Business Process Simulation Working Group (BPSWG)

BPSim 1.0 Implementer's Guide

Version History

Date	Version	Author	Comments
	0.1	TS	Skeleton structure
7 Feb 12	0.2	TS	Added 'Repair car' use case and a few notes about required content.
29 Feb 12	0.3	TS	• Adjusted use case style to be more "fully-dressed" than "casual" • Corrected use case scope and level • Added Lloyd's Loan Origination use case in the common format (using this one as the first as it is simpler). • Added the simulation parts of the Loan example.
06 Mar 12	0.4	AM	Added the Car repair simulation parts
06 Mar 12	0.5	AM	Updated the Car repair model
07 Mar 12	0.6	TS	Cleanup after our first iteration on the embedded questions.
07 Mar 12	0.7	DG	• Renamed Document • Page Layout and diagram inserts
16 Mar 12	0.8	DG	• Removed existing examples and replaced them with 3 new ones • Page layout
22 Mar 12	0.9	TS	• Contribute new car repair use case • Merge technical support use case from AM
26 Mar 12	0.9.1	AM	• Technical support process – description of simulation scenarios

27 Mar 12	0.9.2	TS	• Introduce simulation for each example following our discussion:
			    ○ Example 1 to introduce simulation to validate structural soundness
			    ○ Example 2 to cover structural and temporal perspectives (i.e. adding analysis of cycle time, wait time etc.)
			    ○ Example 3 to cover structural, temporal, cost and resource perspectives (and perhaps priority). In other words adding considerations like utilization maximisation.

27 Mar 12	0.9.3	AM	• Put context regarding the goal trade off to be achieved simulating the technical support process
28 Mar 12	0.9.4	TS	• Reverse order of Simulation Goal and Approach sections as discussed on email • Fix incorrect heading styles in example 2 (promote simulation scenario to level 2).
30 Mar 12	0.10	AM / DG / TS	• First draft of simulation scenario goals to share with Working Group.
19 Apr 12	0.10.1	TS	• Remove suggestion of product evaluation from example 1 introduction. • Reinforce the scope (all scenarios must share one process model) • Split example 1 into two scenarios (the first not triggering the signal events, the second including that data) • Clarify reference of conditions being on gateway to be on sequence flows in example 2. • Adopt eight hour day in example 2 to ensure discussion of resources and calendars does not happen till example 3.
23 Apr 12	0.10.2	AM	Example 3 : Technical Support • Changed the order of the questions to be answered using simulation • Put the simulation data • Defined each simulation scenario
09 May 12	0.10.3	AM	Example 3 : Technical Support • Exploring results for each simulation scenario
10 May 12	0.10.4	AM	Example 3 : Technical Support • Corrected the diagram, change the task *Inform customer the issue is going to be escalated*, to manual type (not indicated on the previous version) • Deeper exploration of each simulation scenario according to time perspective
25 May 12	0.11.0	TS	Release to working group.
13 Jun 12	0.11.1	TS	Example 1: Scenario 1 detailed including Visio, XPDL and BPMN files.
16 Jun 12	0.11.2	TS	Example 2: Scenario 1 detailed.
09 Aug 12	0.12.0	TS	Incorporated feedback from GH on 0.11. XML description of example 1 scenario 1.
15 Sep 12	0.13.0	TS	Example 2: Scenario 1 detailed.

			Updated XML snippets for 0.4 of the XSD
20 Nov 12	0.14.0	TS	• Set final 1.0 schema URI. • Example 1: Scenario 1 added result snippet. • Example 1: Scenario 2 detailed. • Example 2: Scenario 1 added result snippet. • Example 3: detailed excepting resources.
20 Dec 12	0.15.0a	TS	• Added scenario level parameters to all examples. • Ran examples 2 and 3 with one vendor's implementation. • Changed normal distributions for time to truncated normal and added explanatory text. • Changed start event and receive event distributions to Triangular in examples 2 & 3. • Restructured example 3 so that it follows the same pattern as 1 and 2 with goals and simulation data aligned to scenarios rather than lumped together. Removed suggestions of how tools may present results. • Removed example 3 scenario 4. • Added XML snippets for definition and selection of resource simulation parameters and result requests. • Updated request for results from InterTriggerTimer to TriggerCount.
7 Jan 13	0.15.0b	TS	• Add need to specify inherits on second and subsequent scenarios. • Corrections to sample BPMN and XPDL files.
20 Jan 13	0.15.0c	TS	• Added property parameters and expression to Car Repair example. • Add calendar definition snippet to third example. • Corrections to sample BPMN and XPDL files.
7 Feb 13	0.16.0	TS	• Updated namespace to 1.0 release. • Added content to References section. • Corrected inter trigger start interval in example 2 to Triangular distribution. • Grammar corrections • Removed (empty) Glossary • Added section 'Serialization examples'
14 Feb 13	0.16.1	TS	• Simplification of example 1, as discussed. • Conclusions added for example 1.
24 Feb 13	0.17.0	TS	• Various XML corrections: ○ corrected mostLikely to mode for triangularDistributions ;

			○ prefixed identifiers in XPDL with underscore to make them valid QNames. ○ various ordering of elements modified to pass schema validation.

Table of Contents

1 Introduction

1.1 Intended Audience

This document is intended as an introduction to the specification for people and organizations who are:

- ❖ Intending to implement a modeling tool capable of importing and fexporting simulation extensions along with process model in either BPMN or XPDL file formats.
- ❖ Intending to support the simulation of process models containing the simulation extensions.
- ❖ Modelers of business processes already familiar with BPMN process models but who need an introduction to the nature and location of the simulation extensions.

1.2 Purpose

1.3 Introduction to process simulation

This guide is not a complete guide to simulation, some basic points for effective Business Process Simulation are listed below. For further reading two books are recommended, although there is a large body of information written on simulation.

- Simulation Modeling & Analysis – Law & Kelton
- Simulation, The Practice of Model Development and Use – Robinson

Business Process simulation can be used at different levels of complexity, from simple diagram validation and understanding through to resource optimisation and service level agreement determination. What is crucial for success is that the model is at the correct level of granularity for the issue being investigated through simulation, suitable data is used and appropriate result statistics are collected.

Simulation experimentation can be thought of in a similar way to scientific experiments, a 'control' scenario and changes that allow comparison and cause & effect to be understood. Predicting outcomes within a tight tolerance potentially needs great care and longer experimentation, comparison i.e. concluding one scenario is better than another is much safer.

Similarly comparing a simulation generated set of results against 'real life' is also potentially misleading, hence the use of a 'control' scenario that represents the 'As-Is' is often preferable.

Simulation can be powerful and can use probability distributions to represent reality as opposed to constant values. When randomness is introduced multiple replications should be used. A replication is the same scenario but with a different sequence of random variables being produced, similar to a sequence of coin tosses being repeated.

1.3.1 Use of historical data

The use of historical data can be supported by the specification in two ways, either by supplying the actual numbers as parameters using ENUM, 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.

1.4 Scope of the specification

The specification considers a number of scenarios based on the same process model. Changes to the process model, for example combining two tasks into one, require separate simulation data as the existing data no longer applies. Of course tools may choose to offer assistance in updating the simulation data.

Simulation of a process model is in fact a form of ‘execution’, albeit different to the execution in a operational environment. As such a reasonably complete model is required. The BPSim standard does provide some mechanisms for simulating partially complete models e.g. the use of control parameters to replace message flow.

2 Example 1: Repairing a motor vehicle

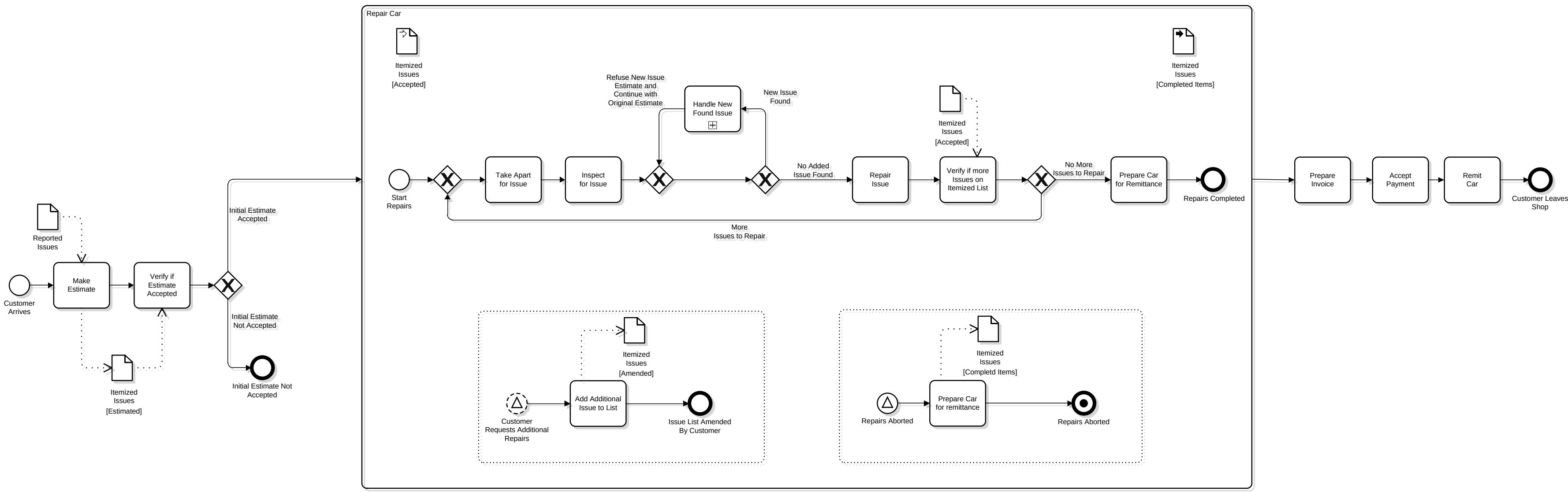
2.1 Use Case: Walk-in customer with car issue(s)

Primary Actor	Customer Service Representative (CSR).
Secondary Actors	Mechanic, Customer
Scope	“System” means all computer systems combined.
Level	Summary
Trigger	Customer walks into the repair shop with an immediate issue.

2.1.1 Process Description

1. The customer walks into the repair shop and explains one or more symptoms to the CSR.
2. The CSR makes a preliminary diagnosis or suggests investigations to provide a diagnosis together with an estimate for these. The customer may choose to accept or not this proposal.
3. If the estimate is accepted the car proceeds to the repair shop for the mechanic to look at it.
    a. The mechanic investigates each item on the estimate and either repairs it, discovers a further item to look into or agrees it will be aborted.
    b. If further items are discovered the customer may either approve or reject each one. This continues until all items are complete or rejected.
    c. During this stage the customer may also call with a further item to be added to the list.
4. An invoice for the items performed is prepared by the CSR and presented to the customer.
5. The customer settles the bill and takes the car away.

2.2 BPMN 2.0 Diagram of: Walk in customer with car issue(s)



2.3 Simulation scenario 1: Validate control perspective of primary path through process model

2.3.1 Approach / Hypothesis

In this scenario we assume that neither simulation nor process execution has yet been performed.

As we will see, it will also demonstrate how simulation can highlight unintended behaviours of the process model.

In this first scenario the simulation data will not trigger either of the signal events ('Customer requests additional repairs' and 'Abort repairs'). Additional scenarios may trigger these paths. However, in line with the scope of the specification, the process model must not change between different scenarios.

2.3.2 Goals

- Validate the control perspective of the process, in other words that it does not get stuck in unexpected loops or bypass expected paths ; and
- To provide a baseline set of data. This data will be compared to our expected behaviour in the real world to provide some confidence that the simulation model is valid.

2.3.3 Identification of simulation parameters

2.3.3.1 Simulation parameters

We will run the simulation for 40 hours, which models a single working week without going to the level of detail of specifying exact working hours, shifts etc.

Each replication will run the simulation with different random seeds. Choosing a suitable number for this depends on the amount of variation in the process being modelled and the degree of confidence needed from the results. A low number of replications will suit an example such as this, especially when merely trying to validate the control perspective.

- Duration: 40 hours.
- Replications: 3
- Time unit: minutes

2.3.3.2 Process Trigger(s)

- Customer 'walk-ins' vary through the day but since we are not exploring the temporal perspective in this scenario we will define an average arrival interval of 24 minutes (equates to 30 customers in a typical 12 hour opening day).
- Number of issues to repair:
  - distribution: truncated normal
  - mean: 2
  - standard deviation: 1
  - minimum: 1

2.3.3.3 Activity Durations

- This scenario does not investigate the temporal perspective so we can omit the tasks' durations.

2.3.3.4 Decision points

- Initial estimate accepted : 20 / day (two thirds).
- Additional issue found : 25% of cases.

2.3.3.5 Resources

- This scenario does not deal with the resource perspective.

2.3.3.6 Results requested

To meet the goals set above we want to receive counts of the number of process instances that pass along each of the following paths:

- Process instances started.
- "Initial estimate not accepted" end event.
- "Start repairs" start event.
- "Repair issue" task.
- "Repairs completed" end event.
- "Customer leaves shop" end event.

2.3.4 How the model provides for that data to be captured

This is the first time we have looked at the serialization format for the simulation experiment data so we have to perform some basic setup steps. Later examples will build on this foundation. In summary we will:

- Add simulation model to the process model ;
- Setup a scenario ;
- Add scenario parameters ;
- Add input parameters to the scenario element ;
- Add property expressions to decrement the number of repair issues ;
- Add expressions to test whether we need to exit the repair loop;
- Add results requests to the scenario element ;
- Add result requests to BPMN elements.

The complete solution is provided in the accompanying BPMN and XPDL files.

2.3.4.1 Add simulation model to the process model

No matter whether the process model is expressed as BPMN or XPDL adding a simulation model consists of the same step: Adding the root element and declaring the namespace.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<semantic:definitions id="CarRepair" name="Car Repair Process"
            targetNamespace="http://www.example.com/definitions/CarRepair"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xmlns:di="http://www.omg.org/spec/DD/20100524/DI"
            xmlns:bpmndi="http://www.omg.org/spec/BPMN/20100524/DI"
            xmlns:dc="http://www.omg.org/spec/DD/20100524/DC"
            xmlns:semantic="http://www.omg.org/spec/BPMN/20100524/MODEL">
    ...
    <semantic:process isExecutable="false" id="_6">
        ...
    </semantic:process>
    <bpmndi:BPMNDiagram
        ...
    </bpmndi:BPMNDiagram>
    <semantic:relationship type="ProcessAnalysisData">
        <semantic:extensionElements>
            <bpsim:ProcessAnalysisData
                  xmlns:bpsim="http://www.bpsim.org/schemas/1.0">
                ...
            </bpsim:ProcessAnalysisData>
        </semantic:extensionElements>
        …
    </semantic:relationship>
</semantic:definitions>
```

XML snippet 1: Declaring simulation namespace and root element in a BPMN file

```
<?xml version="1.0" encoding="UTF-8"?>
<Package Id="CarRepair" Name="Car Repair Process"
            xmlns="http://www.wfmc.org/2009/XPDL2.2">
    <WorkflowProcesses>
        <WorkflowProcess Id="6">
            …
        </WorkflowProcess>
    </WorkflowProcesses>
    <bpsim:ProcessAnalysisData
            xmlns:bpsim="http://www.bpsim.org/schemas/1.0">
        ...
    </bpsim:ProcessAnalysisData>
</Package>
```

XML snippet 2: Declaring simulation namespace and root element in an XPDL file

2.3.4.2 Setup a scenario

Having established the basic model element the next step is to add a scenario to the model.

```
<bpsim:ProcessAnalysisData xmlns:bpsim="http://www.bpsim.org/schemas/1.0">
    <bpsim:Scenario id="default" name="Scenario 1: Main flow only"
                  author="Tim Stephenson" created="2012-06-13T09:47:00">
        ...
    </bpsim:Scenario>
```

```
</bpsim:ProcessAnalysisData>
```

XML snippet 3: Declaring a scenario

2.3.4.3 Add scenario parameters

The first data to add to a scenario will control the simulation experiment's replications, duration and define the time units.

```
<bpsim:Scenario id="default" name="Scenario 1: Main flow only"
                  author="Tim Stephenson" created="2012-06-13T09:47:00">
      <bpsim:ScenarioParameters replication="3" baseTimeUnit="min">
            ...
            <bpsim:Duration>
                  <bpsim:DurationParameter value="PT40H"/>
            </bpsim:Duration>
      </bpsim:ScenarioParameters>
</bpsim:Scenario>
```

2.3.4.4 Add input parameters to the scenario

All parameters are added to the scenario using ElementParameters. There are a number of different types of ElementParameters to serve different purposes.

Parameter type	Purpose
Control	Specify the control flow of a business process element.
Time	Capture time intervals and are defined from an external observer point of view.
Resource	Specify the resources employed by a business process element.
Cost	Specify all costs of an activity fixed or variable, human or non-human.
Priority	Control the priority of the associated business process element.
Instance	Specify simulation values for data instances used by the business process and by implication offer an alternate way to specify most of the other parameter types.

The following model extract shows the control and property parameter that together make up the process trigger for this scenario as described within Process Trigger(s). Note that the specification requires max to be set for a TruncatedNormalDistribution though we don't have a real maximum to set in this example, so we just set it high so it does not affect the values generated.

```
<bpsim:Scenario id="default" name="Scenario 1: Main flow only"
              author="Tim Stephenson" created="2012-06-13T09:47:00">
      <bpsim:ElementParameters elementId="_6-51263">
            <bpsim:ControlParameters>
                  <bpsim:InterTriggerTimer>
                        <bpsim:DurationParameter value="PT24M"/>
                  </bpsim:InterTriggerTimer>
            </bpsim:ControlParameters>
```

```
            <bpsim:PropertyParameters>
              <bpsim:Property name="numberOfIssues">
                  <bpsim:TruncatedNormalDistribution
                        max="1000" mean="2" standardDeviation="1" min="1"/>
              </bpsim:Property>
          </bpsim:PropertyParameters>
      </bpsim:ElementParameters>
      ...
</bpsim:Scenario>
```

XML snippet 4: Process trigger for Example 1, Scenario 1

2.3.4.5 Add property expressions to decrement the number of repair issues

```
<ns1:ElementParameters elementId="6-1645638">
      <ns1:PropertyParameters>
            <ns1:Property name="noOfIssues"
                        value="getProperty('noOfIssues') - 1"/>
      </ns1:PropertyParameters>
</ns1:ElementParameters>
```

2.3.4.6 Add expressions to test whether we need to exit the repair loop

```
<ns1:ElementParameters elementId="6-990368">
      <ns1:ControlParameters>
            <ns1:Condition name="no more issues">
                  <ns1:ExpressionParameter instance="noOfIssues"
                              value="getProperty('noOfIssues') = 0"/>
            </ns1:Condition>
      </ns1:ControlParameters>
</ns1:ElementParameters>
<ns1:ElementParameters elementId="6-1048090">
      <ns1:ControlParameters>
            <ns1:Condition name="have more issues">
                  <ns1:ExpressionParameter instance="noOfIssues"
                              value="getProperty('noOfIssues') != 0"/>
            </ns1:Condition>
      </ns1:ControlParameters>
</ns1:ElementParameters>
```

2.3.4.7 Add result requests to the scenario element

```
<bpsim:ScenarioParameters replication="3" baseTimeUnit="min">
      <bpsim:Start>
            <bpsim:ResultRequest>count</bpsim:ResultRequest>
      </bpsim:Start>
</bpsim:ScenarioParameters>
```

2.3.4.8 Add result requests to BPMN elements

Result requests for each BPMN element are handled in the same way.

```
<bpsim:ElementParameters elementId="_6-2599544">
    <bpsim:ControlParameters>
        <bpsim:InterTriggerTime>
            <bpsim:ResultRequest>count</bpsim:ResultRequest>
        </bpsim:InterTriggerTime>
```

```
    </bpsim:ControlParameters>
</bpsim:ElementParameters>
```

XML snippet 5: Request instance count results

2.4 Simulation scenario 2: Validate control perspective of primary and secondary paths

2.4.1 Approach / Hypothesis

This scenario shows how scenario 1 may be extended with additional simulation data for one of the two signal events (‘Abort repairs’).

2.4.2 Goals

- As scenario 1, though for the additional path too.

2.4.3 Identification of simulation parameters

2.4.3.1 Process Trigger(s):

- Repairs aborted – Let us assume that one customer per day needs to cancel the repair for one or another reason.

2.4.3.2 Activity Durations

This example does not investigate the temporal perspective so we can omit the tasks’ durations.

2.4.3.3 Decision points

No additional decision points need to be modeled.

2.4.3.4 Resources

This example does not deal with the resource perspective.

2.4.3.5 Results requested

In addition to the instance counts already requested in scenario 1, we will request:

- Number of instances of “Repairs aborted” start event.
- Number of instances of “Repairs aborted” end event.

2.4.4 How the model provides for that data to be captured

Scenarios are cumulative so that we need only model any additions or overrides to scenario 1 here. In summary this means:

- Define an additional scenario element using the ‘inherits’ attribute to denote that this will extend the first scenario.
- Add parameters to the secondary path’s start event.
- Add result request parameters as in scenario 1.

2.4.4.1 Define an additional scenario element

Additional scenarios can simply be added as an ordered list. The order controls the overriding semantics.

```
<bpsim:ProcessAnalysisData xmlns:bpsim="http://www.bpsim.org/schemas/1.0">
    <bpsim:Scenario id="default" name="Scenario 1: Main flow only"
                  author="Tim Stephenson" created="2012-06-13T09:47:00">
        ...
    </bpsim:Scenario>
    <bpsim:Scenario id="S2" inherits="default"
            name="Scenario 2: Main and secondary paths" ...>
        ...
    </bpsim:Scenario>
</bpsim:ProcessAnalysisData>
```

XML snippet 6: Adding an additional scenario

2.4.4.2 Add parameters to the secondary path's start event

This is similar to the way that control parameters of the main start event were added in scenario 1. However it may be worth noting that in order to cancel one repair per day we need to discard the 10 repairs whose customer rejected the initial estimate leaving 1 of 20 repairs to be cancelled, which is to say a probability of 0.05.

```
<bpsim:Scenario
      <bpsim:ElementParameters elementId="_6-2166825">
            <bpsim:ControlParameters>
                  <bpsim:InterTriggerTimer>
                        <bpsim:ResultRequest>count</bpsim:ResultRequest>
                        <bpsim:DurationParameter value="PT0M"/>
                  </bpsim:InterTriggerTimer>
                  <bpsim:Probability>0.05</bpsim:Probability>
                  <bpsim:TriggerCount>1</bpsim:TriggerCount>
            </bpsim:ControlParameters>
      </bpsim:ElementParameters>
      ...
</bpsim:Scenario>
```

XML snippet 7: Parameters for the 'Repairs aborted' event.

2.5 Conclusions and further investigations

Where the repair is aborted is not by the process model. *When* the repair is aborted is not the subject of this simulation scenario, which confines itself to control parameters only, but in the interests of not leaving it undefined setting the inter trigger time to 0 indicates it should happen straight after the process start.

Therefore if it is desired to know more about the way the repair can be aborted the reader may wish to:

1. Investigate the temporal perspective of this process model ; and / or
2. Change the process model and perform a new set of simulation scenarios.

3 Example 2: Originating a home loan

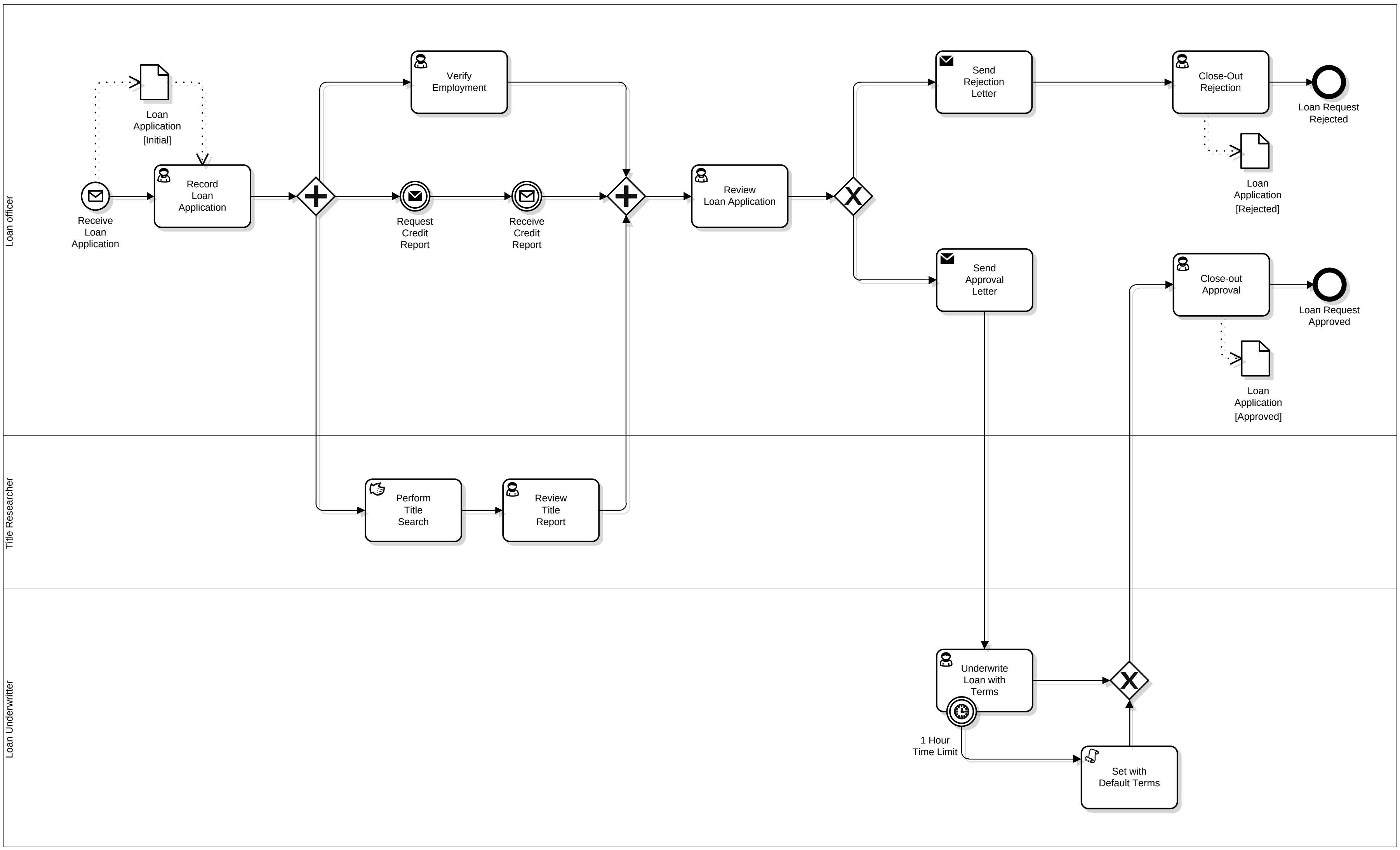
3.1 Use Case: Originate a home loan

Primary Actor	Loan Officer (may be at a Clerk or Supervisor level).
Secondary Actors	Borrower, Title Researcher, Underwriter
Scope	"System" means all computer systems combined.
Level	Summary
Trigger	Borrowers arrive throughout the standard business day to apply for a home loan, providing a completed loan application to a Loan Officer.

3.1.1 Process Description

1. A Loan Officer receives the completed loan application from the borrower, and enters it into the loan origination system.
2. A Loan Officer then verifies provided employment information, recording the result of his/her investigations.
3. The Borrower's credit score and report are requested of and received from the three credit bureaux in a consolidated form.
4. A Title Researcher searches the county title records for the property in question, and then determines whether or not the property is correctly listed and free of liens.
5. A Loan Officer assembles and reviews the case file (loan application with employment verification, credit score and report, and title results) to approve or reject the application.
6. If rejected, a Loan Officer sends a rejection notice to the Borrower, and then closes out the rejected case file.
7. If approved, a Loan Officer sends an approval notice to the Borrower, and then forwards the case file to an Underwriter.
8. An Underwriter underwrites the loan based on the case file, returning it to the Loan Officer, but if this takes more than an hour, then standard loan terms are assigned.
9. The Loan Officer then closes out the approved case file.

3.2 BPMN 2.0 Diagram of: Originate a home loan



3.3 *Simulation scenario 1: Explore temporal perspective*

3.3.1 Approach / Hypothesis

As in the previous use case we assume that neither simulation nor process execution has yet been performed so we wish to validate the control perspective of the process model.

We will then go on to consider the temporal aspects of the model. Specifically we will look at finding values for these questions:

• What is the mean cycle time (time-in-system)?

• What is the mean wait time for each loan as a whole? And broken down by task?

• Is all the process sustainable, in other words does a backlog of work build up and is all work completed by end of day?

As before this is initially to confirm that the model approximates to our real-world experience. Then we will consider some changes to the temporal parameters to evaluate their impact on the overall cycle time.

3.3.2 Goal

Identify potential bottlenecks in the AS-IS process and potential ways to alleviate them.

3.3.3 Identification of simulation parameters

The simulation inputs are as follows:

3.3.3.1 Simulation parameters

We will run this example with the same scenario parameters as previously.

- Duration: 40 hours.
- Replications: 3
- Time unit: minutes

3.3.3.2 Process Triggers

- 30 loan applications for an eight hour business day (approximately one every 16 minutes on average). The particular hours worked are not specified in this example as we are assuming all resources are available for the period, holding such resource considerations as calendars, breaks etc. to the third example.

  We'll use a triangular distribution to model the arrivals as follows:

  - Mode: 16 mins as the most likely value (mode) ;
  - Min: 10 and ;
  - Max: 30
- The 'Receive credit report' event needs to be populated because the specification says that a missing value will be interpreted as blocking the process from continuing.

  We'll use a triangular distribution to model the arrivals as follows:

  - Mode: 5 mins as the as the most likely value (mode) ;

- Min: 4 and ;
- Max: 6

3.3.3.3 Activity Durations

- Each user or manual task's duration is modelled using a truncated normal distribution. The truncation is necessary to avoid nonsensical negative durations and it is typically a good idea to use that for all time-based distributions. For the sake of the example let's truncate at 0 mins though a value greater than 0 may be more accurate in reality. These are the means and standard deviations:
  - Record Loan Application: 20 mins, σ 1
  - Verify employment: 30 mins, σ 4
  - Perform title search: 1 hour, σ 2
  - Review title report: 20 mins, σ 2
  - Review loan application: 30 mins, σ 4
  - Close out rejection: 5 mins, σ 0.25
  - Close out approval: 10 mins, σ 0.25
  - Underwrite loan with terms: 45 mins, σ 10, truncated at 60 mins
- Each automated task is of constant duration
  - Send rejection letter: 1 minute
  - Send approval letter: 1 minute
  - Set default terms: 1 minute

3.3.3.4 Decision points

- 8 loans are approved at the 'Review loan application' activity and accordingly follow the subsequent sequence flow to 'Send approval letter'.

Depending on the number of Underwriters and the constraint that default terms are applied after an hour of wait time the simulation will determine how many loans receive default terms so no input is required on the timer event.

3.3.3.5 Resources

This example does not deal with the resource perspective.

3.3.3.6 Results requested

To support the goals of this example in exploring the temporal perspective we will request the minimum, maximum and mean processing time for all user and manual tasks as well as for the process as a whole.

3.3.4 How the model provides for that data to be captured

In summary we will:

- Add parameters controlling the occurrence of start events.
- Add parameters controlling the processing time of each activity.
  - Truncated normal distributions
  - Fixed (constant) durations

- Add parameters controlling the flows from decision points.
- Add result parameters for the minimum, maximum and mean processing times.

The complete solution is provided in the accompanying BPMN and XPDL files.

3.3.4.1 Add parameters controlling the occurrence of start events

To simulate the arrival of start events (here people submitting load applications) we specify a control parameter representing the time between each of the event triggers. Each trigger starts a new process instance in accordance with the semantic of BPMN start events.

As in the first example we add a ProcessAnalysisData element as the model root and to that add the scenario and parameters.

```
<bpsim:ProcessAnalysisData xmlns:bpsim="http://www.bpsim.org/schemas/1.0">
    <bpsim:Scenario id="default" name="Scenario">
        <bpsim:ElementParameters elementId="_6-195657">
            <bpsim:ControlParameters>
                <bpsim:InterTriggerTimer>
                    bpsim:TriangularDistribution
                                min="10" max="30" mostLikely="16"/>
                </bpsim:InterTriggerTimer>
            </bpsim:ControlParameters>
        </bpsim:ElementParameters>
        ...
    </bpsim:Scenario>
</bpsim:ProcessAnalysisData>
```

XML snippet 8: Specifying that a new process instance will start every 16 minutes.

3.3.4.2 Add parameters controlling the processing time of each activity

Several aspects of the time taken for an activity to be completed may be modelled for simulation. The simplest of these, the first approximation is the processing time. Used alone this can approximate the time that the activity is queued waiting for resources to carry it out *and* the time for those resources to actually process the task. Or if a pool of suitable resources are specified for the activity the simulation engine can calculate how long is spent queuing.
Here we will not specify the resource pool but simply the processing time for the 'Record loan application' task.

```
<bpsim:Scenario id="default" name="Scenario">
    ...
    <bpsim:ElementParameters elementId="_6-223729">
        <bpsim:TimeParameters>
            <bpsim:ProcessingTime>
                <bpsim:TruncatedNormalDistribution
                          mean="20" standardDeviation="1" min=”0”/>
            </bpsim:ProcessingTime>
        </bpsim:TimeParameters>
    </bpsim:ElementParameters>
    ...
</bpsim:Scenario>
```

XML snippet 9: A normal distribution for an activity truncated only on the low side

3.3.4.3 Modeling the duration for the Underwriting Terms activity

This activity is an interesting example because we would like to issue standard terms if the underwriter has not provided them within an hour. The time waiting for an underwriter to be available and the time for the underwriter to review the application and provide terms all contribute the 60 minute limit. As noted above the simplest way to model this is to specify an upper limit to a truncated normal distribution for the processing time.

```
<bpsim:ElementParameters elementId="_6-1419903">
    <bpsim:TimeParameters>
        <bpsim:ProcessingTime>
            <bpsim:TruncatedNormalDistribution
                     mean="45" standardDeviation="10" min="0" max="60"/>
        </bpsim:ProcessingTime>
    </bpsim:TimeParameters>
</bpsim:ElementParameters>
```

XML snippet 10: Truncated Normal Distribution for processing time of an activity

3.3.4.4 Modeling durations for system and script tasks

It may typically be assumed that system and script tasks will be relatively short-lived, that is that compared to the user and manual tasks the time to process them will be relatively less significant. As such they may often be modeled as constant durations.

```
<bpsim:ElementParameters elementId="_6-1597888">
     <bpsim:TimeParameters>
          <bpsim:ProcessingTime>
               <bpsim:DurationParameter value="PT1M"/>
          </bpsim:ProcessingTime>
      </bpsim:TimeParameters>
</bpsim:ElementParameters>
```

XML snippet 11: Constant duration processing time parameter

3.3.4.5 Modeling probabilities of each flow from a decision point

There are two flows from the decision point labelled ‘Approved?’ corresponding to whether the loan application is approved or rejected. We specified above that there would be eight of the thirty loans approved. The specification requires that the total weighting of all possible paths adds up to 1 so we convert these to 0.27 for the approval weighting and 0.73 for rejection as follows:

```
<bpsim:ElementParameters elementId="6-1139375">
    <bpsim:ControlParameters>
        <bpsim:Probability>
            <bpsim:FloatingParameter value="0.73"/>
        </bpsim:Probability>
    </bpsim:ControlParameters>
</bpsim:ElementParameters>
<bpsim:ElementParameters elementId="6-1383663">
    <bpsim:ControlParameters>
        <bpsim:Probability>
            <bpsim:FloatingParameter value="0.23"/>
        </bpsim:Probability>
    </bpsim:ControlParameters>
</bpsim:ElementParameters>
```

XML snippet 12: Probability of following flows from the 'Approved?' decision point

3.3.4.6 Add scenario-level temporal result parameters

In order to request results for the minimum, maximum and mean processing times we add a ScenarioParameters element.

```
<bpsim:Scenario ...>
    <bpsim:ScenarioParameters>
        <bpsim:Duration>
            <bpsim:ResultRequest>min</bpsim:ResultRequest>
            <bpsim:ResultRequest>max</bpsim:ResultRequest>
            <bpsim:ResultRequest>mean</bpsim:ResultRequest>
        </bpsim:Duration>
    </bpsim:ScenarioParameters>
</bpsim:Scenario>
```

XML snippet 13: Result request for process durations

3.3.4.7 Task-level temporal result parameters

Requesting durations for the tasks are very similar but are attached to TimeParameters.

```
<bpsim:ElementParameters elementId="_6-223729">
    <bpsim:TimeParameters>
        <bpsim:ProcessingTime>
            <bpsim:ResultRequest>min</bpsim:ResultRequest>
            <bpsim:ResultRequest>max</bpsim:ResultRequest>
            <bpsim:ResultRequest>mean</bpsim:ResultRequest>
            <bpsim:NormalDistribution mean="20" standardDeviation="1"/>
        </bpsim:ProcessingTime>
    </bpsim:TimeParameters>
</bpsim:ElementParameters>
```

XML snippet 14: Processing time results for the "Record Loan Application" task

3.3.5 Conclusions and further investigations

TODO

4 Example 3: Technical support

4.1 Use Case: Provide solution to a technical problem reported by a customer

Primary Actor	Front Office.
Secondary Actors	1st Level Technical Support Agent, 2nd Level Technical Support Agent and Supplier.
Scope	“System” means all computer systems combined.
Level	Summary
Trigger	Customers call to a call center requiring a solution for a technical problem about a service, equipment or software provided.

4.1.1 Process Description

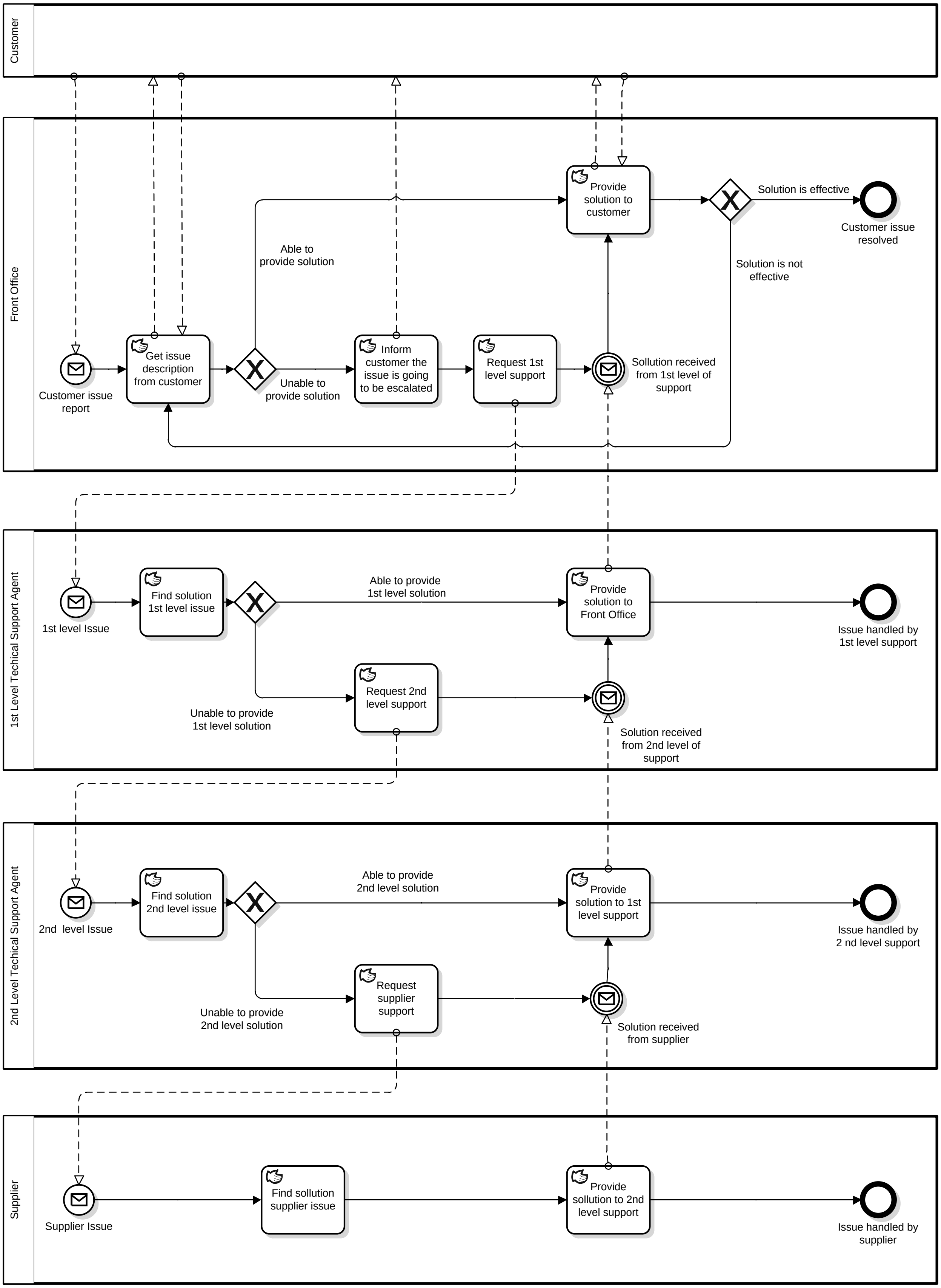
The customer calls the contact center and reports an issue about underperforming service or faulty equipment or software. The Front Office collects information from the Customer and tries to provide a solution directly to the Customer on the other end of the line, otherwise they inform the Customer the issue is going to be escalated to technical experts and they will be contacted again soon. When the Front Office receives the solution from the technical experts, they contact the customer and try to close the issue; otherwise they inform the Customer that the issue is going to be further escalated.

When the issue is escalated to the 1st Level Technical Support Agent, the agent tries to provide a solution to the Front Office; otherwise they request further assistance from the 2nd Level Technical Support Agent and forward the solution to the Front Office when a solution has been provided.

When the issue is escalated to the 2nd Level Technical Support Agent, the agent tries to find a solution for the 1st Level Technical Support Agent; otherwise they request further assistance from the Supplier and forward the solution to the 1st Level Technical Support Agent when provided.

When the Supplier receives a request from the 2nd Level Technical Support Agent they provide a solution to the reported issue.

4.2 BPMN 2.0 Diagram of: Customer calls in with a technical issue



4.3 Simulation scenario 1: Explore control flow perspective

4.3.1 Approach / Hypothesis

Technical support is a process responsible for managing the lifecycle of all problems reported by customers. This process is consumer centric, because if technical support operates poorly the effect will be amplified through the public channels such as, social networks, consumer forums and the company image it will be at risk.

The problems reported by the customers are not all the same and different expertise is necessary. This is why there are multiple layers of technical support and even the supplier of the service/equipment/software is included. Some problems are from customers that don't have the basic skills and this kind of issue can be easily solved by the first line support and there are complex problems that need in depth investigation.

Poor performance is being experienced. In this first scenario we will explore who is involved in resolving issues.

4.3.2 Goals

The goal of this scenario is providing answers create a frequency baseline:

- What are the most / less used paths?

4.3.3 Identification of simulation parameters

The simulation inputs are as follows:

4.3.3.1 Simulation parameters

- Duration: 1 month
- Replications: 3
- Time unit: minutes

4.3.3.2 Process Triggers

- There are 2200 new calls raised by customers in each 24h period. Approximately this many arrive for each time period. These will be initially be modelled using a triangular distribution across the entire period as follows:
    - Mode: 2
    - Min: 0.35
    - Max: 2

  In the later scenarios we attempt a close approximation using calendars.

4.3.3.3 Decision points

- Under the Front office responsibility:
    - 60% of the times is able to provide a solution;
    - 15% of the times the solution is not effective.
- Under the 1st level Technical support agent:
    - 70% of the times is able to provide a solution;

- Under the 2nd level Technical support agent:
    - 80% of the time is able to provide a solution.

4.3.3.4 Results requested

Results provided should indicate important Key Performance Indicators (KPI's) like the ones below.

Description	**Rationale**
Tentative instances	Total number of requests that the customer communicated to the contact center
Processed instances	Total number of requests attended by the contact center
1st level escalated instances	Total number of requests attended by the 1st level support team
2nd level escalated instances	Total number of requests attended by the 2nd level support team
Supplier escalated instances	Total number of requests attended by the supplier team
Completed instances	Total number of requests that reached the *Customer issue resolved* End state
In progress instances	Total number of requests that are being processed by the contact center and did not reach the *Customer issue resolved* End state, for the simulation period duration
Completeness ratio	= (Completed instances /Processed instances) X 100
Loss Ratio	= (Processed instances/ Tentative instances) X 100

The results should drive the user to conclude how well the contact center performs regarding ability to process the incoming requests (Loss Ratio) and process all the attended requests (Completeness Ratio). These numbers are the starting point to establish an incoming service level agreement.

4.3.4 How the model provides for that data to be captured

There are no additional constructs needed to run this scenario.

4.3.5 Conclusions and further investigations

As we can see, most of the process instances are processed thought the first layer where the customer interacts with the Front Office. A lack of balancing the right number of Front office resources, will result in incapacity of answer the incoming requests, escalate to the Technical layers and provide solutions to customers. Thus one of the challenges is to balance the trade-off of keep the process flowing and provide an acceptable service level for incoming requests.

Ranking[1]	Activity Description
1	Get issue description from customer
2	Inform customer the issue is going to be escalated
3	Request 1st level Support
4	Provide solution to customer
5	Find solution 1st level issue
6	Request 2nd level support
7	Provide solution to Front Office
8	Find solution 2nd level issue
9	Request supplier support
10	Provide solution to 1st level support
11	Find solution supplier issue
11	Provide solution to 2nd level support

These control perspective walkthrough, should be explored in further perspectives like:

- Under the Front office responsibility:
  - 40% of the time is able to provide a solution;
  - 35% of the time the solution is not effective.
- Under the 1st level Technical support agent:
  - 60% of the time is able to provide a solution;
- Under the 2nd level Technical support agent:
  - 70% of the time is able to provide a solution.

[1] By descending order - Total number of instances processed - It counts the absolute number of requests that were processed. It sums the requests that were processed more than once in the same activity. This can happen because the solution provided was not effective. Hence the request is reprocessed.

4.4 *Simulation scenario 2: Explore temporal perspective*

4.4.1 Approach / Hypothesis

Here we will add time parameters to explore the performance of the support center.

4.4.2 Goals

The goal of this scenario is providing answers to the following question:

- Performance baseline:
    - What is mean time for providing a solution to the customer that reported a problem? And how can we use this data to setup internal acceptable (SLA).

In this scenario we want to understand what happens to the following KPI's:

- Completeness ratio
- Loss Ratio
- Cycle time

4.4.3 Identification of simulation parameters

This scenario uses the same control parameters mentioned above and additionally provides the following:

4.4.3.1 Activity Durations

These tasks' durations are normally distributed. Once again we will truncate these on the minimum side at 0. These are the values:

Activity Description	**Mean (min)**[2]	**Standard Deviation (min)**
Get issue description from customer	4,0	0,5
Provide solution to customer	10,0	2,5
Find solution 1st level issue	4,0	0,5
Provide solution to Front Office	1,0	0,5
Find solution 2nd level issue	7,0	1,0
Request supplier support	3,0	1,0
Provide solution to 1st level support	1,0	0,5
Find solution supplier issue	300,0	30,0
Provide solution to 2nd level support	2,0	0,5

[2] (,) means decimal separator, i.e. 2,5 means 2 and a half minutes.

These tasks' durations are constant.

Activity Description	Duration (min)[3]
Request 1st level Support	0,5
Request 2nd level support	0,5

4.4.3.2 Results requested

- Min durations of the process instance.
- Max durations of the process instance.
- Mean durations of the process instance.

4.4.4 How the model provides for that data to be captured

This scenario does not require any new constructs.

4.4.5 Conclusion and further investigations

TODO

Further investigations that may be interesting to pursue include:

- Activity *Get issue from customer* increases to double or triple;
- Activity *Provide solution to customer* increases to double or drops to half ;
- Activity *Find solution 1st level issue* increases to double or triple;
- Activity *Find solution 2ndt level issue* increases to double or triple;

Results should be explored in best case scenario and worst case scenario. Combinations of case scenario and worst case scenario under each resource type responsibility should also be provided, i.e. degradation / improvement of *Get issue from customer* and *Provide solution to customer* activities, while keeping the others with baseline values and so forth.

4.5 Simulation scenario 3: Explore resource perspective

4.5.1 Goals

The goal of this scenario is providing answers to the following question:

- Balancing the workforce with these constraints:
  - Number of reported requests during the day;

In the previous scenario, we concluded the baseline provides a calculated loss of incoming requests.

Before we concentrate on resource leveling let's balance Front Office resources in order to meet a desired acceptable service level (Loss Ratio) >= 95%. With this Front Office setup, the contact

[3] (,) means decimal separator, i.e. 2,5 means 2 and a half minutes.

center can assure that 95% of the customer's issues are processed at first tentative contact. This is a service level agreement the company wants to comply with.

4.5.2 Identification of simulation parameters

This scenario again builds on the parameters of previous scenarios.

4.5.2.1 Process Triggers

Here we are going to override the process start interval from scenario 1 as follows.

- 6 a.m. to 9 a.m. : 170 per hour
- 9 a.m. to 12 p.m. : 70 per hour
- 12 p.m. to 3 p.m. : 110 per hour
- 3 p.m. to 6 p.m. : 60 per hour
- 6 p.m. to 10 p.m. : 140 per hour
- 10 p.m. to 1 a.m. : 90 per hour
- 1 a.m. to 6 a.m. : 30 per hour

4.5.2.2 Resources

The Contact Center operates 24/7. And supports customers based on a particular time zone. This means it is expected that it will handle very few requests during the night.

Supplier operates nonstop, 9 a.m. to 10 p.m. on weekdays only.

	6 a.m. to 9 a.m.	9 a.m. to 12 p.m.	12 p.m. to 3 p.m.	3 p.m. to 6 p.m.	6 p.m. to 10 p.m.	10 p.m. to 1 a.m.	1 a.m. to 6 a.m.
Front Office	200	90	130	60	150	100	40
1st Level Technical Support Agent	3	3	3	3	3	3	0
2nd Level Technical Support Agent	2	2	2	2	2	2	0
Supplier	0	1	1	1	1	0	0

Performers are assumed to be 100% available. In other words this is the sole task they perform. Front Office cannot perform 1st Level Technical support and so on.

Comment on how 'black box' pool is simulated using control parameters in place of a full model

4.5.3 How the model provides for that data to be captured

As before, this scenario contains simulation parameters for activity durations and weightings of various flows from decision points please refer to previous examples on how those are stored in the simulation model. Here we will:

- Define calendars
- Add parameters controlling start events that each apply during a part of the day as defined in a calendar object.
- Add parameters controlling the resources' availability, also associated with calendar objects.
- Add resource selection expressions to the activities.

The complete solution is provided in the accompanying BPMN and XPDL files.

4.5.3.1 Define Calendars for use by the scenario

```
      <bpsim:Scenario ... >
            ...
            <ns1:Calendar id="C1" name="6-9am">BEGIN:VCALENDAR
BEGIN:VEVENT
DTSTAMP:20121220T202424
UID:1356035064823@localhost
DTSTART:20121220T060000
DTEND:20121220T090000
RRULE:FREQ=DAILY
END:VEVENT
PRODID:PAF Editor
VERSION:2.0
END:VCALENDAR</ns1:Calendar>
            <ns1:Calendar id="C2" name="9-noon">BEGIN:VCALENDAR
BEGIN:VEVENT
DTSTAMP:20121220T202500
UID:1356035100473@localhost
DTSTART:20121220T090000
DTEND:20121220T120000
RRULE:FREQ=DAILY
END:VEVENT
PRODID:PAF Editor
VERSION:2.0
END:VCALENDAR</ns1:Calendar>
            <ns1:Calendar id="C3" name="noon-3pm">BEGIN:VCALENDAR
BEGIN:VEVENT
DTSTAMP:20121220T202529
UID:1356035129707@localhost
DTSTART:20121220T120000
DTEND:20121220T150000
RRULE:FREQ=DAILY
END:VEVENT
PRODID:PAF Editor
VERSION:2.0
END:VCALENDAR</ns1:Calendar>
            <ns1:Calendar id="C4" name="3-6pm">BEGIN:VCALENDAR
BEGIN:VEVENT
DTSTAMP:20121220T202631
UID:1356035191095@localhost
DTSTART:20121220T150000
DTEND:20121220T180000
```

```
RRULE:FREQ=DAILY
END:VEVENT
PRODID:PAF Editor
VERSION:2.0
END:VCALENDAR</ns1:Calendar>
            <ns1:Calendar id="C5" name="6-10pm">BEGIN:VCALENDAR
BEGIN:VEVENT
DTSTAMP:20121220T202706
UID:1356035226921@localhost
DTSTART:20121220T180000
DTEND:20121220T220000
RRULE:FREQ=DAILY
END:VEVENT
PRODID:PAF Editor
VERSION:2.0
END:VCALENDAR</ns1:Calendar>
            <ns1:Calendar id="C6" name="10-1am">BEGIN:VCALENDAR
BEGIN:VEVENT
DTSTAMP:20121220T202726
UID:1356035246845@localhost
DTSTART:20121220T100000
DTEND:20121221T010000
RRULE:FREQ=DAILY
END:VEVENT
PRODID:PAF Editor
VERSION:2.0
END:VCALENDAR</ns1:Calendar>
            <ns1:Calendar id="C7" name="1-6am">BEGIN:VCALENDAR
BEGIN:VEVENT
DTSTAMP:20121220T202752
UID:1356035272706@localhost
DTSTART:20121220T010000
DTEND:20121220T060000
RRULE:FREQ=DAILY
END:VEVENT
PRODID:PAF Editor
VERSION:2.0
END:VCALENDAR</ns1:Calendar>
            <ns1:Calendar id="C8" name="9am-10pm-weekdays">BEGIN:VCALENDAR
BEGIN:VEVENT
DTSTAMP:20121220T203015
UID:1356035303891@localhost
DTSTART:20121220T090000
DTEND:20121220T220000
RRULE:FREQ=WEEKLY;BYDAY=MO,TU,WE,TH,FR
END:VEVENT
PRODID:PAF Editor
VERSION:2.0
END:VCALENDAR</ns1:Calendar>
            ...
      </bpsim:Scenario>
```

4.5.3.2 Add parameters controlling start events associated with a calendar

Since this is the first time we have introduced different inter trigger times at different times of day, let's look at how we associate each time parameter with a different calendar. We will model the inter-trigger interval as constant within each period of the day.

```
    <bpsim:Scenario ... >
```

```
  ...
  <bpsim:ElementParameters elementId=" FFF ">
      <bpsim:ControlParameters>
          <bpsim:InterTriggerTimer>
              <bpsim:DurationParameter value="PT21S" validFor="6-9am"/>
              <bpsim:DurationParameter value="PT51S" validFor="9-noon"/>
              <bpsim:DurationParameter value="PT33S" validFor="noon-3pm"/>
              <bpsim:DurationParameter value="PT1M"  validFor="3-6pm"/>
              <bpsim:DurationParameter value="PT26S" validFor="6-10pm"/>
              <bpsim:DurationParameter value="PT40S" validFor="10-1am"/>
              <bpsim:DurationParameter value="PT2M"  validFor="1-6am"/>
          </bpsim:InterTriggerTimer>
      </bpsim:ControlParameters>
   </bpsim:ElementParameters>

    ...

  </bpsim:Scenario>
```

4.5.3.3 Add parameters controlling the resources' availability

```
<bpsim:Scenario ...>
  ...
  <bpsim:ElementParameters elementId="frontOffice">
        <bpsim:ResourceParameters>
              <bpsim:Quantity>
                    <bpsim:NumericParameter value="200" />
              </bpsim:Quantity>
        </bpsim:ResourceParameters>
  </bpsim:ElementParameters>
  <bpsim:ElementParameters elementId="firstLevelSupport">
        <bpsim:ResourceParameters>
              <bpsim:Quantity>
                    <bpsim:NumericParameter value="3" />
              </bpsim:Quantity>
        </bpsim:ResourceParameters>
  </bpsim:ElementParameters>
  <bpsim:ElementParameters elementId="secondLevelSupport">
        <bpsim:ResourceParameters>
              <bpsim:Quantity>
                    <bpsim:NumericParameter value="2" />
              </bpsim:Quantity>
        </bpsim:ResourceParameters>
  </bpsim:ElementParameters>
  <bpsim:ElementParameters elementId="supplier">
        <bpsim:ResourceParameters>
              <bpsim:Quantity>
                    <bpsim:NumericParameter value="1" />
              </bpsim:Quantity>
        </bpsim:ResourceParameters>
  </bpsim:ElementParameters>
  ...
</bpmsim:Scenario>
```

4.5.3.4 Add resource selection expressions to the activities

```
<bpmsim:Scenario>
  ...
  <bpsim:ElementParameters elementId="_10-57">
        <bpsim:ResourceParameters>
```

```
            <bpsim:Selection>
                  <bpsim:ExpressionParameter
                              value="getResource('frontOffice', 1)" />
            </bpsim:Selection>
      </bpsim:ResourceParameters>
</bpsim:ElementParameters>
<bpsim:ElementParameters elementId="_10-138">
      <bpsim:ResourceParameters>
            <bpsim:Selection>
                  <bpsim:ExpressionParameter
                              value="getResource('frontOffice', 1)" />
            </bpsim:Selection>
      </bpsim:ResourceParameters>
</bpsim:ElementParameters>
<bpsim:ElementParameters elementId="_10-235">
      <bpsim:ResourceParameters>
            <bpsim:Selection>
                  <bpsim:ExpressionParameter
                              value="getResource('frontOffice', 1)" />
            </bpsim:Selection>
      </bpsim:ResourceParameters>
</bpsim:ElementParameters>
<bpsim:ElementParameters elementId="_10-920">
      <bpsim:ResourceParameters>
            <bpsim:Selection>
                  <bpsim:ExpressionParameter
                              value="getResource('frontOffice', 1)" />
                  </bpsim:Selection>
      </bpsim:ResourceParameters>
</bpsim:ElementParameters>
<bpsim:ElementParameters elementId="_10-303">
      <bpsim:ResourceParameters>
            <bpsim:Selection>
                  <bpsim:ExpressionParameter
                              value="getResource('firstLevelSupport', 1)" />
            </bpsim:Selection>
      </bpsim:ResourceParameters>
</bpsim:ElementParameters>
<bpsim:ElementParameters elementId="_10-380">
      <bpsim:ResourceParameters>
            <bpsim:Selection>
                  <bpsim:ExpressionParameter
                              value="getResource('firstLevelSupport', 1)" />
            </bpsim:Selection>
      </bpsim:ResourceParameters>
</bpsim:ElementParameters>
<bpsim:ElementParameters elementId="_10-458">
      <bpsim:ResourceParameters>
            <bpsim:Selection>
                  <bpsim:ExpressionParameter
                              value="getResource('firstLevelSupport', 1)" />
            </bpsim:Selection>
      </bpsim:ResourceParameters>
</bpsim:ElementParameters>
<bpsim:ElementParameters elementId="_10-526">
      <bpsim:ResourceParameters>
            <bpsim:Selection>
                  <bpsim:ExpressionParameter
                              value="getResource('secondLevelSupport', 1)" />
```

```
                </bpsim:Selection>
            </bpsim:ResourceParameters>
        </bpsim:ElementParameters>
        <bpsim:ElementParameters elementId="_10-617">
            <bpsim:ResourceParameters>
                <bpsim:Selection>
                    <bpsim:ExpressionParameter
                            value="getResource('secondLevelSupport', 1)" />
                </bpsim:Selection>
            </bpsim:ResourceParameters>
        </bpsim:ElementParameters>
        <bpsim:ElementParameters elementId="_10-670">
            <bpsim:ResourceParameters>
                <bpsim:Selection>
                    <bpsim:ExpressionParameter
                            value="getResource('secondLevelSupport', 1)" />
                </bpsim:Selection>
            </bpsim:ResourceParameters>
        </bpsim:ElementParameters>
        <bpsim:ElementParameters elementId="_10-794">
            <bpsim:ResourceParameters>
                <bpsim:Selection>
                    <bpsim:ExpressionParameter
                            value="getResource('supplier', 1)" />
                </bpsim:Selection>
            </bpsim:ResourceParameters>
        </bpsim:ElementParameters>
        <bpsim:ElementParameters elementId="_10-847">
            <bpsim:ResourceParameters>
                <bpsim:Selection>
                    <bpsim:ExpressionParameter
                            value="getResource('supplier', 1)" />
                </bpsim:Selection>
            </bpsim:ResourceParameters>
        </bpsim:ElementParameters>
        ...
    <bpmsim:Scenario>
```

4.5.3.5 Result requests

In addition to the results requested for other scenarios we will request for each resource:

- Sum of processing time.
- Sum of wait time.
- Count of the times a resource is triggered.

These are represented in the model as follows:

```
<bpsim:ElementParameters elementId="developer">
  ...
  <bpsim:TimeParameters>
    <bpsim:ProcessingTime>
      <bpsim:ResultRequest>sum</bpsim:ResultRequest>
    </bpsim:ProcessingTime>
    <bpsim:WaitTime>
      <bpsim:ResultRequest>sum</bpsim:ResultRequest>
    </bpsim:WaitTime>
  </bpsim:TimeParameters>
  <bpsim:ControlParameters>
```

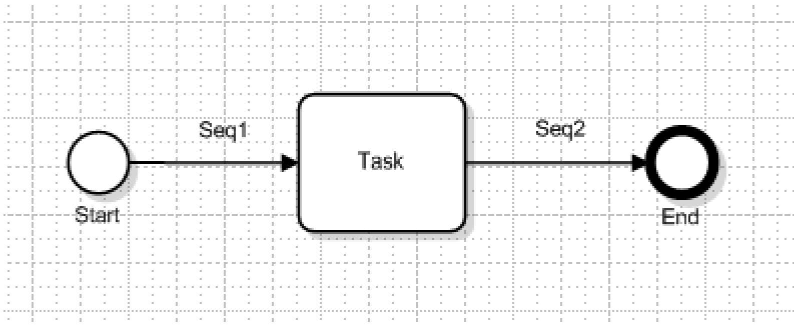
```
      <bpsim:TriggerCount>
        <bpsim:ResultRequest>count</bpsim:ResultRequest>
      </bpsim:TriggerCount>
    </bpsim:ControlParameters>
    ...
  </bpsim:ElementParameters>
```

4.5.3.6 Conclusions and further investigations

TODO Put conclusion here.

5 Serialization examples

5.1 *Time Parameters*



5.1.1 Duration

You can set the duration for the ***Task*** to 5 minutes using the processing time.

```
<ElementParameters elementRef="task">
    <TimeParameters>
        <ProcessingTime>
            <DurationParameter value="PT5M"/>
        </ProcessingTime>
    </TimeParameters>
</ElementParameters>
```

5.1.2 Lag Time

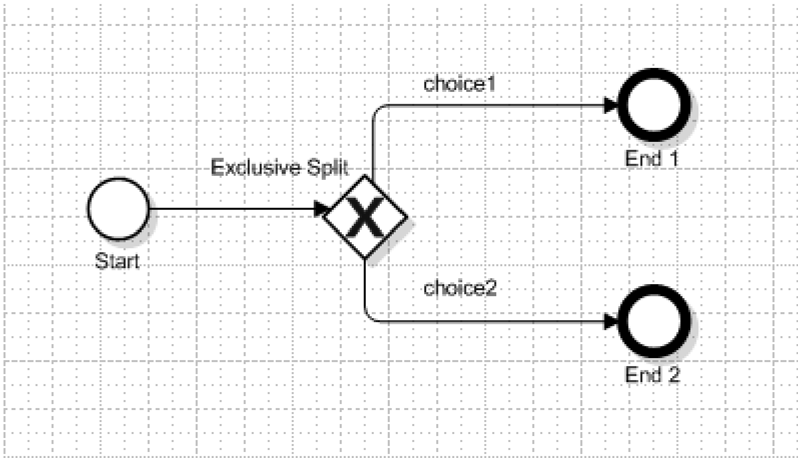
You can set the lag time of ***Seq1*** to 10 seconds using the wait time.

```
<ElementParameters elementRef="seq1">
    <TimeParameters>
        <WaitTime>
            <DurationParameter value="PT10S"/>
        </ WaitTime >
    </TimeParameters>
</ElementParameters>
```

5.2 *Control Parameters*

5.2.1 Routing using Probabilities

The probability attribute can be used to control splits inside a BPMN drawing.

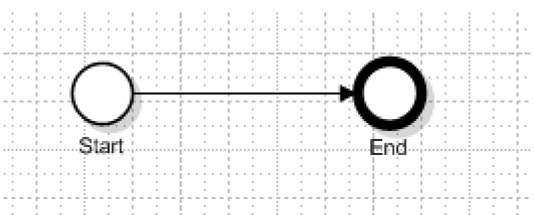


To determine the odds of a split going 25% to ***choice1*** and 75% to ***choice2***, you can use the control parameters.

```
<ElementParameters elementRef="choice1">
      <ControlParameters>
            <Probability>
                  <FloatingParameter value="0.25"/>
            </Probability>
      </ControlParameters>
</ElementParameters>
<ElementParameters elementRef="choice2">
      <ControlParameters>
            <Probability>
                  <FloatingParameter value="0.75"/>
            </Probability>
      </ControlParameters>
</ElementParameters>
```

5.2.2 Control Process Instantiation

To control the start of start of a process:



To start this process every 5 minutes, you can use the inter trigger timer on the start.

```
<ElementParameters elementRef="start">
      <ControlParameters>
            <InterTriggerTimer>
                  <DurationParameter value="PT5M"/>
            </InterTriggerTimer>
      </ControlParameters>
</ElementParameters>
```

You can also determine the number of times a process starts using the starting instance count parameter (starts 100 tokens).

```
<ElementParameters elementRef="start">
      <ControlParameters>
```

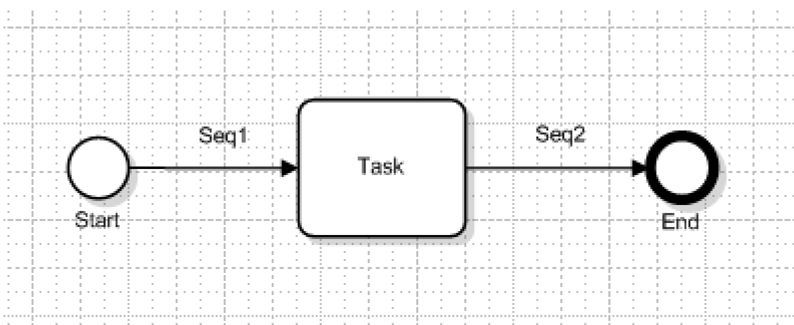
```
                <TriggerCount>
                        <NumericParameter value="100"/>
                </ TriggerCount >
        </ControlParameters>
</ElementParameters>
```

You could also combine the inter trigger with the instance count to start 100 instances but start one every 5 minutes.

```
<ElementParameters elementRef="start">
        <ControlParameters>
                < TriggerCount >
                        <NumericParameter value="100"/>
                </ TriggerCount >
                <InterTriggerTimer>
                        <DurationParameter value="PT5M"/>
                </InterTriggerTimer>
        </ControlParameters>
</ElementParameters>
```

5.3 Using advanced parameterisation

Using the following simple diagram, in this section we presents various advances way of defining the duration of the task.



5.3.1 Distribution

Here is how you can express the duration to be random from 3 to 10 minutes.

```
<Scenario id="default" name="Scenario">
        <ScenarioParameters  baseTimeUnit="min"/>
        <ElementParameters elementRef="task">
                <TimeParameters>
                        <ProcessingTime>
                                <UniformDistribution min="3" max="10"/>
                        </ProcessingTime>
                </TimeParameters>
        </ElementParameters>
</Scenario>
```

Here we used a uniform distribution from 3 to 10 and we defined at the scenario level that the base time unit is minutes.

We could do the same in seconds.

```
<Scenario id="default" name="Scenario">
        <ScenarioParameters baseTimeUnit="s"/>
        <ElementParameters elementRef="task">
                <TimeParameters>
```

```
                <ProcessingTime>
                      <UniformDistribution min="180" max="600"/>
                </ProcessingTime>
          </TimeParameters>
    </ElementParameters>
</Scenario>
```

Both of these examples are equivalent

5.3.2 User Distribution

We could also specify that the duration is 5 minutes 90% of the time but 10 minutes 10% of the time.

```
<TimeParameters>
      <ProcessingTime>
            <UserDistribution>
                  <UserDistributionDataPoint probability="0.9">
                        <DurationParameter value="PT5M"/>
                  </UserDistributionDataPoint>
                  <UserDistributionDataPoint probability="0.1">
                        <DurationParameter value="PT10M"/>
                  </UserDistributionDataPoint>
            </UserDistribution>
      </ProcessingTime>
</TimeParameters>
```

5.3.3 Enumeration (historical data)

We could express the duration of the task using an enumeration of historical data gathered from an existing system. In this example, we measured 5 different duration for the task.

```
<ElementParameters elementRef="task">
      <TimeParameters>
            <ProcessingTime>
                  <EnumParameter>
                        <DurationParameter value="PT3M4S"/>
                        <DurationParameter value="PT6M10S"/>
                        <DurationParameter value="PT2M44S"/>
                        <DurationParameter value="PT3M55S"/>
                        <DurationParameter value="PT4M11S"/>
                  </EnumParameter>
            </ProcessingTime>
      </TimeParameters>
</ElementParameters>
```

5.4 Using calendars

We can vary the duration using a calendar. For instance, we could do this example where the task duration is normally 5 minutes but on Friday afternoon it takes 7 minutes.

```
<Scenario id="default" name="Scenario">
      <ScenarioParameters baseTimeUnit="min"/>
      <ElementParameters elementRef="task">
            <TimeParameters>
                  <ProcessingTime>
                        <DurationParameter value="PT5M"/>
                        <DurationParameter validFor="C1" value="PT7M"/>
                  </ProcessingTime>
            </TimeParameters>
      </ElementParameters>
```

```
      <Calendar id="C1" name="Friday Afternoon">BEGIN:VCALENDAR
BEGIN:VEVENT
DTSTAMP:20120525T142704
UID:1337970424871@localhost
DTSTART:20020101T120000
DTEND:20020101T170000
RRULE:FREQ=WEEKLY;BYDAY=FR
END:VEVENT
PRODID:PAF Editor
VERSION:2.0
END:VCALENDAR
      </Calendar>
</Scenario>
```

5.5 Using an expression

You can also use an expression to determine the value of the duration. This example uses the XPATH function to retrieve the Instance parameter named duration.

```
<ElementParameters elementRef="task">
      <TimeParameters>
            <ProcessingTime>
                  <ExpressionParameter
                              value="bpsim:getProperty('duration')"/>
            </ProcessingTime>
      </TimeParameters>
      <InstanceParameters/>
</ElementParameters>
```

5.6 Results

5.6.1 Time Parameters

5.6.1.1 Minimum/Maximum and Mean on a Processing Time

You can request the minimum, maximum and mean time on the Time Parameter Processing Time.

```
<ElementParameters elementRef="task">
      <TimeParameters>
            <ProcessingTime>
                  <ResultRequest>min</ResultRequest>
                  <ResultRequest>max</ResultRequest>
                  <ResultRequest>mean</ResultRequest>
            </ProcessingTime>
      </TimeParameters>
</ElementParameters>
```

This will give an output that will have the following format.

```
<ElementParameters elementRef="task">
      <TimeParameters>
            <ProcessingTime>
                  <DurationParameter value="PT1M" result=”min”/>
                  <DurationParameter value="PT5M" result=”max”/>
                  <DurationParameter value="PT3M" result=”mean”/>
            </ProcessingTime>
      </TimeParameters>

</ElementParameters>
```

5.6.1.2 Count/Sum of a Processing Time

We can continue on the same example but now request the count and the sum of the processing time

```
<ElementParameters elementRef="task">
      <TimeParameters>
            <ProcessingTime>
                  <ResultRequest>count</ResultRequest>
                  <ResultRequest>sum</ResultRequest>
            </ProcessingTime>
      </TimeParameters>
</ElementParameters>
```

This will give an output of the following format.

```
<ElementParameters elementRef="task">
      <TimeParameters>
            <ProcessingTime>
                  <DurationParameter value="PT15M" result=”sum”/>
                  <IntegerParameter value="5" result=”count”/>
            </ProcessingTime>
      </TimeParameters>
</ElementParameters>
```

5.6.2 Control Parameters

5.6.2.1 Requesting everything about an InterTriggerTimer on a signal intermediate event

Requesting the inter trigger time min/max and mean duration waiting for a trigger

```
<ElementParameters elementRef="signal">
      <ControlParameters>
            <InterTriggerTimer>
                  <ResultRequest>min</ResultRequest>
                  <ResultRequest>max</ResultRequest>
                  <ResultRequest>mean</ResultRequest>
                  <ResultRequest>sum</ResultRequest>
            </InterTriggerTimer>
      </ControlParameters>
</ElementParameters>
```

Result is:

```
<ElementParameters elementRef="signal">
      <ControlParameters>
            <InterTriggerTimer>
                  <DurationParameter value="PT2M" result=”min”/>
                  <DurationParameter value="PT5M" result=”max”/>
                  <DurationParameter value="PT4M" result=”mean”/>
                  <DurationParameter value="PT24M" result=”sum”/>
            </InterTriggerTimer>
      </ControlParameters>
</ElementParameters>
```

5.6.3 Replications effects on results

When more than one replication is used, the output should be tagged to a specific replication identifier. For instance doing 3 replications and wanting to know the mean processing time:

```
<ScenarioParameters replication="3"/>
<ElementParameters elementRef="task">
      <TimeParameters>
            <ProcessingTime>
                  <ResultRequest>mean</ResultRequest>
            </ProcessingTime>
      </TimeParameters>
</ElementParameters>
```

The expected result should now have the replication instance identifier present

```
<ElementParameters elementRef="task">
      <TimeParameters>
            <ProcessingTime>
                  <DurationParameter value="PT4M" result="mean"
                              instance="3218nd-21332dsda"/>
                  <DurationParameter value="PT5M" result="mean"
                              instance="321h2d-321321dd"/>
                  <DurationParameter value="PT6M" result="mean"
                              instance="321jd9d-3123213"/>
            </ProcessingTime>
      </TimeParameters>
</ElementParameters>
```

6 References

- BPSim web site: http://www.bpsim.org/
- 1.0 Specification: http://bpsim.org/specifications/1.0/WFMC-BPSWG-2012-01.pdf
- 1.0 XML Schema: http://bpsim.org/schemas/1.0/