Document name:	**Use Case description**
Document version:	**1.0**
Task code:	**T5.4**
Deliverable code:	**D5.4.1**
WP Leader (organisation):	**PetalsLink**
Deliverable Leader (organisation):	**PetalsLink**
Authors (organisations):	**Christophe GATTI (THA)**
Date of first version:	**14/05/12**

Document history

Changes	Author / Entity	Code of version
Creation of the document	Christophe GATTI	0.1
Outline added	Christophe GATTI	0.2
‘Draft review” version	Christophe GATTI	1.0

Abstract

The purpose of this deliverable is to describe how the “Air Traffic Flow Management” use-case will be used in the frame of the SocEDA platform simulation.
The business description of the scenario has been presented in the previous deliverable. This description led to the definition of business process models.
Dealing with these models with regard to the simulation platform is exactly what will be described in this document.

Keyword list

Use Case, scenario, requirement, Airport, architectural design, implementation, services, choreographies, demonstration, deployment diagram, BPMN2.0, SOAML, service architecture

Acronyms & Abbreviations

Item	Description
ATC	Air Traffic Control
BPMN	Business Process Modelling Notation
CDM	Collaborative Decision Making
DOW	Description Of Work
DPWS	Devices Profile for Web Services
FI	Future Internet
HTTP	Hypertext Transfer Protocol
MID	Mobile Internet Device
QoS	Quality of Service
TBD	To Be Defined
MID	Mobile Internet Device (e.g. laptops, smartphones)
OMG	Object Management Group
REST	Representational state transfer
SOAML	Service oriented architecture Modeling Language
SOAP	Simple Object Access Protocol
SoTA	State of The Art
WS-*	All types (*) Web Service
WS-I	Web services Interoperability (e.g. “a WS-I profile”)
UML	Unified Modeling Language

Table of Contents

Table of Figures

1. Introduction

1.1. Purpose of the document

This deliverable is about describing how the ATFM use-case will be used in the frame of the SocEDA platform simulation.
The business description of the scenario has been presented in the previous deliverable. This description led to the definition of business process models. Dealing with these models with regard to the simulation platform will mainly be part of this document.

1.2. Document outline

After an introductive section and some restatements concerning the use-case itself, the document is structured according to three main parts:

- The first one briefly recalls the context of the use-case,
- The second one emphasises the connections between the services of the ATFM use-case and the SocEDA platform,
- The third one focuses on the services code projects needed to build the use-case.

2. Scenario – Air Traffic Flow Management

2.1. Overview

The “Air Traffic Flow Management” use case is focused on improving the coordination between mostly **critical** airport actors and services (such as *airline*, *airline ground staff*, *air traffic control*, *Security Company*) in air airplane rerouting context. In this particular case coordination between these services already exists since it is crucial to a correct air traffic management. Still, there is room for improvement through systematic specification and automation in a domain where heterogeneous ad-hoc solutions are often used. We provide choreographies in this perspective.

The solution we provide using SocEDA deals with a challenge that is difficult to tackle with traditional service coordination approaches based on orchestrations: **coordination runtime responsibility**. Indeed, using an orchestration, one need to identify a central control point: the orchestrator. In this highly distributed airport context, where numerous actors are involved, none has the necessary prerogatives to incarnate this central point. Moreover, it would be very risky to have such a single point of failure. The solution to this issue is naturally provided by the choreography paradigm that is distributed by nature: there is no single responsibility point; the responsibility is dispatched to all the participants.

As a solution to this challenge, a three-layered architecture is proposed:

- Simulated Environment: Web services are deployed to simulate the rerouting of planes and the arrival handling of the impacted passengers. The flow of information about the situation is traduced into events and sent to the SocEDA Platform,
- Dynamic of the collaboration: EasyESB (The *Entreprise Service Bus*) in collaboration with a BPEL engine enhance the interactions between actors, *i.e.* the orchestration of the flow management tasks,
- SocEDA platform: The SocEDA platform deals with events and business rules.

2.2. Use-case events elicitation

The ATFM use-case is implemented through Web services. In this context, Web services are considered as Event Sources. They publish events instances respecting WS-Notification specification. Events instances flow through the SocEDA platform. This activity is performed thanks to the SeaCloud component.

Events can either be continual or punctual. For instance, the Weather Forecast service periodically publishes events about the weather forecast and conditions while other services publish only punctual events, to notify the successful end of an activity of the workflow, or the availability of a resource for instance.

For the implementation of the ATFM use-case, seven event types are defined:

- Alert: produced when a problem occurs,
- Demand: produced when an actor asks for resources,
- Offer: produced when an actor answers about a demand,
- Resources Status: event about the status of resources,
- Activity Status: event about the status of activities,
- Instruction: produced when a decision is made,
- Report: event produced at the end of some activities.

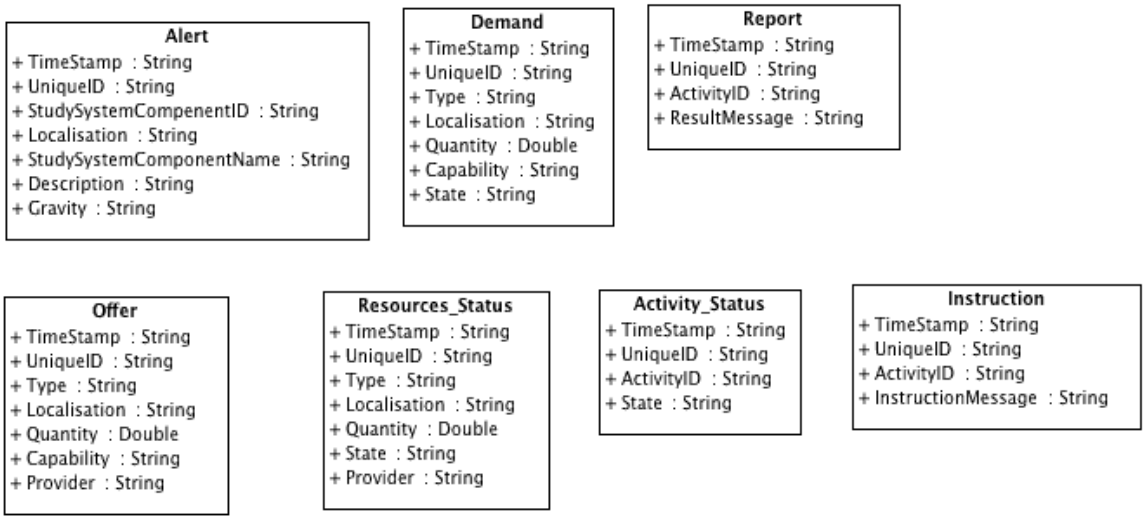

Figure 1: Description of the attributes for each kind of event.

The above table shows an example of mapping between events defined within the use-case and event type defined previously:

Event name	Event type
Running/Done	Activity_status
Accept/Reject Proposal	Report
Propose Reroute	Instruction
Confirm Approach	Report
Send Reroute Warning	Report
Request Amenities	Demand
Provide Available Amenities	Resouces_Status
Book All Available Amenities	Offer

For example, there is an instance of the RequestAmenities event:

```
<?xml version="1.0" encoding="UTF-8"?>
<ns14:Notify xmlns:ns10="http://com.petalslink.easyesb/data/admin/1.0"
xmlns:ns11="http://com.petalslink.esstar/admin/model/datatype/1.0"
xmlns:ns12="http://com.petalslink.easyesb/admin/model/datatype/1.0"
xmlns:ns13="http://www.w3.org/2005/08/addressing" xmlns:ns14="http://docs.oasis-open.org/wsn/b-2"
xmlns:ns15="http://docs.oasis-open.org/wsrf/bf-2" xmlns:ns16="http://docs.oasis-open.org/wsn/t-1"
xmlns:ns17="http://docs.oasis-open.org/wsrf/rp-2"
xmlns:ns18="http://com.petalslink.esstar/data/management/user/1.0"
xmlns:ns19="http://com.petalslink.esstar/data/management/admin/1.0"
xmlns:ns2="http://com.ebmwebsourcing.easyesb/soa/model/endpoint"
xmlns:ns20="http://com.petalslink.easyesb/component/bpel/data/1.0"
xmlns:ns3="http://com.petalslink.easyesb/soa/model/datatype/1.0"
xmlns:ns4="http://com.ebmwebsourcing.easyesb/soa/model/service"
xmlns:ns5="http://com.ebmwebsourcing.easyesb/soa/model/component"
xmlns:ns6="http://com.ebmwebsourcing.easyesb/soa/model/node"
xmlns:ns7="http://com.ebmwebsourcing.easyesb/soa/model/registry"
xmlns:ns8="http://com.petalslink.easyesb/exchange/1.0"
xmlns:ns9="http://com.petalslink.easyesb/transporter/1.0">
<ns14:NotificationMessage>
<ns14:SubscriptionReference>
<ns13:Address>http://localhost:9803/AirlineGroundStaffMIDEventProducersSOAPEndpoint</ns13:Address>
```

```
<ns13:ReferenceParameters/>
</ns14:SubscriptionReference>
<ns14:Topic Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">top:resourcesEvent</ns14:Topic>
<ns14:ProducerReference>
<ns13:Address>http://localhost:9300/AirlineGroundStaffMIDSOAPEndpoint</ns13:Address>
<ns13:ReferenceParameters/>
</ns14:ProducerReference>
<ns14:Message>
<ns2:demandEvent xmlns:ns2="http://www.thalesgroup.com/atfmevent"
xmlns:ns3="http://www.petalslink.org/AirlineGroundStaffMID/"
xmlns:ns4="http://www.petalslink.org/SimulatedEventProducers/" xmlns:ns5="http://docs.oasis-
open.org/wsrf/rp-2" xmlns:ns6="http://docs.oasis-open.org/wsrf/bf-2"
xmlns:ns7="http://www.w3.org/2005/08/addressing" xmlns:ns8="http://docs.oasis-open.org/wsn/b-2"
xmlns:ns9="http://docs.oasis-open.org/wsn/t-1" xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2">
<ns2:uid>DemandEvent-0</ns2:uid>
<ns2:timestamp>2012-02-23T09:46:24.399+01:00</ns2:timestamp>
<ns2:capability/>
<ns2:state/>
<ns2:uncertainty>
<ns2:unit/>
<ns2:value/>
<ns2:origin/>
</ns2:uncertainty>
<ns2:resources>
<ns2:type>car</ns2:type>
<ns2:localisation/>
<ns2:quantity>10.0</ns2:quantity>
</ns2:resources>
</ns2:demandEvent>
</ns14:Message>
</ns14:NotificationMessage>
<ebm:emissionDate xmlns:ebm="http://wwww.petalslink.com/wsoui/wsnotification"
xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"/>
</ns14:Notify>
```

Figure 2: Example of RequestAmenities event in WS-Notification.

3. Simulation Environment

3.1. Global Description

The simulation environment is implemented through Web services deployed on several ESB (Entreprise Service Bus). The interactions between operations of Web services are executed thanks to workflow written in BPEL (Business Process Execution Language) and executed through a BPEL engine.

The BPEL engine interprets event in WS-Notification format. Therefore, our simulation environment is divided into two sub-environments:

- The use-case environment: each event is written in WS-Notification format. This sub-environment is composed of the Web service of the use-case and the BPEL executed by the BPEL engine of an ESB,
- The platform environment that represents the connection between the SocEDA platform and the SeaCloud.

The SeaCloud makes the link between these two environments.

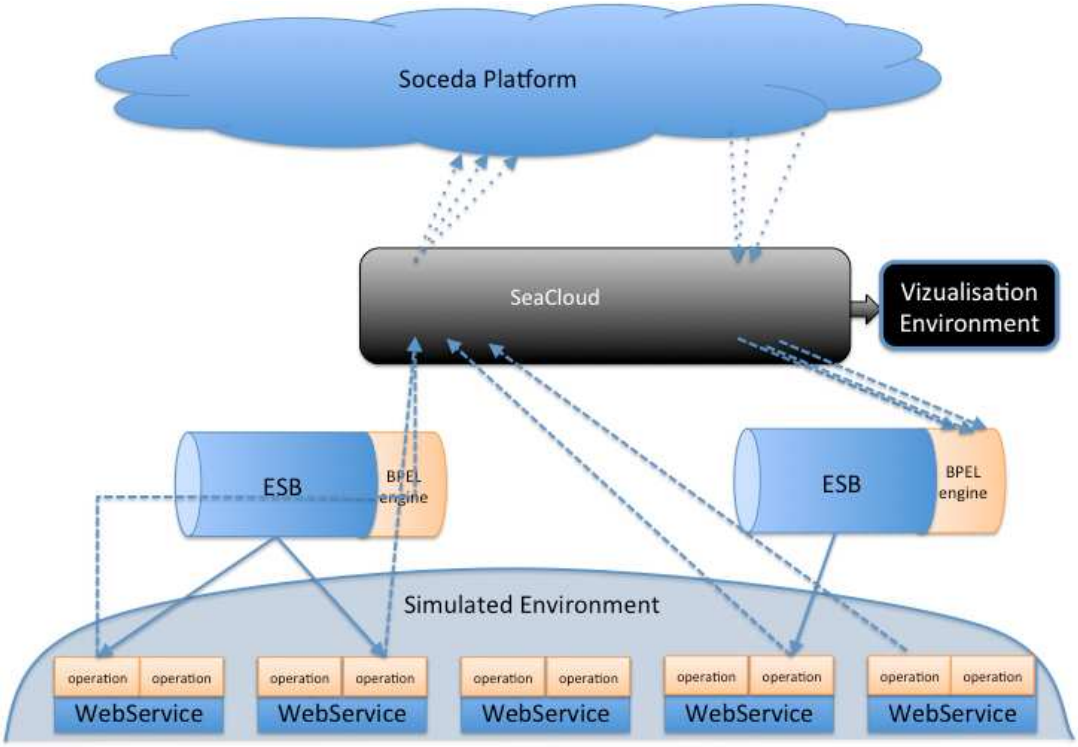

Figure 3: Overview of the simulation platform

3.2. Events schemas

The seven kinds of events of the ATFM use-case are specified in the table below:

Event Topic	Event Type
SituationalTopic	AlertEvent
ActivityTopic	ActivityStatusEvent
ResourcesTopic	DemandEvent
	OfferEvent
	ResourcesStatusEvent
ConsequenceTopic	InstructionEvent
	ReportEvent

3.3. Implementation of Web services

Our simulated environment is based on 52-55 workflows (it depends on services instances number). These workflows could be merged into 3 processes (Rerouting, Manage Unexpected Arriva, and Arrival Handling). Indeed, because of scalability aspects, lots of processes are realized by several actors at the same time.

In the whole of the simulated environment, a lot of Web services operations are called. The following table summarizes the number of operations and Web services by actor:

Actors	Number of Web services	Number of operations
Weather Forecast Service	1	2
ATC	1	7
Airplane	1	3
Central Flow Management Unit	1	4
Airport Bus Company	2	8
Ground Transportation Company	2	4
Luggage Handling Company	2	8
Security Company	2	8
Airline Ground Staff MID	1	14
Airline	1	8
Travel Agency	1	8
Stand and Gate Management	1	8
Airport	1	15
Bookable Amenity	3	4
Hotel	3	5
Total	**23**	**106**

3.4. CEP rules

The following CEP-rules are expressed in natural language and refer to the generation of complex events from technical events. This process is done by the CEP. The rules are written using the CEP graphical tool editor.

1. IF t = *Activity is running*.time + A.length x 1.1 AND Not(*Activity is done*) □ *Alert-Delay*

If one activity takes more than 110% of expected time, then send a DELAY alert

2. IF t = *Activity is running*.time + A.length x 2 AND Not(*Activity is done*) □ *Alert-Warning*

If one activity takes more than 200% of expected time, then send a WARNING alert

This list is not exhaustive and will be completed.

4. Conclusion

This documents permits to consider in a more concrete way the problematic of integration of the use-case with the SocEDA platform. The development of the use-case through the use of Web services has reached a mature point. The next point to consider is to enlarge the spectre of CEP rules, express ELA inside the services description, implement BPEL process files in order to enrich the platform with more and more workflows, and test the whole use-case in the target environment, in a distributed way to get convincing performance results.