="2">Yes - but only for Event Based Gateway starting a process</td><td colspan="2">No</td></tr>
<tr><td rowspan="4">Connecting Objects</td><td>Sequence Flow</td><td colspan="2" rowspan="3">No</td><td colspan="2">Yes</td></tr>
<tr><td>Message Flow</td><td colspan="2" rowspan="3">No</td></tr>
<tr><td>Data Association</td></tr>
<tr><td>Association</td></tr>
<tr><td rowspan="2">Data</td><td>Data Object</td><td colspan="4" rowspan="2">No - BPMN Data Elements have no impact on the Simulation or the Analysis of the Process</td></tr>
<tr><td>Data Store</td></tr>
<tr><td rowspan="2">Swimlanes</td><td>Lane</td><td colspan="4" rowspan="2">No - BPMN Swimlanes have no impact on the Simulation or the Analysis of the Process</td></tr>
<tr><td>Pool</td></tr>
<tr><td>Artifacts</td><td>Artifact</td><td colspan="4">No - BPMN Artifacts have no impact on the Simulation or the Analysis of the Process</td></tr>
<tr><td rowspan="2">Attributes</td><td>ResourceRole</td><td colspan="4" rowspan="2">No</td></tr>
<tr><td>Resource</td></tr>
</table>

The InterTriggerTimer parameter is applicable to all BPMN types of Start Events, Intermediate Catching Events and Event Sub Processes without decomposition. The InterTriggerTimer parameter can also be applied to BPMN Activities that are initiating the process (i.e.BPMN elements that do not have an incoming sequence flow:Task, Sub Process, Transaction and Call Activity).

The TriggerCount parameter can be applied to all BPMN types of Start Events, Event Sub Process without decomposition and Activities that are initiating the process (i.e.BPMN elements that do not have an incoming sequence flow (Task, Sub Process, Transaction and Call Activity).

InterTriggerTimer and TriggerCount parameters can be applied to BPMN Boundary Intermediate Events. They can also be applied on BPMN Event Sub Process without decomposition or on BPMN Start Event inside the decomposition of the Event Sub Process. Probability and Condition parameters can also be applied on an BPMN Event Based Gateway that starts a process.

7.3 Resource Parameters

		Resource Parameters			
		availability	quantity	role	selection
Events	Start Event	No			
	Intermediate Event				
	End Event				
Activities	Task	No			
	Sub Process				
	Transaction				
	Call Activity				
	Event Sub Process				
Gateways	Gateway	No			
Connecting Objects	Sequence Flow	No			
	Message Flow				
	Data Association				
	Association				
Data	Data Object	No - BPMN Data Elements have no impact on the Simulation or the Analysis of the Process			
	Data Store				
Swimlanes	Lane	No - BPMN Swimlanes have no impact on the Simulation or the Analysis of the Process			
	Pool				
Artifacts	Artifact	No - BPMN Artifacts have no impact on the Simulation or the Analysis of the Process			
Attributes	ResourceRole	No			**Yes**
	Resource	**Yes**			No

Availability, Quantity and Role parameters can be applied to BPMN Resource elements, while the Selection parameter can be applied to BPMN ResourceRole elements.

7.4 Cost Parameters

		Cost Parameters	
		fixedCost	unitCost
Events	Start Event	No	
	Intermediate Event		
	End Event		
Activities	Task	**Yes**	
	Sub Process		
	Transaction		
	Call Activity		
	Event Sub Process		
Gateways	Gateway	No	
Connecting Objects	Sequence Flow	No	
	Message Flow		
	Data Association		
	Association		
Data	Data Object	No - BPMN Data Elements have no impact on the Simulation or the Analysis of the Process	
	Data Store		
Swimlanes	Lane	No - BPMN Swimlanes have no impact on the Simulation or the Analysis of the Process	
	Pool		
Artifacts	Artifact	No - BPMN Artifacts have no impact on the Simulation or the Analysis of the Process	
Attributes	ResourceRole	No	
	Resource	**Yes**	

The FixedCost and UnitCost parameters can be applied to BPMN Activities and to BPMN Resource elements.

7.5 Property Parameters

<table>
<tr><th colspan="2" rowspan="2"></th><th colspan="2">Property Parameters</th></tr>
<tr><th>property</th><th>queueLength</th></tr>
<tr><td rowspan="3">Events</td><td>Start Event</td><td rowspan="3">Yes</td><td rowspan="3">No</td></tr>
<tr><td>Intermediate Event</td></tr>
<tr><td>End Event</td></tr>
<tr><td rowspan="5">Activities</td><td>Task</td><td rowspan="5">Yes</td><td>Yes</td></tr>
<tr><td>Sub Process</td><td rowspan="4">Yes – But only for activities without decompositions</td></tr>
<tr><td>Transaction</td></tr>
<tr><td>Call Activity</td></tr>
<tr><td>Event Sub Process</td></tr>
<tr><td>Gateways</td><td>Gateway</td><td colspan="2">No</td></tr>
<tr><td rowspan="3">Connecting Objects</td><td>Sequence Flow</td><td rowspan="2">Yes</td><td rowspan="2">No</td></tr>
<tr><td>Message Flow</td></tr>
<tr><td>Data Association</td><td colspan="2" rowspan="2">No</td></tr>
<tr><td>Association</td></tr>
<tr><td rowspan="2">Data</td><td>Data Object</td><td colspan="2" rowspan="2">No - BPMN Data Elements have no impact on the Simulation or the Analysis of the Process</td></tr>
<tr><td>Data Store</td></tr>
<tr><td rowspan="2">Swimlanes</td><td>Lane</td><td colspan="2" rowspan="2">No - BPMN Swimlanes have no impact on the Simulation or the Analysis of the Process</td></tr>
<tr><td>Pool</td></tr>
<tr><td>Artifacts</td><td>Artifact</td><td colspan="2">No - BPMN Artifacts have no impact on the Simulation or the Analysis of the Process</td></tr>
<tr><td rowspan="2">Attributes</td><td>ResourceRole</td><td colspan="2" rowspan="2">No</td></tr>
<tr><td>Resource</td></tr>
</table>

Property parameters can be set and evaluated for BPMN Events, Activities, Sequence Flows and Message Flows.

Queue Length can only be used for result requests.

7.6 Priority Parameters

		Priority Parameters	
		interruptible	priority
Events	Start Event	No	
	Intermediate Event		
	End Event		
Activities	Task	**Yes**	
	Sub Process	**Yes - but only for activities without decomposition**	
	Transaction		
Gateways	Gateway	No	
Connecting Objects	Sequence Flow	No	
	Message Flow		
	Data Association		
	Association		
Data	Data Object	No - BPMN Data Elements have no impact on the Simulation or the Analysis of the Process	
	Data Store		
Swimlanes	Lane	No - BPMN Swimlanes have no impact on the Simulation or the Analysis of the Process	
	Pool		
Artifacts	Artifact	No - BPMN Artifacts have no impact on the Simulation or the Analysis of the Process	
Attributes	ResourceRole	No	
	Resource		

Priority parameters can only be applied to BPMN Activities without decomposition.

8. Result Request Applicability

Using BPSim input scenarios, it is possible to specify requests for results against a specific BPSim parameter of a specific process model element (e.g. the resulting processing time of a specific process task). To do so, the BPSim Parameter applied to a specific process model element will have a result request attribute set to a desired result type (MIN, MAX, MEAN, COUNT, SUM).

A BPSim input scenario containing parameter result requests will generate a corresponding output scenario with a parameter value for each of the requests made in the input scenario.

This section presents both the constraints on the kind of result requests that can be applied on BPSIm parameters for a given process model element and the expected semantic of the returned results in the corresponding output scenario. Applicability constraints are presented in the tables while semantic is described in the brief text under the table.

Given the volume and complexity of these constraints, they are presented in a table format in order to simplify readability. BPSim parameters are listed at the top of the table while result request types are listed to the left of the table. Applicability constraints are provided within the cells of the table. Each table is followed by a brief clarification text and interpretation of the returned values within an output scenario.

Note that in this section, we only refer to the BPMN 2.0 [1] notational elements as a substitute to the actual process model element as both BPMN 2.0 and XPDL 2.2 uses the same notation for their process model definition.

8.1 Time Parameters

	Time Parameters						
	transferTime	queueTime	waitTime	setupTime	processingTime	validationTime	reworkTime
MIN	**Yes**						
MAX							
MEAN							
COUNT							
SUM							

When used on a BPMN Activity, the result request of type MIN returns the minimum value of that specific time parameter (i.e. transferTime, queueTime, waitTime, setupTime, processing Time, validationTime, reworkTime, lagTime, duration, elapsedTime), the result request of type MAX returns the maximum value of that specific time parameter and the result request of type MEAN returns the mean of that specific time parameter.

The result request of type COUNT returns the number of times where that specific time parameter was completed and the result request of type SUM returns the total amount of time for that time specific parameter.

Result requests can be applied to Time parameters on the same process model elements that accept them as input (including lagTime, duration and elapsedTime). In the specific context of result requests it is also possible to add Time Parameters to the following BPMN elements:

- Resource
- ResourceRole
- Activities with decomposition
- Process

Note that the fact that you can add time parameters to these elements does not infer that you can parameterize them as input.

When used on a BPMN Process element, the result request behaves exactly like on a BPMN Activity and returns the result abstracted at the Process level.

When used on a BPMN Resource or ResourceRole element, only the SUM of the waitTime, setupTime, processingTime, validationTime, reworkTime and duration can be requested. This will return respectively the amount of time that the resource waited while available (idle) and the amount of time that the resource was busy.

8.2 Control Parameters

<table>
<tr><td></td><td colspan="4">Control Parameters</td></tr>
<tr><td></td><td>interTriggerTimer</td><td>triggerCount</td><td>probability</td><td>condition</td></tr>
<tr><td>MIN</td><td rowspan="3">Yes</td><td rowspan="3">No</td><td rowspan="4" colspan="2">No</td></tr>
<tr><td>MAX</td></tr>
<tr><td>MEAN</td></tr>
<tr><td>COUNT</td><td>No</td><td>Yes</td></tr>
<tr><td>SUM</td><td>Yes</td><td>No</td></tr>
</table>

Result requests can be applied to InterTriggerTimer on the same process model elements that accept it as input. In such a case, the result request of type MIN returns the minimum time before a trigger occurred, the result request of type MAX returns the maximum time before a trigger occurred and the result request of type MEAN returns the mean time before the trigger occurred. The result request of type SUM returns the total time before a trigger occurred.

The COUNT of the triggerCount parameter can be requested on all BPMN Events, Activities, Gateways and Sequence Flows elements. It returns the number of time that element was triggered by a token (started).

In the specific context of result requests on the COUNT for the triggerCount, it is also possible to add Control Parameters to the following BPMN elements:

- Activities with decomposition
- Process

Note that the fact that you can add Control Parameters to these elements does not infer that you can parameterize them as input.

8.3 Resource Parameters

	Resource Parameters			
	availability	quantity	role	selection
MIN	No			**Yes**
MAX				
MEAN	No			
COUNT				
SUM				

The result request of type MIN and the result request of type MAX of the selection parameter can be applied on the same process model elements that support it as input. It returns the least and most selected BPMN Resource.

8.4 Cost Parameters

	Cost Parameters	
	fixedCost	unitCost
MIN	No	
MAX		
MEAN		
COUNT		
SUM	**Yes**	

The result request of type SUM of Cost Parameters can be applied on the same process model elements that support it as input. It returns the total cost for that BPMN element.

In the specific context of result requests for the SUM, it is also possible to add Cost Parameters to the following BPMN elements:

- Activities with decomposition
- Process

Note that the fact that you can add Cost Parameters to these elements does not infer that you can parameterize them as input.

8.5 Property Parameters

	Property Parameters	
	property	queueLength
MIN	No	**Yes**
MAX		
MEAN		
COUNT	No	
SUM	No	

Result requests can only be applied to queueLength for MIN, MAX and MEAN. The result request of type MIN returns the minimum number of tokens in the queue for the element, the result request of type MAX returns the maximum number of tokens in the queue for the element. The result request of type MEAN returns the mean number of tokens in the queue for the element.

8.6 Priority Parameters

Result requests cannot be applied to Priority parameters.