Implementer's Guide

Document Status – Released

Feb 18, 2016

Version V1.0.2

Table of Contents

1 Introduction

1.1 Intended Audience

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

- Intending to implement a modeling tool capable of importing and exporting 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.

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.

Whilst the purpose of simulation is typically to improve on an outcome from 'real-life', care should be taken to ensure that the data used is representative. Obviously that real historical data represents a snapshot that may be either more optimistic or more pessimistic than is typical. More subtly, using real data may either include or exclude rare outliers (data points significantly different to typical). Hence it is theoretically more sound to determine the distribution that the sample comes from. This allows you to run multiple replications based upon the execution history, as opposed to running the execution history which is only a single replication.

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 or generating a distribution. When generating 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 to ensure a fair comparison. Of course tools may choose to offer assistance in updating the simulation data when making such process model changes.

Simulation of a process model is in fact a form of 'execution', albeit different to the execution in an 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 simulate the triggering of events.

2 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/

3 Example 1: Repairing a motor vehicle

3.1 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.

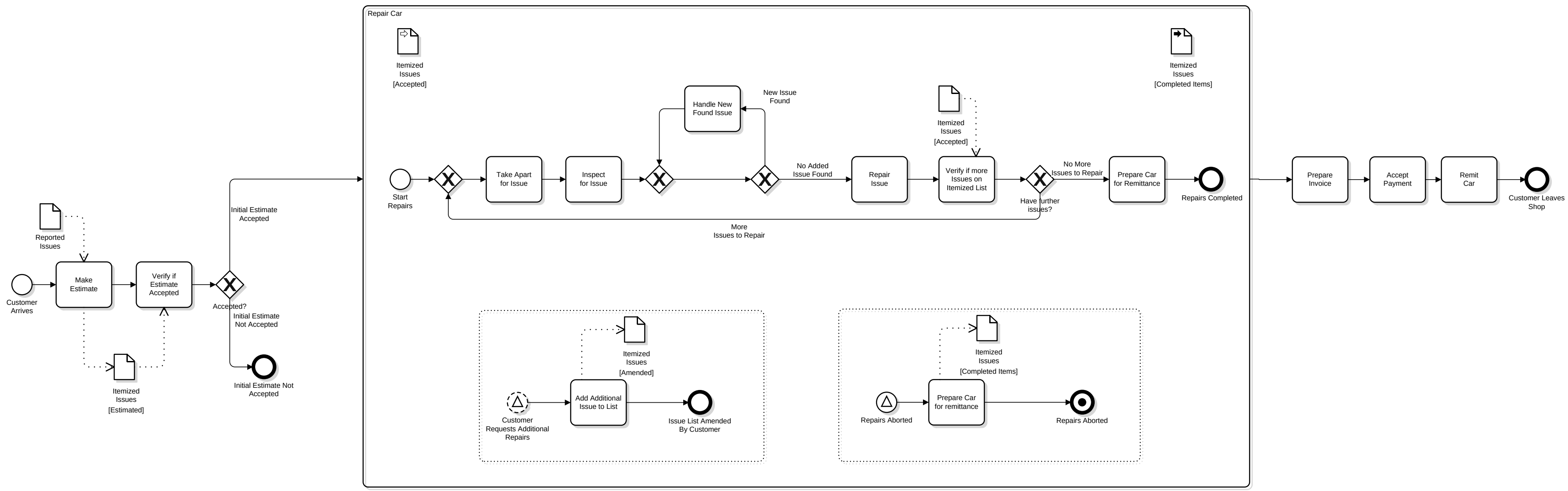
3.1.1 Process Description

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 with the customer to abandon it.
    b. If further items are discovered the customer may either approve or reject each one. This continues until all items are completed or rejected.
    c. During this stage the customer may also call with an additional 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.

For this example, the following BPMN diagram was prepared as part of a theoretical business process study and has been provided to the simulation modeling team. However, the BPSim specifications of the two study scenarios exercise a subset of this model which illustrates the use of BPSim to limit the scope of a study scenario to a subset of a larger model.

In the following diagram of a car repair process, the Signal Event "Customer Requests Additional Repairs" triggers "Add Additional Issue to List" which in turn, updates the "Itemized Issues" list. Parameters for this Signal Event are not specified in either of the two example scenarios which imply this component of the total process will not be exercised.

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



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

3.3.1 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.

3.3.2 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.

3.3.3 Identification of simulation parameters

3.3.3.1 Simulation parameters

We will run the simulation to model a single working week without going to the level of detail of specifying exact working hours, shifts etc. We will simply assume the garage is open 12 hours per day.

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: 60 hours.
- Replications: 3
- Time unit: minutes

3.3.3.2 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

3.3.3.3 Activity Durations

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

3.3.3.4 Decision points

- Initial estimate accepted : 20 / day (two thirds).
- Additional issue found : 25% of cases.
- Have further issues: based on a counter (propertyParameter) of the Itemized Issues.

3.3.3.5 Resources

- This scenario does not deal with the resource perspective.

3.3.3.6 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.

3.3.4 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 BPMN Process ;
- Add result requests to BPMN elements.

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

* Note about the use of expressions to simulate business logic
Simulators often provide the capability to execute code to represent business logic, e.g. rules for process flow and for resource allocation. In this example, we use a PropertyParameter on the process instance to represent the number of issues the customer requires to be fixed on his car. When an issue is fixed there is an expression to decrement this parameter, if a new issue is found this is incremented. The parameter is also used to determine choice of sequence flow from a gateway. This logic is expressed within the BPSim data and therefore is not visible directly on the BPMN diagram except by means of textual description. See section 3.3.4.5 and 3.3.4.6 for the BPSim serialization of this business logic.

3.3.4.1 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="BPSimData">
        <semantic:extensionElements>
            <bpsim:BPSimData
                  xmlns:bpsim="http://www.bpsim.org/schemas/1.0">
                ...
            </bpsim:BPSimData>
        </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:BPSimData
            xmlns:bpsim="http://www.bpsim.org/schemas/1.0">
        ...
    </bpsim:BPSimData>
```

```
</Package>
```

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

3.3.4.2 Setup a scenario

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

```
<bpsim:BPSimData xmlns:bpsim="http://www.bpsim.org/schemas/1.0">
    <bpsim:Scenario author="Tim Stephenson" created="2013-03-06T21:00:00"
id="S1" modified="2013-03-06T21:00:00" name="Scenario 1: Main flow without
sub-processes">
    </bpsim:Scenario>
</bpsim:BPSimData>
```

XML snippet 3: Declaring a scenario

3.3.4.3 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="S1" 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="PT60H"/>
            </bpsim:Duration>
      </bpsim:ScenarioParameters>
</bpsim:Scenario>
```

3.3.4.4 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**
Time	Capture time intervals and are defined from an external observer point of view.
Control	Specify the control flow of a business process element.
Resource	Specify the resources employed by a business process element.
Cost	Specify all costs of an activity fixed or variable, human or non-human.
Property	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.
Priority	Control the priority of the associated business process element.

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="S1" name="Scenario 1: Main flow only"
                    author="Tim Stephenson" created="2012-06-13T09:47:00">
            <bpsim:ElementParameters elementRef="_51BDA265-2FF5-40CB-B68D-
1FBF9DAAA74C">
                  <bpsim:ControlParameters>
                        <bpsim:InterTriggerTimer>
                              <bpsim:DurationParameter value="PT24M"/>
                        </bpsim:InterTriggerTimer>
                  </bpsim:ControlParameters>
                  <bpsim:PropertyParameters>
                        <bpsim:Property name="noOfIssues">
                              <bpsim:TruncatedNormalDistribution
                                    max="1000"
                                    mean="2"
                                    min="1"
                                    standardDeviation="1"/>
                        </bpsim:Property>
                  </bpsim:PropertyParameters>
            </bpsim:ElementParameters>
            ...
      </bpsim:Scenario>
```

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

3.3.4.5 Add property expressions to decrement the number of repair issues

```
      <bpsim:ElementParameters elementRef="_071B5D0A-5225-4E55-9105-8D15169DAC96">
            <bpsim:PropertyParameters>
                  <bpsim:Property name="noOfIssues">
                        <bpsim:ExpressionParameter
                                    value="bpsim:getProperty('noOfIssues') -1"/>
                        </bpsim:Property>
                  </bpsim:PropertyParameters>
      </bpsim:ElementParameters>
```

XML snippet 5: expression to decrement property parameter

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

```
      <bpsim:ElementParameters elementRef="_9C46FDB0-2EBD-4B2F-A72A-0189A4D76A73">
            <bpsim:ControlParameters>
                  <bpsim:Condition>
                        <bpsim:ExpressionParameter
value="bpsim:getProperty('noOfIssues') > 0"/>
                  </bpsim:Condition>
            </bpsim:ControlParameters>
      </bpsim:ElementParameters>
      <bpsim:ElementParameters elementRef="_4799D2B0-A204-4A08-94ED-796C25D9AE57">
            <bpsim:ControlParameters>
                  <bpsim:Condition>
```

```
                    <bpsim:ExpressionParameter
                          value="bpsim:getProperty('noOfIssues') <= 0"/>
                    </bpsim:Condition>
              </bpsim:ControlParameters>
</bpsim:ElementParameters>
```

XML snippet 6: conditional flow expressions

3.3.4.7 Add result requests to the scenario element

```
<bpsim:Scenario...>
      <bpsim:ElementParameters elementRef="_6">
            <bpsim:ControlParameters>
                  <bpsim:TriggerCount>
                        <bpsim:ResultRequest>count</bpsim:ResultRequest>
                  </bpsim:TriggerCount>
            </bpsim:ControlParameters>
      </bpsim:ElementParameters>
      ...
</bpsim:Scenario>
```

3.3.4.8 Add result requests to BPMN elements

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

```
<bpsim:ElementParameters elementRef="_FA77D7D8-9C02-4006-942A-B003AEDA5E3C">
    <bpsim:ControlParameters>
        <bpsim:InterTriggerTime>
            <bpsim:ResultRequest>count</bpsim:ResultRequest>
        </bpsim:InterTriggerTime>
    </bpsim:ControlParameters>
</bpsim:ElementParameters>
```

XML snippet 7: Request instance count results

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

3.4.1 Approach / Hypothesis

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

3.4.2 Goals

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

3.4.3 Identification of simulation parameters

3.4.3.1 Process Trigger(s):

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

3.4.3.2 Activity Durations

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

3.4.3.3 Decision points

No additional decision points need to be modeled.

3.4.3.4 Resources

This example does not deal with the resource perspective.

3.4.3.5 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.

3.4.4 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.

3.4.4.1 Define an additional scenario element

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

```
<bpsim:BPSimData 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="S1"
            name="Scenario 2: Main and secondary paths" ...>
        ...
    </bpsim:Scenario>
</bpsim:BPSimData>
```

XML snippet 8: Adding an additional scenario

3.4.4.2 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:ElementParameters elementRef="_F9A272EE-D325-44CF-AFFC-4A615D2C9971">
      <bpsim:ControlParameters>
            <bpsim:Probability>
                  <bpsim:FloatingParameter value="0.05"/>
            </bpsim:Probability>
            <bpsim:InterTriggerTimer>
                  <bpsim:DurationParameter value="PT0M"/>
            </bpsim:InterTriggerTimer>
            <bpsim:TriggerCount>
                  <bpsim:ResultRequest>count</bpsim:ResultRequest>
                  <bpsim:NumericParameter value="1"/>
            </bpsim:TriggerCount>
      </bpsim:ControlParameters>
</bpsim:ElementParameters>
            ...
```

XML snippet 9: Parameters for the 'Repairs aborted' event.

3.5 Conclusions and further investigations

Where the repair is aborted is not defined in this 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.

4 Example 2: Originating a home loan

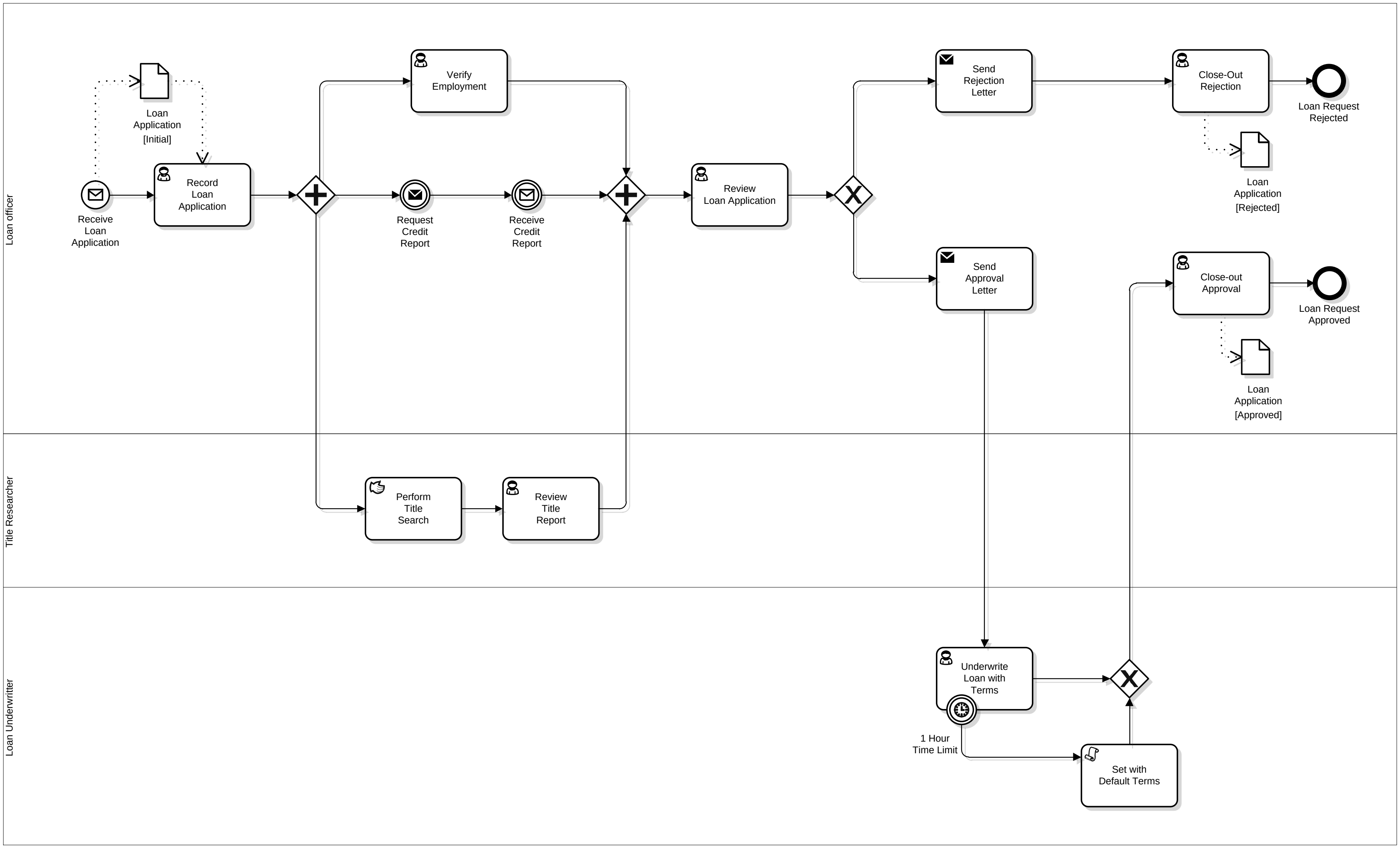
4.1 Use Case: Originate a home loan

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.

4.1.1 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.

4.2 BPMN 2.0 Diagram of: Originate a home loan



4.3 Simulation scenario 1: Explore temporal perspective

4.3.1 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.

4.3.2 Goal

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

4.3.3 Identification of simulation parameters

The simulation inputs are as follows:

4.3.3.1 Simulation parameters

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

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

4.3.3.2 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

4.3.3.3 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: 50 mins, σ 10, interrupted 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

4.3.3.4 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.

4.3.3.5 Resources

This example does not deal with the resource perspective.

4.3.3.6 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.

4.3.4 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.

4.3.4.1 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 BPSimData element as the model root and to that add the scenario and parameters.

```
    <bpsim:BPSimData xmlns:bpsim="http://www.bpsim.org/schemas/1.0">
        <bpsim:Scenario id="default" name="Scenario">
            <bpsim:ElementParameters elementRef="_4E504F9B-2618-424F-ABAA-
C4D4CC2D75E0">
                <bpsim:ControlParameters>
                    <bpsim:InterTriggerTimer>
                        bpsim:TriangularDistribution
                                    min="10" max="30" mode="16"/>
                    </bpsim:InterTriggerTimer>
                </bpsim:ControlParameters>
            </bpsim:ElementParameters>
            ...
        </bpsim:Scenario>
    </bpsim:BPSimData>
```

XML snippet 10: Specifying that a new process instance will start every 16 minutes.

4.3.4.2 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 elementRef="_2017EC19-4BD5-40D7-9014-
E7D337A68E01">
            <bpsim:TimeParameters>
                <bpsim:ProcessingTime>
```

```
                <bpsim:TruncatedNormalDistribution
                          mean="20" standardDeviation="1" min=”0” max=”1000”/>
            </bpsim:ProcessingTime>
        </bpsim:TimeParameters>
    </bpsim:ElementParameters>
    ...
</bpsim:Scenario>
```

XML snippet 11: A normal distribution for an activity

4.3.4.3 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 elementRef="_57EE71B2-5287-46A2-9B24-83A68131714C">
    <bpsim:TimeParameters>
        <bpsim:ProcessingTime>
            <bpsim:TruncatedNormalDistribution
                     mean="45" standardDeviation="10" min="0" max="60"/>
        </bpsim:ProcessingTime>
    </bpsim:TimeParameters>
</bpsim:ElementParameters>
```

XML snippet 12: Truncated Normal Distribution for processing time of an activity

4.3.4.4 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 elementRef="_D6383F72-7F7A-48F8-A742-A26F36A8DC10">
     <bpsim:TimeParameters>
          <bpsim:ProcessingTime>
               <bpsim:DurationParameter value="PT1M"/>
          </bpsim:ProcessingTime>
      </bpsim:TimeParameters>
</bpsim:ElementParameters>
```

XML snippet 13: Constant duration processing time parameter

4.3.4.5 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 elementRef="_93351D59-990B-46B0-80AB-FEEF088E9D7B">
    <bpsim:ControlParameters>
```

```
            <bpsim:Probability>
                <bpsim:FloatingParameter value="0.73"/>
            </bpsim:Probability>
        </bpsim:ControlParameters>
    </bpsim:ElementParameters>
    <bpsim:ElementParameters elementRef="_3D6F2455-BFFB-44BB-A052-892FB15FEEB4">
        <bpsim:ControlParameters>
            <bpsim:Probability>
                <bpsim:FloatingParameter value="0.27"/>
            </bpsim:Probability>
        </bpsim:ControlParameters>
    </bpsim:ElementParameters>
```

XML snippet 14: Probability of following flows from the 'Approved?' decision point

4.3.4.6 Add scenario-level temporal result parameters

In order to request results for the minimum, maximum and mean processing times we add a Result Request on the process element.

```
    <bpsim:ElementParameters elementRef="_6">
    <bpsim:TimeParameters>
          <bpsim:ProcessingTime>
                <bpsim:ResultRequest>min</bpsim:ResultRequest>
                <bpsim:ResultRequest>max</bpsim:ResultRequest>
                <bpsim:ResultRequest>mean</bpsim:ResultRequest>
          </bpsim:ProcessingTime>
    </bpsim:TimeParameters>
    <bpsim:ControlParameters>
          <bpsim:TriggerCount>
                <bpsim:ResultRequest>count</bpsim:ResultRequest>
          </bpsim:TriggerCount>
    </bpsim:ControlParameters>
    </bpsim:ElementParameters>
```

XML snippet 15: Result request for process durations

4.3.4.7 Task-level temporal result parameters

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

```
    <bpsim:ElementParameters elementRef="_2017EC19-4BD5-40D7-9014-E7D337A68E01">
        <bpsim:TimeParameters>
            <bpsim:ProcessingTime>
                <bpsim:ResultRequest>min</bpsim:ResultRequest>
                <bpsim:ResultRequest>max</bpsim:ResultRequest>
                <bpsim:ResultRequest>mean</bpsim:ResultRequest>
                <bpsim:TruncatedNormalDistribution max="1000" mean="20" min="0"
standardDeviation="1"
            </bpsim:ProcessingTime>
        </bpsim:TimeParameters>
    </bpsim:ElementParameters>
```

XML snippet 16: Processing time results for the "Record Loan Application" task

4.3.5 Conclusions and further investigations

The following table provides an indication of where the total time to process a Loan application is spent by ranking them. Specific tools will provide slightly different results so here we just list them in order but you would typically be able to get a percentage of time spent in each activity.

Approximately one third of the total time is spent on Title Search, underwriting takes around a quarter of the time with verifying employment, recording and reviewing the application and reviewing the title report all contributing significantly too. It may be interesting to define further scenarios to explore how cycle time is affected if it is possible hypothesize improved processing times in one or more of these areas.

Perform Title Search
Underwrite Loan with Terms
Verify Employment
Review Loan Application
Record Loan Application
Review Title Report
Close-out Approval
Close-Out Rejection
Receive Loan Application
Loan Request Approved
Loan Request Rejected
1 Hour Time Limit

5 Example 3: Technical support

5.1 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.

5.1.1 Process Description

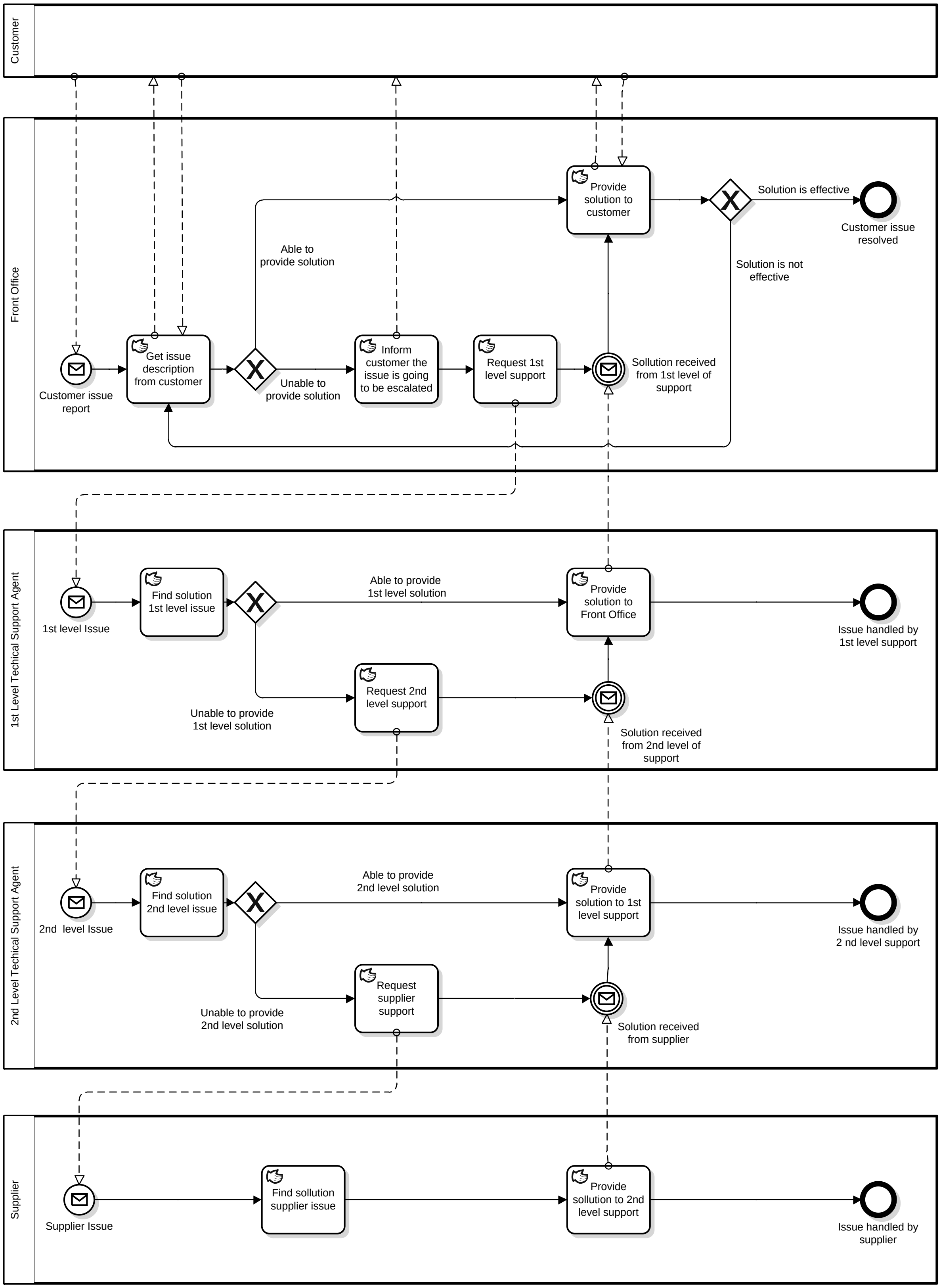
The customer calls the contact center and reports an issue about underperforming service or faulty equipment or software. The Front Office collects information from the Customer and tries to provide a solution directly to the Customer on the other end of the line, otherwise they inform the Customer the 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.

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



5.3 *Simulation scenario 1: Explore control flow perspective*

5.3.1 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.

5.3.2 Goals

The goal of this scenario is to provide a frequency baseline:

- What are the most / less used paths?

5.3.3 Identification of simulation parameters

The simulation inputs are as follows:

5.3.3.1 Simulation parameters

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

5.3.3.2 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: 2.5

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

5.3.3.3 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.

5.3.3.4 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.

5.3.4 How the model provides for that data to be captured

There are no additional constructs needed to run this scenario.

5.3.5 Conclusions and further investigations

The table below shows ranks various activities in order of the number of occurrences. It may be seen that the parts of the organisation nearest the customer occur most often, as common sense would expect. So we may start to gain confidence that the model is valid at the most basic level. Since different simulation vendors should be expected to produce slightly different results based on the statistical seeds used we have not included percentages of instances reaching each activity but this is the kind of information that might be included by those tools.

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

This control perspective scenario may be explored further to see how different levels of issue resolution in each organisation affect the overall workload. This will suggest the relative number of staff needed in each area but before we can draw firm conclusions we should explore the termporal perspective.

[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.

5.4 Simulation scenario 2: Explore temporal perspective

5.4.1 Approach / Hypothesis

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

5.4.2 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

5.4.3 Identification of simulation parameters

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

5.4.3.1 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

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

Activity Description	Mean (min)[2]	Standard Deviation (min)
Provide solution to 2nd level support	2,0	0,5

These tasks' durations are constant.

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

5.4.3.2 Results requested

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

5.4.4 How the model provides for that data to be captured

This scenario does not require any new constructs.

5.4.5 Conclusion and further investigations

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 2nd^{t} 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.

5.5 Simulation scenario 3: Explore resource perspective

5.5.1 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;

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

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 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.

5.5.2 Identification of simulation parameters

This scenario again builds on the parameters of previous scenarios.

5.5.2.1 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

5.5.2.2 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.

As may be seen from the diagram, activities in the Customer pool are not modelled explicitly nor the number of customers specified during the simulation. This is a modelling choice to focus on the activity within the Support organisations. Instead of providing a full model we use control parameters.

5.5.3 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.

5.5.3.1 Define Calendars for use by the scenario

```
      <bpsim:Scenario ... >
            ...
<bpsim:Calendar id="C1" name="shift-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</bpsim:Calendar>

<bpsim:Calendar id="C2" name="shift-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</bpsim:Calendar>
```

```
<bpsim:Calendar id="C3" name="shift-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</bpsim:Calendar>

<bpsim:Calendar id="C4" name="shift-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</bpsim:Calendar>

<bpsim:Calendar id="C5" name="shift-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</bpsim:Calendar>

<bpsim:Calendar id="C6" name="shift-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</bpsim:Calendar>

<bpsim:Calendar id="C7" name="shift-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</bpsim:Calendar>

<bpsim:Calendar id="C8" name="shift-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</bpsim:Calendar>
...
      </bpsim:Scenario>
```

5.5.3.2 Add parameters controlling the resources' availability 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 elementRef="frontOffice">
         <bpsim:ResourceParameters>
            <bpsim:Quantity>
                  <bpsim:NumericParameter value="0"/>
                  <bpsim:NumericParameter value="200" validFor="C1"/>
                  <bpsim:NumericParameter value="90" validFor="C2"/>
                  <bpsim:NumericParameter value="130" validFor="C3"/>
                  <bpsim:NumericParameter value="60" validFor="C4"/>
                  <bpsim:NumericParameter value="150" validFor="C5"/>
                  <bpsim:NumericParameter value="100" validFor="C6"/>
                  <bpsim:NumericParameter value="40" validFor="C7"/>
             </bpsim:Quantity>
          </bpsim:ResourceParameters>
      </bpsim:ElementParameters>
        ...

      </bpsim:Scenario>
```

5.5.3.3 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.

These are represented in the model as follows:

```
          <bpsim:ElementParameters elementRef="frontOffice">
```

```
    ...
    <bpsim:TimeParameters>
      <bpsim:ProcessingTime>
        <bpsim:ResultRequest>sum</bpsim:ResultRequest>
      </bpsim:ProcessingTime>
      <bpsim:WaitTime>
        <bpsim:ResultRequest>sum</bpsim:ResultRequest>
      </bpsim:WaitTime>
    </bpsim:TimeParameters>
    ...
  </bpsim:ElementParameters>
```

5.5.4 Conclusions and further investigations

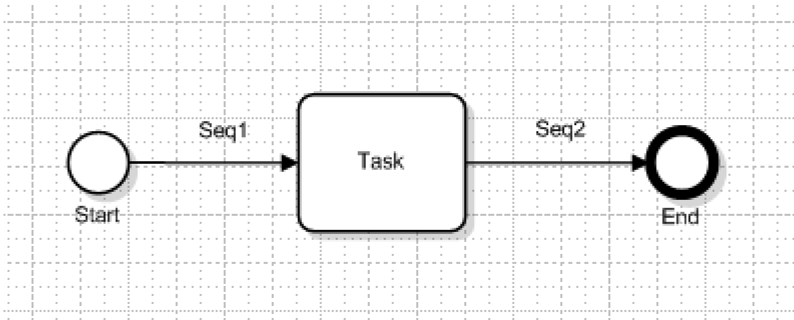
When this simulation scenario is run through a BPSim-compatible simulation tool, we can see that most of the process instances are escalated beyond the Front Office. Therefore it will be necessary to ensure a good balance of staffing between the Front Office and the more technical organisations to ensure a resolution to the customer as quickly as possible.

With the specified resourcing the Front Office can be seen to have by far the largest waiting time, orders of magnitude larger than the other resource pools. So whilst this resourcing ensures it is always possible to report issues, issue resolution is taking longer than it would if we were to move some of the staff into the other parts of the organisation.

Simulation also shows that the Supplier is running at full capacity suggesting that it may be desirable to negotiate more resource from them or to train up the 2nd level Support organisation so that fewer tasks are escalated to the Supplier.

6 Serialization examples

6.1 Time Parameters



6.1.1 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>
```

6.1.2 Lag Time

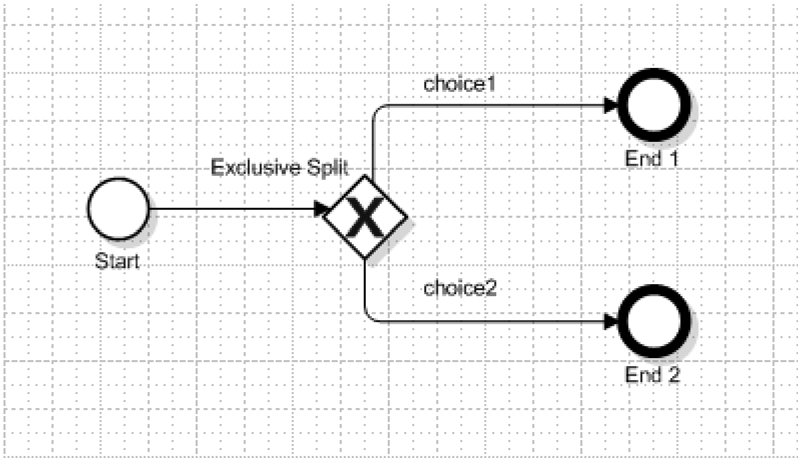
You can set the lag time of ***Seq1*** to 10 seconds using the wait time.

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

6.2 Control Parameters

6.2.1 Routing using Probabilities

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

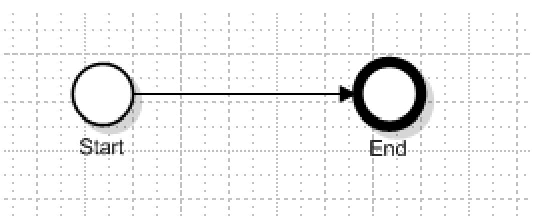


To determine the odds of a split going 25% to ***choice1*** and 75% to ***choice2***, you can use the control parameters.

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

6.2.2 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).

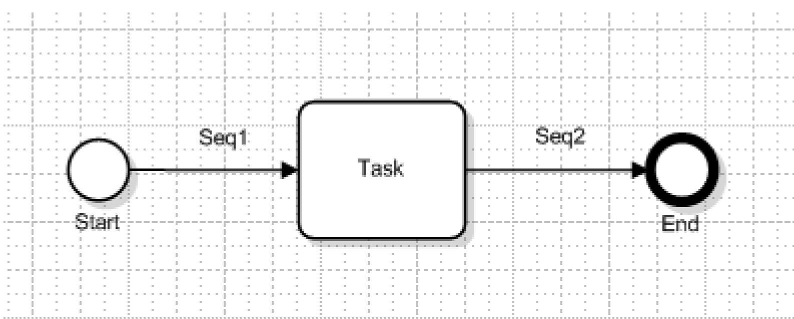
```
<ElementParameters elementRef="start">
      <ControlParameters>
            <TriggerCount>
                  <NumericParameter value="100"/>
            </ TriggerCount >
      </ControlParameters>
</ElementParameters>
```

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>
```

6.3 Using advanced parameterisation

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



6.3.1 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

6.3.2 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>
```

6.3.3 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>
```

6.4 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>
```

6.5 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>
```

6.6 Results

6.6.1 Time Parameters

6.6.1.1 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>
```

6.6.1.2 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>
```

6.6.2 Control Parameters

6.6.2.1 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>
```

6.6.3 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>
```