Grant Agreement N° 215483

Title: *Initial Definition of Validation Scenarios*

Authors: *UniDue, Tilburg, CITY, FBK, Polimi, SZTAKI, TUW, UPM, USTUTT, UniHH*

Editor: *Raman Kazhamiakin (FBK)*

Reviewers: *Ita Richardson (Lero-UL)*

Manuel Carro (UPM)

Identifier: *PO-IA-3.2.1*

Type: *Deliverable*

Version: *1*

Date: *05 October 2009*

Status: *Final*

Class: *External*

Management Summary

This deliverable aims to collect and to systematically document the scenarios existing in S-Cube materials. These scenarios will be used in the follow up deliverable CD-IA-3.2.2 to validate the research framework and to support the validation of the individual research results. To approach this goal, the following three elements are represented in this document: First, the systematic guidelines are defined in order to describe the scenarios in a uniform way and to link them with the industrial case-studies provided in IA-2.2. Second, the existing scenarios produced by the S-Cube partners are collected and documented according to the guidelines. Third, based on this initial set of scenarios, we demonstrate with the help of examples how the scenarios should be used to validate S-Cube results.

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement n° 215483 (S-Cube).

Members of the S-Cube consortium:

University of Duisburg-Essen (Coordinator)	Germany
Tilburg University	Netherlands
City University London	U.K.
Consiglio Nazionale delle Ricerche	Italy
Center for Scientific and Technological Research	Italy
The French National Institute for Research in Computer Science and Control	France
Lero - The Irish Software Engineering Research Centre	Ireland
Politecnico di Milano	Italy
MTA SZTAKI – Computer and Automation Research Institute	Hungary
Vienna University of Technology	Austria
Université Claude Bernard Lyon	France
University of Crete	Greece
Universidad Politécnica de Madrid	Spain
University of Stuttgart	Germany
University of Hamburg	Germany
Vrije Universiteit Amsterdam	Netherlands

Published S-Cube documents

All public S-Cube deliverables are available from the S-Cube Web Portal at the following URL:

http://www.s-cube-network.eu/results/deliverables/

The S-Cube Deliverable Series

Vision and Objectives of S-Cube

The Software Services and Systems Network (S-Cube) will establish a unified, multidisciplinary, vibrant research community which will enable Europe to lead the software-services revolution, helping shape the software-service based Internet which is the backbone of our future interactive society.

By integrating diverse research communities, S-Cube intends to achieve world-wide scientific excellence in a field that is critical for European competitiveness. S-Cube will accomplish its aims by meeting the following objectives:

- Re-aligning, re-shaping and integrating research agendas of key European players from diverse research areas and by synthesizing and integrating diversified knowledge, thereby establishing a long-lasting foundation for steering research and for achieving innovation at the highest level.
- Inaugurating a Europe-wide common program of education and training for researchers and industry thereby creating a common culture that will have a profound impact on the future of the field.
- Establishing a pro-active mobility plan to enable cross-fertilisation and thereby fostering the integration of research communities and the establishment of a common software services research culture.
- Establishing trust relationships with industry via European Technology Platforms (specifically NESSI) to achieve a catalytic effect in shaping European research, strengthening industrial competitiveness and addressing main societal challenges.
- Defining a broader research vision and perspective that will shape the software-service based Internet of the future and will accelerate economic growth and improve the living conditions of European citizens.

S-Cube will produce an integrated research community of international reputation and acclaim that will help define the future shape of the field of software services which is of critical for European competitiveness. S-Cube will provide service engineering methodologies which facilitate the development, deployment and adjustment of sophisticated hybrid service-based systems that cannot be addressed with today's limited software engineering approaches. S-Cube will further introduce an advanced training program for researchers and practitioners. Finally, S-Cube intends to bring strategic added value to European industry by using industry best-practice models and by implementing research results into pilot business cases and prototype systems.

S-Cube materials are available from URL: http://www.s-cube-network.eu/

Table of Contents

List of Figures

List of Tables

List of acronyms

ASD – Automatic Service Deployment
ASN – Agile Service Network
BPEL – Business Process Execution Language
BPM – Business Process Management
BPMN – Business Process Modelling Notation
JSDL – Job Submission Description Language
KPI – Key Performance Indicator
MFSS – Maxillo-Facial Surgery Simulation
PPM – Process Performance Metric
REST – Representational State Transfer
RFID – Radio Frequency Identification
QoS – Quality of Service
SAM – Service Adaptation and Monitoring
SBA – Service-based Application
SCC – Service Composition and Coordination
SLA – Service-Level Agreement
SMGB – Self-monitoring of Blood Glucose
SOA – Service-Oriented Architecture
SOAP – Simple Object Access Protocol
SOC – Service-Oriented Computing
SCOR – Supply Chain Operational Reference
SRV – SLA-based Resource Virtualization
UML –Unified Modelling Language
UMTS – Universal Mobile Telecommunications System
WCF – Windows Communication Foundation
WSDL – Web Service Definition Language
XML – Extensible Mark-up Language

1 Introduction

With respect to validation of the goals of work-package WP-IA-3.2 (“Integration Framework: Validation and Personalization”) is the validation of the S-Cube integrated framework [10] through suitable industrial case studies and the support for the validation of the individual research results produced by the joint research activities. This deliverable reports the outcomes of the initial activities of this work-package towards these goals. In particular, it covers the definition of the methodology and of a set of guidelines for documenting scenarios; a collection of an initial set of validation scenarios based on existing S-Cube materials and an example on how the scenarios should be used to validate research results.

The collection of the validation scenarios – and, more in general, the validation of the integrated framework – is an activity that will occur through the full life of the project. It will see the participation of all the S-Cube partners as part of the research activities undertaken in JRA-1 and JRA-2. Moreover, the validation activities are strongly integrated with work-package IA-2.2, which is responsible for defining the industrial case studies to be used in the validation activities, and with work-package IA-3.1, which is responsible for refining and revising the integrated framework according to the outcomes of the validation activities. For these reasons, it has been important to define a shared methodology and a set of guidelines to be adopted in these activities. These guidelines, described in Section 2, cover the aspects related to the identification of the validation scenarios, to the definition of the validation plans, and to the modalities for reporting the results of the validation.

In addition to the guidelines, the deliverable contains the initial set of validation scenarios contributed by the S-Cube partners (Section 3). This initial set of scenarios is hence not exhaustive since, as we already mentioned, the collection of scenarios is a continuous activity that will be undertaken throughout the entire life of the project. The collected scenarios serve mainly three goals:

1) First, they have been used to test and refine the validation guidelines provided in this deliverable and to provide concrete examples of their application.
2) Second, they will serve as an input for the actual validation of the integration framework, which is the object of Deliverable CD-IA-3.2.2 (Month 24).
3) Finally, the collected set of scenarios allows for a first analysis of coverage of the different components of the integration framework. This analysis will identify gaps or overlaps in the collected scenarios and will help drive the collection of new scenarios. This aspect is discussed in more detail in Deliverable PO-IA-2.2.3.

This deliverable also contains an example of the application of one scenario to an individual research result. This example demonstrates how the scenarios will be used in the joint research activities (Section 4). These reported examples are useful to illustrate the validation methodology and guidelines, and will serve as a reference for future validation activities.

The deliverable is completed by some concluding remarks and by a discussion of the future activities (Section 5).

2 Validation Scenario Writing Guidelines

This chapter describes how case studies and scenarios are used in S-Cube, especially in IA-2.2, IA-3.1 and IA-3.2. The rational for this section is to develop a synergic approach by producing and using case studies and scenarios in the S-Cube workpackages. In this light, the description of work uses the term "case study" in the description of IA-2.2 and "scenario" in the description of IA-3.2. To develop a clear understanding of these terms we use the following definitions:

- *Case Studies* (synonyms in DoW: pilot case studies, use cases): Case studies are real life problems contextualized within the corresponding application domain. A case study description introduces the relevant vocabulary of the domain (glossary), the business goals and domain assumptions on which the considered problem is based, the actors involved in the scenario, their relationships and dependencies, and a number of coarse-grained scenarios or situations that show how the actors interact in order to fulfil the business goals, given the domain assumptions. In [7] examples of case study descriptions are presented. Of course a case study can be addressed by specific solutions, which are strongly related and dependent on the case study descriptions. In this deliverable, however, we aim at developing case study descriptions that are completely independent from specific technological solutions so that the case study can be flexibly used to identify, develop and evaluate different technological solutions.
- *Validation Scenarios*: Validation scenarios are always related to case studies and describe possible ways the actors in a case study behave. Typically this behaviour is defined by sequences of activities within this case study. In this way the case study and possibly one or more of its coarse-grained scenarios is refined. An exemplary "Distribution and Sales" grained validation scenario is described in [7]: *During the sales phase the Wine Producer interacts with the Retailer to place orders. The orders are delivered by the Delivery Company. The wine producer can monitor the wine conditions, such as the temperature, during and after its production. It is possible, for instance, to monitor whether the temperature of the wine remained within pre-defined limits.*

This deliverable deals exclusively with scenarios, which are intended to be used to validate individual research results and the integration framework. The scenarios collected in this deliverable are derived from S-Cube partner's current and/or future research project, e. g. on the basis of existing papers or existing paper drafts. Each partner was then asked to document the scenarios according to a unique format (scenario writing guidelines). These scenario writing guidelines are described in detail in Subsection 2.1. The application of the guidelines is demonstrated in Subsection 2.2 with the help of an example. The guidelines also ensure that the scenarios are well aligned to the industrial case studies. This aspect is covered in Subsection 2.3 of this document.

2.1 Scenario Writing Guidelines

Scenarios should be used to validate the usefulness of our S-Cube artefacts such as the individual research results produced in the JRAs and the integration framework. This section discusses how those scenarios should be documented and linked to the industrial case studies introduced in the IA-2.2 workpackage.

Generally each scenario IA-3.2 scenario description should provide the following three classes of information:

1) *Introduction*: An introduction of the scenario as non-structured English text. The introduction should explain the context of the scenario, which may be accompanied by illustrations if necessary.
2) *Scenario Documentation Template*: The second section should contain the completed scenario documentation template. This template provides structured information to the scenario so that

it can be easily linked to the IA-2.2 case studies and to the IA-3.1 integration framework elements. This scenario is described in detail in Subsection 2.1.1.

3) *Structured Scenario Description*: The structured scenario description contains the scenario steps and their control flow. Two formats may be used for this description: structured English text or UML activity or sequence diagrams (cf. Subsection 2.1.2).

2.1.1 Scenario Documentation Template

Scenarios are used in a variety of ways in the domains of Human Computer Interaction, Requirements Engineering and Information Systems. A useful classification of those scenarios usages can be found in [1]. Since we only want to use the scenarios for the validation of research results and the validation of the integration framework, we concentrate on distinguishing the scenarios based on the following properties [2]:

- *Scenario Type*: Each SBA depends on an environment or context. Scenarios are mostly used to describe an interaction between the environment (e.g. users, other IT systems) and the SBA (*interaction scenarios*). However, scenarios may also be used to describe activities within the SBA (e. g. particular workflows) or activities in the environment (e. g. the interaction of two users while using a SBA). In the former case we speak about *internal scenarios* and in the latter case about *environmental scenarios*. The distinction of the three scenario types is relevant since they can only be used in conjunction with specific validation types (research methods). While an internal scenario can be most effectively evaluated on the basis of experiments, the interaction of a user and the SBA might require an empirical setup. Consequently, the distinction of the different scenario types helps us to understand whether the correct validation type (research method) was chosen for the scenario at hand.
- *Abstraction Level*: Under this category we distinguish between instance, type and mixed scenarios. *Instance scenarios* describe one concrete execution of the service-based system and contain specific names for actors, events and messages. In contrast to instance scenarios, *type scenarios* do not contain concrete actors, events and messages but actor types (e. g. customer, manager), event and message types (e. g. in form of a method description). Instance scenarios are also called concrete scenarios while type scenarios are called abstract scenarios. If a scenario contains both type and instance information we call it a *mixed scenario*. Since we usually validate a concrete case, we will most likely use instance scenarios.
- *Scenario Usage*: In this category we describe whether the scenario models a wanted or unwanted interaction sequence. A *positive scenario* (synonym: regular scenario) describes a wanted interaction sequence while a *negative scenario* (synonym: misuse case) describes an unwanted interaction sequence [3].

Based on the different kinds of scenarios discussed before, we propose the following template to be used to document validation scenarios (cf. light gray elements in Table 1). Besides the usual scenario documentation elements such as ID, name, authors and source and the before-mentioned scenario type, abstraction level and scenario usage elements, we include a revision element in the template.

Since scenarios may be modified during their lifetime and since they are linked to other S-Cube results, such as the IA-2.2 case studies, it is important to know the version of the scenario participating in this link. Therefore, we also introduce a revision element. The revision is a number in the format xx.yy, where xx denotes a major revision and yy denotes a minor revision. Every scenario contained in this deliverable has the initial revision 1.0. Every update of the scenario must increment either the minor or major revision number.

Table 1: Scenario Documentation Template

Scenario Description	ID	Unique identifier of the scenario.
	Revision	Current Revision of the scenario.
	Name	Name of the scenario.
	Description	Brief summary of the scenario.
	Authors	Authors of the scenario.
	Source	Link(s) – preferably to bibadmin – to the paper(s), in which this scenario was used.
	Scenario Type	Context, interaction, internal scenario.
	Abstraction level	Instance, type or mixed scenario.
	Scenario usage	Positive vs. negative scenario.
	Scenario Steps	Link to the description of the interaction (course) of the scenario including messages and events. This interaction can be described verbally (structured English) or in form of UML Sequence or Activity Diagrams.
Relation to Case Study	Case Study	Case study in IA-2.2 (Name and ID and Revision of the Case Study).
	Goal	Goal(s) of the IA-2.2 case study related to this scenario.
	Actors	Actors in the IA-2.2 case study relevant for this scenario.
	Additional Materials	Link to the activity diagrams and/or use case diagrams contained in the IA-2.2 case studies. The scenario may for instance refine a use case or an activity.

To link the scenario to the IA-2.2 case studies we refer to the case study templates introduced in the deliverable CD-IA-2.2.2 [7] (cf. dark gray elements in Table 1). The different parts of each case study are again documented by templates. The elements of those templates are: Field, Unique ID, Short Name, Type, Description, Rationale, Involved Stakeholders, Conflicts, Supporting Materials, Priority of accomplishments. Given this information, linking the scenario to the different parts of the case study does not only ensure the internal consistency of the scenarios and case studies but also adds additional information such as priorities to the scenarios.

Each scenario must refine one case study (cf. subsection 2.3.1). Therefore, the scenario references a case study. Each scenario should be related to one or more goal it fulfils (positive scenario) or not (negative scenario). The line "goal" should be used to reflect this link. In addition, each scenario interaction sequence is executed by a number of actors. Ideally, the actors used in the scenario should correspond to those actors in the case studies (actors' row in the template). Finally, some parts of the case studies in CD-IA-2.2.2 contain activity diagrams and/or use case diagrams as additional materials. If the scenario refines a use case or one or more activities, this link should be made explicit in the additional materials row of the template.

2.1.2 Structural Scenario Description

For the documentation of the scenario steps, we allow three types of notations: a textual notation using structured English, Sequence Diagrams of the UML (for instance and mixed scenarios) and Activity Diagrams for type scenarios. UML diagrams are the preferred way of documenting scenarios. Each diagram must respect the UML notation [5]. To allow an exchange of those UML diagrams within the project, it is recommended to use StarUML [6] as tool for drawing the diagrams.

In case a textual notation is used, the scenario steps should be documented in structured English organised in a table. Each column of this table contains a scenario course while the rows contain the scenario steps. The left column contains the so called normal course. It is the sequence of steps in case no error and no exception occur in the scenario. The most likely alternative course scenario is documented in the next column to the right. It starts in the row where the error and/or the exception are handled. The first entry in this scenario is the condition under which the alternative course is executed. The scenario steps of this alternative course are documented afterwards. At the end this alternative course indicates where it joins the normal course again. More alternative courses may be added as new columns where necessary. If alternatives in those alternative courses are considered, they should also be added as new column to the right.

Each scenario step is numbered. The normal course scenario uses Arabic numbers starting with 1). The first level alternative courses use three numbers separated by a dot such as a.b.c. a indicates the step of the normal course where the alternative course forks; b indicates the number of the alternative course and c the step in this alternative course. Abstract examples are provided below (cf. Table 2 and Figure 1).

Table 2: Abstract Template for Structured English Description of a Scenario

Normal Course	Alternative Course 1 (1st level)	Alternative Course 2 (1st level)	Alternative Course 3 (2nd level)
1) Step A			
2) Step B	[cond. 1.1] 2.1.1) Step E	[cond. 1.2] 2.2.1) Step H	[cond. 2.1] 2.2.1.1.1) Step J
			2.2.1.1.2) Step K
			2.2.1.1.2) Step L
			➔ Continue with 2.2.2)
	2.1.2) Step F	2.2.2) Step I	
	2.2.3) Step G		
	➔ Continue with 3)	➔ End of scenario	
3) Step C			
4) Step D			

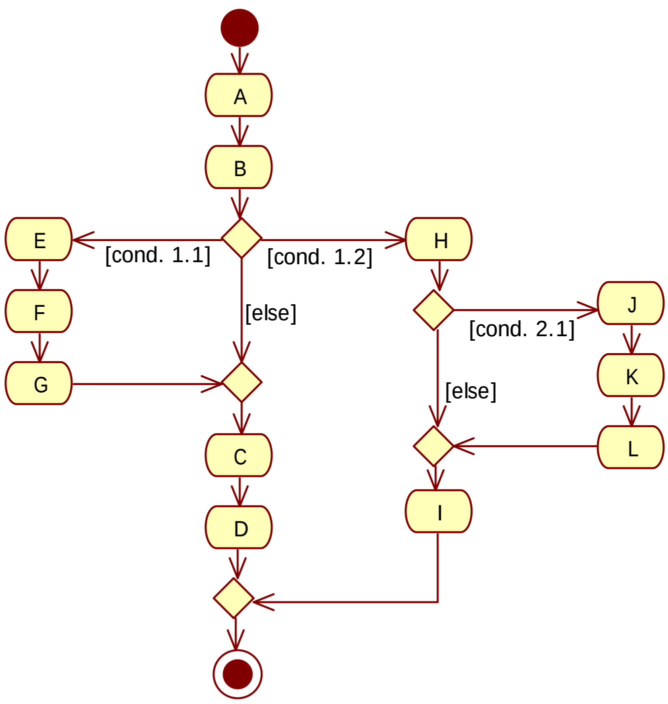

Figure 1: Activity Diagram of Abstract Scenario in Table 2

2.2 Example of a Documented Scenario

2.2.1 Introduction

The aim of this section is to provide a concrete example of a documented scenario to illustrate our approach. The scenario is based on the wine case study documented in [7] and is used in a recent workshop paper [8]. The control flow of this scenario is depicted in Figure 2.

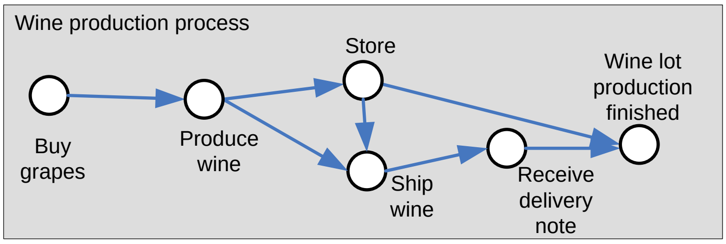

Figure 2: Wine Production and Shipment Process

The process starts with buying wine grapes (*Buy Grapes*). After this activity, the production (*Produce wine*) is carried out. Afterwards the scenario might proceed in two different ways. The first way is storing the wine in oak barrels to mature (*Store*), which may be followed by the shipment activity (*Ship wine*) in case enough high quality wine was produced or by finishing the production activity if the amount of wine produced is not sufficient (*Wine lot production finished*). The alternative course is to ship the wine immediately (*Ship wine*) followed by receiving a delivery note (*Receive delivery note*) and finished by the activity *Wine lot production finished*. This alternative is only triggered in case the quality of the grapes is not high enough to produce high quality wines.

2.2.2 Scenario Documentation Template

The resulting scenario template is depicted in Table 3.

Table 3: Scenario Template for the Wine Example

ID	Wine-Example-1
Revision	1.0
Name	Wine Production & Delivery.
Description	Describes how the wine is produced and delivered to the customer.
Authors	Andreas Gehlert, Julia Hielscher, Olha Danylevych, Dimka Karastoyanova
Source	http://bibadmin.s-cube-network.eu/show.php?id=123
Scenario Type	Internal scenario
Abstraction level	Type Scenario
Scenario usage	Positive Scenario
Scenario Steps	See Table 4 and Figure 3
Case Study	Wine case study; WINERY-S-DS, WINERY-S-HFM
Goal	Stipulate Contracts.
Actors	Wine Producer, Retailer, Delivery Company
Additional Materials	Refines “Processing of the Grapes” activity in WINERY-S-HFM and “Deliver Order” activity in WINERY-S-HFM.

2.2.3 Structural Scenario Description

The textual description using structured English is contained in Table 4. The resulting model is depicted in Figure 3.

Table 4: Structured English Description of the Scenario

Normal Course	Alternative Course 1
1) Buy Grapes	
2) Produce Wine	[wine quality high] 2.1.1) Store wine ➔ continue with step 3) in case of sufficient wine quantity; end scenario otherwise.
3) Ship Wine	
4) Receive Delivery Note	

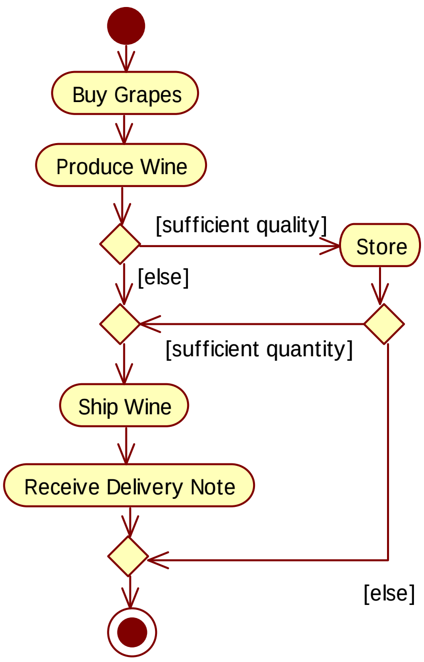

Figure 3: Activity Diagram of the Scenario

2.3 Using Case Studies and Scenarios within S-Cube

Case studies and scenarios should be used in the project with the purposes of validating research results and of identifying new research challenges and gaps. For the moment we focus on the validation aspect as this is the most transversal one with respect to the S-Cube workpackage structure.

To document validation scenarios we propose the following process (see also Figure 4):

1) *Choose a validation scenario*. Assume a scenario is needed to validate S-Cube research results. In this case, choose one of the scenarios collected by IA-3.2. If there is no suitable scenario for the specific validation purpose, it is possible to define a new one by refining an IA-2.2 case study. If an appropriate case study cannot be found, it is also possible considering a new case study to be included in IA-2.2.
2) *Define a validation plan*. The validation plan should include the validation goal, the technique, approach or method under evaluation and its relation to the integration framework, the organisational setting as well as the validation type (research method) used for the validation.

The validation goal should be clearly related to the S-Cube vision and goals [9]. The technique, approach or method under evaluation must be related to the integration framework (IA-3.1). This relation allows us to analyse which parts of the integration framework are already evaluated and which parts need additional evaluations. The organisational setting describes the environment in which the validation is performed. Organisational setting is understood in its broadest sense and covers an industrial setup as well as a computational setup in a laboratory. The validation type (research method) describes the necessary techniques and tools to carry out the validation. This may include:

- empirical research methods (such as laboratory experiments, case studies, field studies etc.), e. g. to validate the usage of systems, techniques or methods by users;
- prototyping, e. g. building a mock-up to demonstrate the feasibility of an approach;
- technical experiments, e. g. to demonstrate the effectiveness of an algorithm;
- conceptual evaluation, e. g. to prove claims based on theories or mathematical reasoning.

3) *Report your validation results*. Once the scenario is written and the validation is done, report your scenario to IA-3.1, IA-3.2 and IA-2.2 together with the validation results.
4) *Packaging the results*. IA-3.2 will package the validation aspects and will provide them on the web portal. In parallel IA-2.2 will coherently align the new scenarios to the existing case studies and will extend and/or modify the case studies where necessary. In addition, IA-2.2 will align the validated approaches, tools, techniques or methods (solutions) to the case studies. Finally, IA-3.1 will ensure that the obtained results are in line with the S-Cube research vision and roadmap.

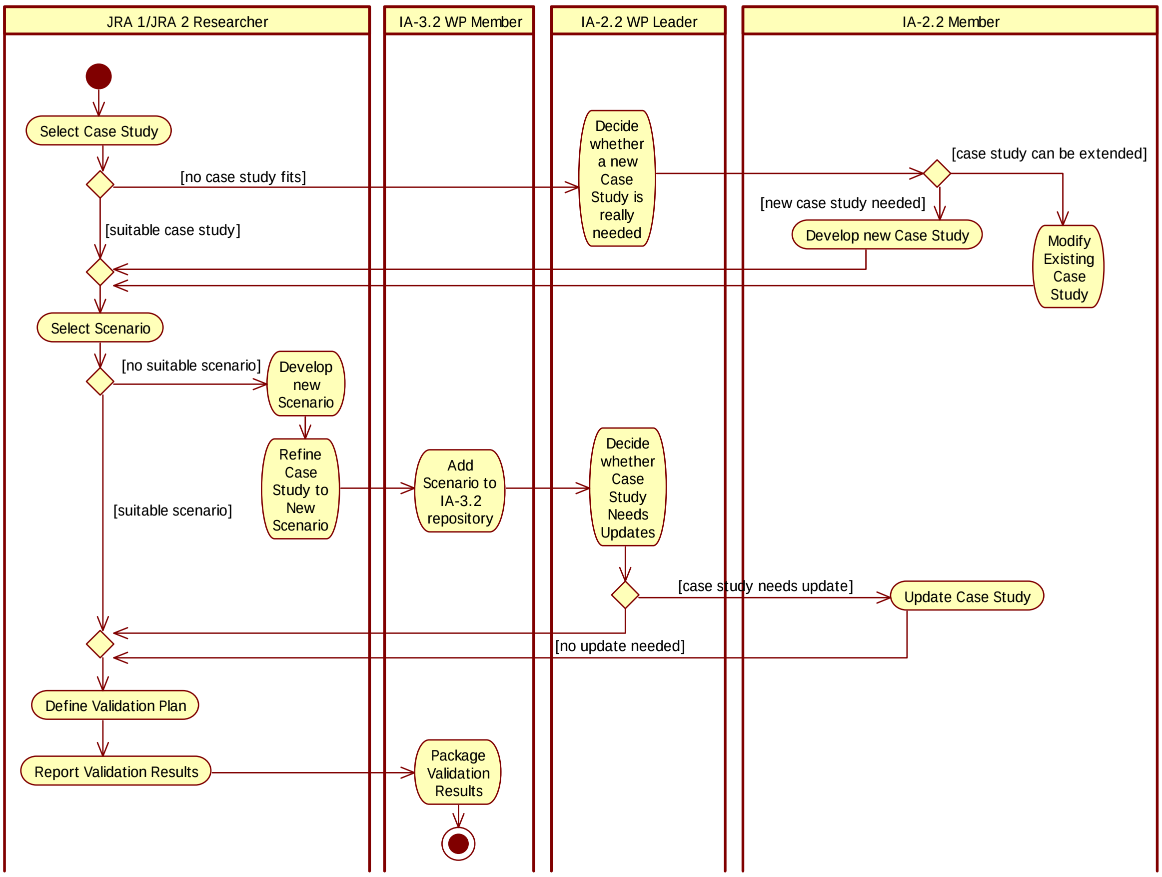

Figure 4: Process of Using and Developing Scenarios

New scenarios and case studies can be introduced if the existing ones are not suitable for the purpose of validation. Guidelines for introducing further S-Cube case studies and scenarios are:

- If you do not find a scenario previously defined within S-Cube that is suitable for your purpose, you can define a new scenario by refining an IA-2.2 case study and relate it to this case study later on. In this case notify the IA-2.2 WP leader so that they can check whether the new scenario has an influence on existing case studies. If the intended scenario is not covered by IA-2.2 case studies, contact the IA-2.2 WP leader to discuss the inclusion of a new industrial case study.
- Case studies were introduced in deliverable IA-2.2.2 for the Automotive, E-Health, E-Government, Wine and Traffic Management domains [7]. Although it is still possible to add case studies to S-Cube, a new case study should be very well motivated (e.g., in terms of unique features not covered by the current scenarios) and it must be confirmed by the workpackage leader of IA-2.2 (POLIMI).Once created, the new case study is to be shared with the project and become part of IA-2.2.
- While case studies are the stable element in S-Cube, scenarios can always be added as needed. However, each scenario must be clearly related to a case study. Assume, for instance, that there is the need to add a new scenario to describe the usage of a navigation system, which is currently not included in IA-2.2. In this case, the scenario could be related to the transportation use case and be tailored to be used in traffic management. Introducing new scenarios (as part of IA-3.2) could require that new business goals or domain assumptions should be added to the description of the related case study. Consequently, upon adding a new validation scenario in IA-3.2 the leader of IA-2.2. (POLIMI) should be notified to check whether the new scenario has an influence on existing case studies. This ensures that there is a complete overview of the scenarios used in the project and that the case studies remain consistent with the scenarios.
- To ensure the integration of our results, all work reported to IA-3.2 and IA-2.2 must be based on scenarios which fulfil these conditions.
- In turn, all scenarios and case studies will be made available through the project website. A structure for this website will be set up as part of IA-3.2 and will be used by both IA-2.2 and IA-3.2 workpackages.

2.3.1 Case Studies and Scenarios in S-Cube Deliverables

The use cases and scenarios are intended to be used as follows in the deliverables:

- IA-2.2.2, delivered at month 12, contains a description of five case studies, each one structured in high level scenarios. These scenarios should be extended as needed and new scenarios might be added according to what was discussed in Section 1.2.
- IA-3.2.1, due in month 19, will document the scenarios contributed by all partners as well as sample evaluation plans for these scenarios.
- IA-2.2.4, due in month 21, will harmonize and aggregate the scenarios collected in IA-3.2.1 and relate them to each of main case studies.
- IA-3.1.3, due in month 21, will refine the integration framework baseline and, based on the evaluation plans that will be received, will identify the aspects of the integration framework that will be subject of evaluation in this first round.
- IA-3.2.2, due in month 24, will focus on the validation of the integration framework through the usage of the scenarios defined in IA-3.2.1.

2.3.2 Relations between Research WPs and Integration WPs

The research work carried out in JRA 1 & 2 is strictly interlinked to these deliverables. In particular, JRA 1 & 2 will take the case studies defined in IA-2.2.2, refine its scenarios and use those scenarios to validate workpackage specific research results such as approaches, techniques and methods.

The research WPs provide input for the integration WPs in three forms:

- The scenarios used for the validation are fed back to IA-2.2 in order to be harmonized and aggregated.
- The scenarios and the corresponding validation plans are provided to IA-3.2 for re-use by other partners.
- The validation results are reported to IA-3.2's new task on empirical evaluation, which was created as a replacement of the closed IA-1.2 (EDSL) workpackage.

In addition, the JRAs should also ensure that the research is in line with the research framework, defined in IA-3.1, e.g., by guaranteeing that the investigated methods and techniques are consistent with the building blocks of the research framework and contribute to its refinement. Moreover, they should contribute to the identification of the gaps and needs for alignments between the S-Cube research agenda and industry. Such gaps and needs for alignments need to be reported as part of IA-2.2.

3 Initial Set of Validation Scenarios

In this chapter we present the set of the validation scenarios identified, presented, and exploited by the S-Cube partners within various research activities of the workpackages.

3.1 Purchase Order Processing

The scenario consists of a customer, a reseller, its suppliers, a banking service, and a shipping service. The reseller offers certain products to its customers. It holds a certain part of the products in stock and orders missing products from suppliers if necessary. The customer sends a purchase order request with details about the required products and needed amounts to the reseller. The latter checks whether all products are available in stock. If some products are not in stock, they are ordered from suppliers. If the purchase order can be satisfied, the customer receives a confirmation, otherwise the order is rejected. The reseller waits, if needed, for the supplier to deliver the needed products. When all products are in place, the warehouse packages the products and hands them over to the shipment service, which delivers the order to the customer, and finally notifies the reseller about the shipment. In parallel to the packaging and shipment, the payment subprocess is performed. For that, the customer decides on the payment style and gives its payment details. The reseller contacts a banking service which authorizes the customer and credits the agreed amount.

The business process of the reseller is illustrated in the BPMN diagram shown in Figure 5. The resulting activity model is depicted in Figure 6.

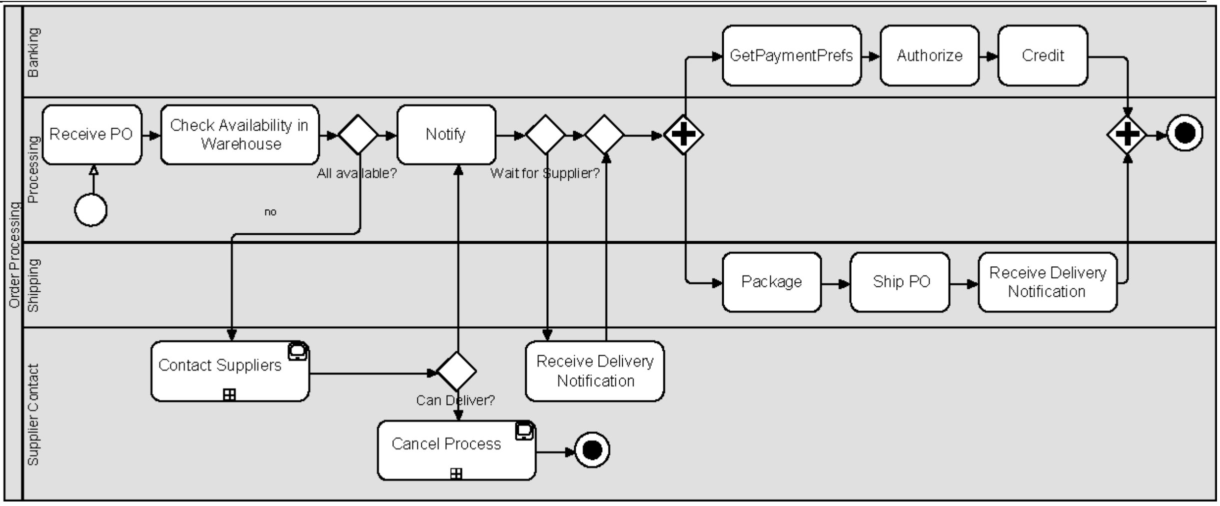

Figure 5: Business Process of the Reseller

The above scenario can be seen as a refinement of the "Source-to-stock" activity of the Main Supply Chain Process (identified as IBM_SC_01 in the Automotive case study in [7]). Thereby, the customer is the Automobile Inc. which orders parts needed for the manufacturing process at regional suppliers and which act as resellers in our scenario.

Table 5: Purchase Order Scenario Template

ID	Purchase_Order_Processing_01
Revision	1.0
Name	Purchase Order Scenario
Description	Describes how a purchase order is processed by a reseller of car parts.
Authors	Wetzstein, Branimir, Leitner Philipp, et. al.
Source	http://bibadmin.s-cube-network.eu/show.php?id=127&type=bib
Scenario Type	internal scenario
Abstraction level	type scenario
Scenario usage	positive scenario
Scenario Steps	see Table 6 and Figure 6
Case Study	Automotive, IBM_SC_01 (Main Supply Chain Process)
Goal	Process a customer purchase order
Actors	Customer (Automobile Inc.), Reseller (Regional Supplier), Suppliers, Shipper (Distribution Provider)
Additional Materials	Refines "Source-to-stock" activity in IBM_SC_01 by describing ordering of car parts from regional suppliers

Table 6: Purchase Order Scenario Structured Description

Normal Course	Alternative Course 1
1) Receive Purchase Order	
2) Check availability in warehouse	[not available] 2.1) Contact Suppliers [cannot deliver] 2.1.1 Cancel Process; end [else] continue with step 3)
3) Notify Customer	[wait for supplier] 3.1) Receive Delivery Notification
4) Package & Ship	
5) Process Invoice	

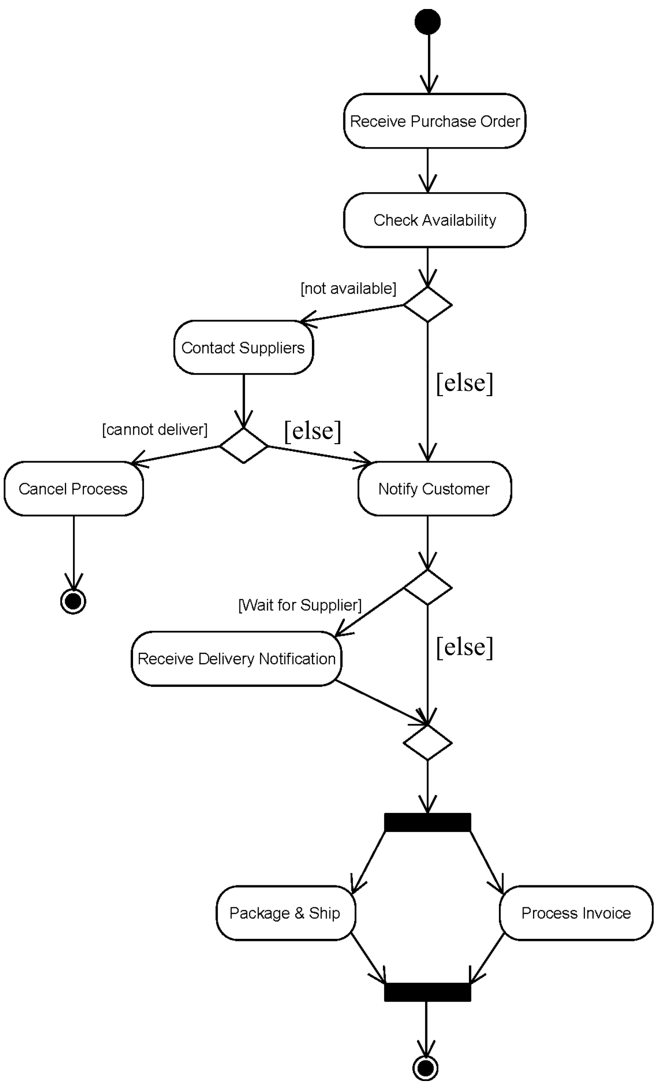

Figure 6: UML Activity Model for Reseller Process

3.2 Web service message wrapper

In recent SOA environments different protocol standards have originated. The referred protocols are those used for message exchange between customer and service provider in a process known as binding. These protocols are used to marshal the data (e.g., invocation parameters) into a format both parties can understand. Among the most known the SOAP and REST Protocol stack can be found. Although protocol provides invocation transparency to the actual applications, the discrepancy between the protocol implementations can prevent a service consumer to bind to the required service. One solution is a proxy framework that can translate from a unified message format to the format demanded by the Web service. The main steps of the simulation are:

(i) Each provider publishes its services to a service registry.
(ii) The consumer finds the published Web Service via the service registry.
(iii) If the consumer realizes it has no handler for the message protocol required by the server it communicates with the server via the proxy-framework.

The main advantage of the framework is that the consumer remains independent from the service protocol specifications, which not only can be different, but also change over time, as long as it implements the unified message format specified by the framework. The layout of the scenario is presented in Figure 7: Protocol Handler Framework. The figure shows the core of the framework. It consists of a unified message interface, the message wrapper and the protocol stack repository.

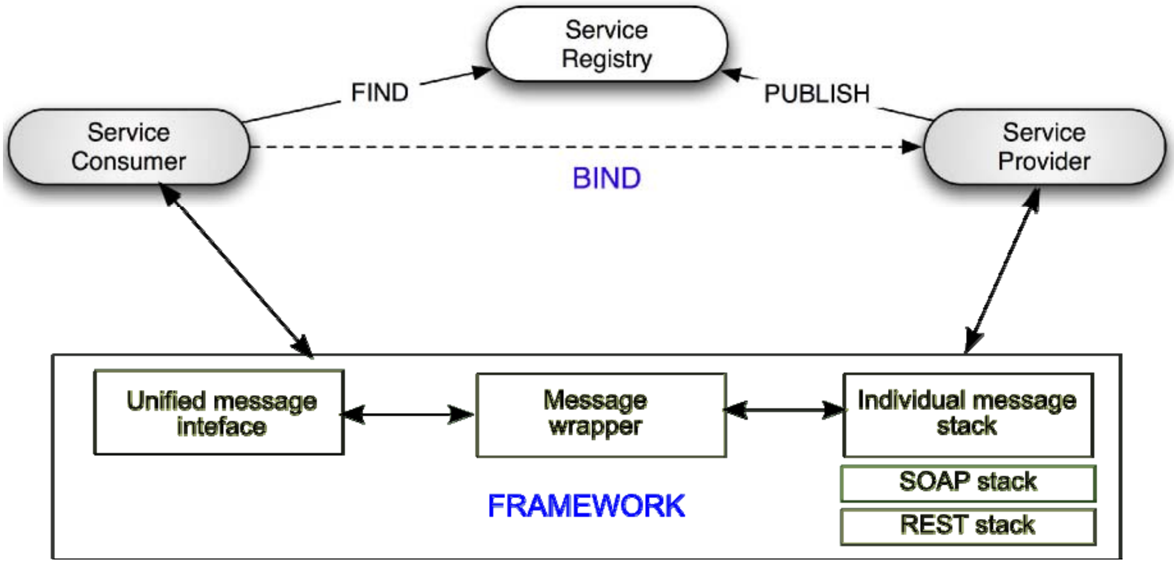

Figure 7: Protocol Handler Framework

The above scenario can be seen as a refinement to all activities in the Main Supply Chain Process (IBM_SC_01 in the Automotive case study in [7]). The case study's framework enables transparent selection and interaction of, e.g., all supplying partners operating with Web service interfaces. In such a scenario, only the presented framework guarantees freedom of choice when selecting the best fitting suppliers.

Table 7: Web Service Wrapper Scenario Template

ID	Message-wrapper-example-1
Revision	1.0
Name	Web service message wrapper
Description	Describes how services with different message format can be invoked by all kinds of consumers

Authors	Leitner Philipp, Florian Rosenberg, et. al.
Source	http://bibadmin.s-cube-network.eu/show.php?id=19&type=bib
Scenario Type	interaction scenario
Abstraction level	mixed scenario
Scenario usage	positive scenario
Scenario Steps	see Table 8, Figure 8
Case Study	IBM_SC_01 in the Automotive case study
Goal	demonstrate effectiveness of proxy framework
Actors	consumer, proxy, provider
Additional Materials	n.a.

First, clients have to find a service that they want to invoke (service discovery phase). This step is external to the framework. In this scenario service discovery problems are out of scope.

In the second step the service has to be bound (pre-processing phase). During this phase the framework will collect all necessary internal service information, e.g., for a SOAP/WSDL-based service the service's WSDL interface will be compiled in order to obtain endpoint, operation and type information.

The final step is the actual invocation of the service (dynamic invocation phase). During this phase the user input (i.e., an input message) will be converted into the encoding expected by the service (for instance a SOAP operation of a WSDL/SOAP-based service, or a HTTP GET request for REST), and the invocation will be launched using a SOAP or REST service stack. When the invocation response (if any) is received by the service stack it will be converted back into an output message and returned to the client.

The resulting activity model is depicted in Figure 8.

Table 8: Web Service Wrapper Scenario Description

Normal Course	Alternative Course 1	Alternative Course 2
1) request WSDL		
2) test invocation requirements	[requirements fulfilled] 2.1.1) message protocol required by server is handled by client ➔ continue with step 4)	
3) invoke wrapper		[cannot wrap] 3.2.1wrapper cannot handle service protocol ➔ return failure
4) invoke Web service		
5) return result		

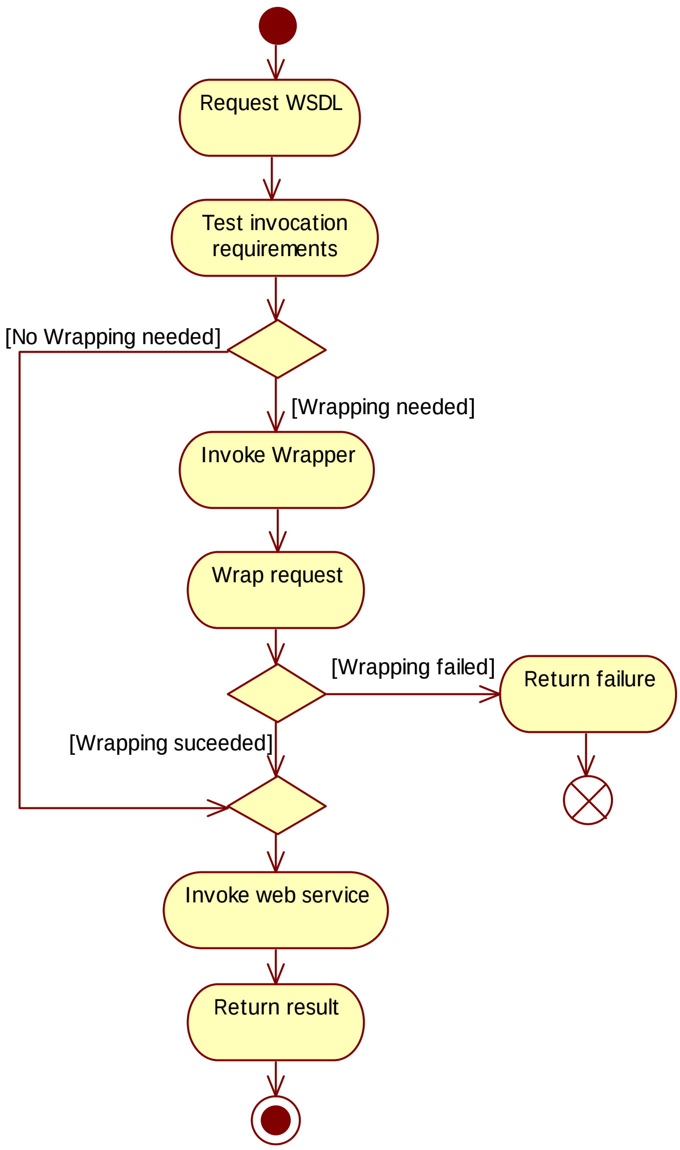

Figure 8: Web Service Wrapper Activity Diagram

3.3 Telephone number porting Web service

This scenario describes the problems arising with different implementation versions of a Web service. The reason for keeping more versions of the same service is to satisfy the requirements of the different customers/consumers. Some prospective customers will demand state-of-art implementation, whilst the regular customers will insist that on an update their requests' requirements are still satisfied. The following scenario is related to a telecommunications provider (TELCO) that provides a telephone number porting Web service to its competitors. In its beginning, the service was implemented using the state of the art technology of that time. In these days, the interface was intentionally kept basic, including only the number to port and the new provider as input, and a confirmation of the successful porting operation as output. However, not all potential users of the service (i.e., other providers) were satisfied with the simple service interface: some provider's business processes intended to send additional customer information (e.g., name, address), and in some special occasions, an indirect port via a third party was necessary. After some discussion, a new variant of the service was created with an extended interface that included this additional information. Even worse, after a company merge the new IT management of the TELCO decided to switch to Microsoft's .NET platform. Both service flavours were, therefore, ported to .NET and the Windows Communication Foundation (WCF) platform, a step that was not received benevolently by all users of the service. It was, therefore, agreed to keep the original services online for some time. Both variants of the service had to be adapted one more time – the initial definition of the confirmation turned out to be too limited, and had to be expanded. Eventually, a last evolutionary step was necessary: number porting was getting more and more popular over time, and the server hosting the number porting partner services could not deal with the load; the original server was therefore replaced by a more powerful service host. However, the old server was kept online as a fallback solution for the not yet fully tested new machine.

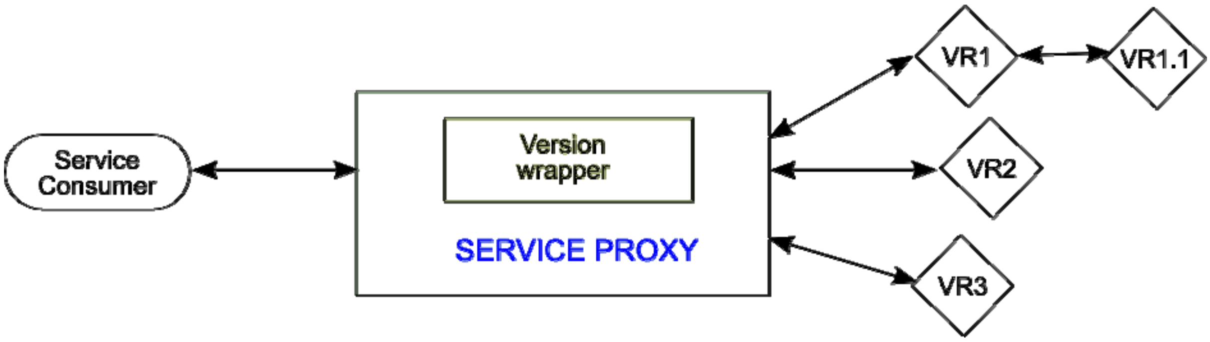

Figure 9: Version Handler Proxy

The above scenario can be seen as a refinement of possibly still offered legacy and current services in the TIS-ENG-1 scenario of the E-Government case study in [7]. As with laws, public services often decide to offer a transitional period to the public in order to let the registered citizens and companies to adapt to the new version.

Table 9: TELCO Scenario Template

ID	Number-porting-example-1
Revision	1.0
Name	Telephone number porting Web service
Description	Describes how different versions of services can be invoked by all kinds of consumers
Authors	Leitner Philipp, Florian Rosenberg, Anton Michlmayr, Schahram Dustar, et. al.
Source	http://bibadmin.s-cube-network.eu/show.php?id=15&type=bib
Scenario Type	interaction scenario
Abstraction level	mixed scenario
Scenario usage	positive scenario
Scenario Steps	see Table 18; Figure 10
Case Study	TIS-ENG-1 scenario of the E-Government case study
Goal	demonstrate effectiveness of proxy
Actors	consumer, proxy, provider
Additional Materials	n.a.

Table 10: Description of the Number Porting Scenario

Normal Course	Alternative Course 1
1) invoke wrapper	
2) choose Web service version	[not found] 2.1.1) requested service version is not available. ➔ return failure
3) invoke Web service	
4) return result	

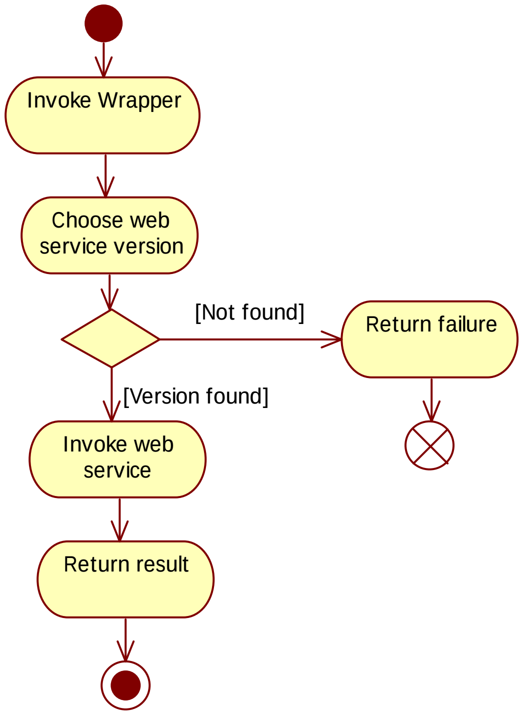

Figure 10: Number Porting Activity Diagram

The most important assumption is that all customer requests are directed to the service proxy instead of directly to the service implementations. The rest of the invocation steps is as follows. First, the client invokes the wrapper with the requested service suggesting the required version. In the second step the proxy locates the correct service version implementation and forwards/wraps the request. In the final step the response is directly returned or previously wrapped to an appropriate format and then returned to the requester.

3.4 Medical application in a virtualized distributed environment

In the E-Health scenario doctors reserve treatments on patients. The E-Health organization is then responsible for the organization of the treatments. The organization schedules the treatments depending on the handling levels assigned to each treatment by the doctors during reservation. Then the treatments are simulated in order to determine the level of expertise and resources required. In case of facial surgeries the Maxillo-Facial Surgery Simulation (MFSS) application facilitates the work of medical practitioners and provides the pre-operative virtual planning of maxillo-facial surgery. The application consists of a set of components possibly running on local and different remote machines. These components could be organized as a workflow in order to simplify the work of the end users. The main steps of the simulation are:

(i) mesh generation is used for the generation of meshes necessary for the finite element simulation;
(ii) mesh manipulation defines the initial and boundary conditions for the simulation;
(iii) finite element analysis usually running on a remote HPC cluster. In the followings we describe step by step how MFSS can be executed on the SLA-based resource virtualization (SRV) architecture.

This service management and execution architecture consists of three main components: the Meta-Negotiator (MN) responsible for agreement negotiation; the Meta-Broker (MB) responsible for selecting the appropriate service provider environment; and the Automatic Service Deployer (ASD) that supervises on-demand resource virtualization.

The MFSS application can be modelled and executed using a QoS-aware Grid modelling, planning, and execution tool. In MFSS, meta-negotiation for the MGSequence activity (used for mesh generation) is specified by means of (a) negotiation terms, (b) security restrictions, (c) negotiation protocols, (d) document languages and (e) preconditions for the agreement establishment. Negotiation terms are specified as begin time, end time, and price. In order to initiate a negotiation, GSI security is required. The negotiation is performed based on the alternate offers protocol. Therefore, the workflow application understands only the alternate offers protocol, and negotiation with resources which do not provide alternate offers protocol cannot be properly accomplished. Additional limitation considers document language used for the specification of SLAs. QoS is specified using WSLA. The service constraints are transformed into a XML-based meta-negotiation document, and this document is passed to the Meta-Broker. During the execution of the workflow, the Meta-Broker receives the service description in JSDL and the SLA terms in the meta-negotiation document. First a matchmaking process is started to select a broker that is able to execute the job with the specified requirements (resource requirements and agreement terms). The broker with the best performance values is selected, and the description and agreement is translated to the format understandable by the broker. Thereafter the broker is invoked with the transformed descriptions. The selected broker receives the descriptions and calls the ASD to deploy a service on a Cloud or a Grid, taking into account the cost requirements of the agreement, or chooses an already deployed, idle computing service. By this time the treatment simulation is ready to run and finish before the treatment should be executed on the patient. Therefore its workflow is executed and the results are returned to the EHealth organization that assigns the necessary human resources and equipment for the surgery. Finally the ASD decommissions the service in order to avoid overcharging the EHealth organization by the Cloud provider.

Table 11: Medical Application Scenario Template

ID	SZTAKI_MED
Revision	1.0
Name	Executing a medical application in a virtualized distributed environment
Description	Describes how medical planning is aided with a virtualized environment
Authors	SZTAKI, TUW
Source	http://bibadmin.s-cube-network.eu/show.php?id=135
Scenario Type	Internal scenario
Abstraction level	Type scenario
Scenario usage	Positive scenario.
Scenario Steps	See Figure 11
Case Study	EHealth: Complex Diagnostic Workflow case study's treatment planning activity (see domain description)
Goal	Allow the EHealth organization to plan facial surgeries using a flexible infrastructure
Actors	Doctor, EHealth organization
Additional Materials	CD-IA-2.2.2 figure 4.15

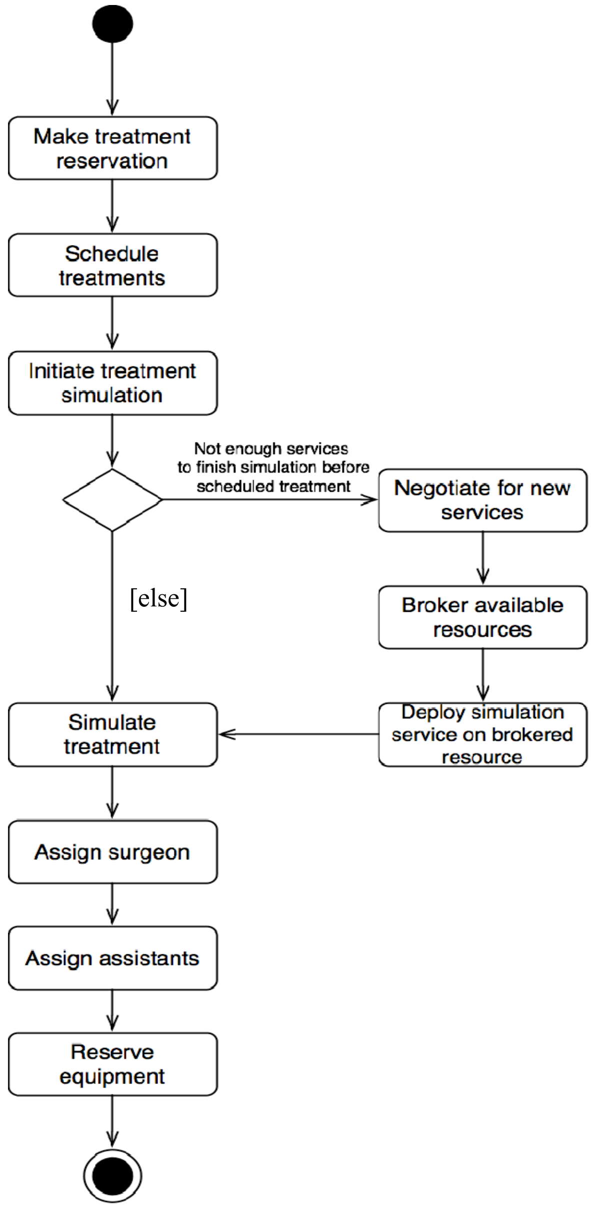

Figure 11: E-Health Scenario Activity Diagram

3.5 Self Scheduling Scenario

"E-Health seeks to provide new kind of services and a better integration of new and existing ones, thus supporting the work of the overall healthcare staff. In particular, it takes the viewpoint of medical staff and the patient during a diagnostic workflow."

"The system shall be able to reduce the overall duration of healthcare activities […]. Patient data shall be recorded from any activity of the medical staff, that is, Doctors directly involved in the patient's diagnosis, but also staff persons performing only examinations or treatments prescribed by the Doctor. Moreover, any data coming from consultations of experts shall be recorded and made available.
"The system shall facilitate the ubiquitous access to expert consultancy whenever a doctor working for a diagnosis for a specific patient needs it." [7]

The overall goal is to improve the reliability of healthcare activities through easier planning of examinations, therapies and any kind of treatments. Certain activities are possible when the necessary data are ready or other prerequisites are provided. External experts should be scheduled so that they are present whenever needed but they do not have to wait. Overall, the workflow must be organized so that it is optimal with some criteria and constraints, especially time and costs.

In general, the first step of any kind of design is to determine the components that will not be decomposed any further, elementary components, elementary operations (activities). Planning starts with a dataflow like description. The vertices of this dataflow-like graph correspond to particular activities, whereas the edges correspond to data connections between the activities. The graph describes the propagation of the data between activities. It may also contain information that enables one to derive the control structure. This graph is called the Control-Data-Flow (CDF) graph. One starts from adapting activities and defining the properties of the activities. The second step is the scheduling of activities of the CDF graph. The last step is called allocation that determines which activity will be executed by which service. Allocation, however, is strongly influenced by scheduling, whereas scheduling is restricted by the possibilities of allocation. The optimization problem is known to be NP-complete.

Even if a considerable set of scheduling and allocation heuristics exists for constant-execution time systems, allowing different execution times for activities increases the need for run-time adaptation.

Table 12: Self-Scheduling Scenario Template

ID	SZTAKI_SELF-SCHEDULING
Revision	1.0
Name	Self-scheduling workflow of services
Description	Self-scheduling workflow allocation to services
Authors	SZTAKI
Source	Zs. Palotai, T. Kandár, Z.Mohr,T. Visegrády, G. Ziegler, P. Arató, A. Lőrincz: Value prediction in HLS allocation problems using intellectual properties. Applied Artificial Intelligence , 16:117-157, 2002
Scenario Type	Internal scenario
Abstraction level	Type scenario.
Scenario usage	Positive scenario
Scenario Steps	See Table 13
Case Study	IA-2.2 EHealth: Complex DiagnosticWorkflow
Goal	Map workflow activities to services based on constraints and preferences
Actors	Doctors, medical personnel
Additional Materials	

Table 13: Scenario Steps of the Scheduling Scenario

Normal Course	Alternative Course 1
1) Separate activities (doctor plans the examinations to be performed)	
2) Prepare an initial CDF	
3) Schedule activities	
4) Assign activities	
5) Start processing the CDF. Initiate the enabled activitiess	
6) Try alternative schedules simultaneously	[there is a better schedule/assignment] ➔ keep the new schedule/assignment goto 5)
7) End	[there are activities to be enacted] ➔ goto 5)

3.6 Prescription Scenario

The role of this scenario is to show the interaction of services within a complex E-Health service based system during the prescription of medication to a patient. Upon appointment (regular, check up, etc) the doctor first contacts the system to load core patient data (name, medical number, etc) then using these data looks up the medical history of the patient. While checking the patient's status and symptoms, the doctor consults the current medication record of the patient and accesses examination results. The patient status is documented. The scenario takes various routes from this point on.

If the patient cannot be properly diagnosed based on the presented information, the doctor may require further specialised examinations. If no further examinations are required, the doctor assesses the patient's status. If the patient is a returning patient and considered cured, the process stops as not prescription is needed. If the patient is a returning patient with a terminal illness and with stable medical condition, the current medication is to be continued. If the patient needs change in therapy, or the patient is a new patient, the drug database is consulted to find the most effective medication. The drug database system receives patient information from the system (e. g. sensitivity to medicine, allergy, etc) that intelligently filters out unsuitable medicines from the list of recommended medications. Once the medication is selected - based on various rules (cost, effectiveness, preference, etc.) - the doctor proceeds to the final part of the process, in which he/she prepares the prescription, updates the central prescription database as well as stores relevant medication information in the patient's own health record.

Table 14: Scenario Template for the Prescribe Medication Scenario

ID	SZTAKI_PRESCRIPTION
Revision	1.0
Name	Create prescription /Prescribe medicatio
Description	The doctor prescribes medication to patient at given appointment visit
Authors	SZTAKI
Source	
Scenario Type	Internal scenario
Abstraction level	Type scenario.
Scenario usage	Positive scenario
Scenario Steps	See Figure 12
Case Study	E-Health case study
Goal	To prescribe medication for a patient the most efficient way
Actors	Doctor, drug database
Additional Materials	IA-2.2 E-Health case study.

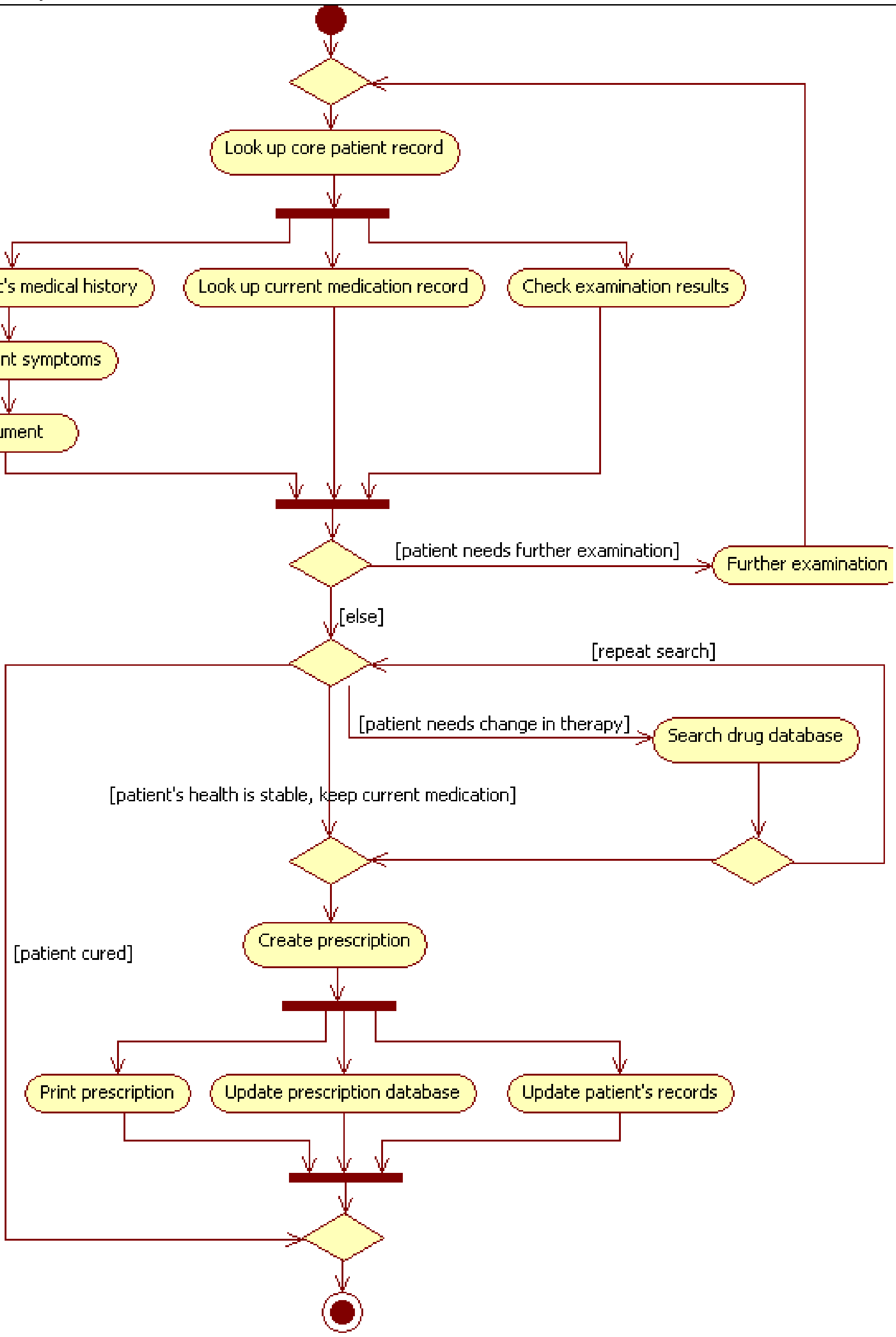

Figure 12: Activity Diagram for the Prescribe Medication Scenario

3.7 Autonomic arrangement of testing services with virtualized execution environment

In this scenario Auto Inc. is in the manufacturing step (see the step called M1.3 of IBM_SC_1 Scenario in [7]). This step can be detailed on arbitrary levels. Figure 13, however, does not plan to give a full overview on the car manufacturing process. The main objective of the figure is to show the relation between the production and testing processes of the manufacturing step. The production process is split to several phases by Auto Inc. After the completion of every phase of the production process there is an option for testing the partially ready product. Here we choose the decision after the phase 'Pressing' to show the initiation of the testing process. This decision however happens after every phase (e.g. initial assembly or painting), and the corresponding testing process could be also initiated after their completion.

Table 15: Car Assembly Testing Scenario Template

ID	SZTAKI_AUTONOMIC_CAR
Revision	1.0
Name	Autonomic arrangements of car assembly testing services with virtualized service execution environments
Description	Describes how do the service infrastructure adapts itself to the different testing procedures of the car manufactory
Authors	SZTAKI, TUW
Source	-
Scenario Type	Internal scenario
Abstraction level	Type scenario
Scenario usage	Positive scenario.
Scenario Steps	See Figure 13
Case Study	IBM SC 04 (Automotive)
Goal	Allow the Auto Inc. to adapt its service infrastructure to the current testing needs or use cloud resources if needed
Actors	Manufacture, Supplier, Warehouse
Additional Materials	CD-IA-2.2.2 figure 4.9 step M1.3

In the following paragraphs we are going to discuss the scenario steps after the decision is made. The available computing resources are limited by the Auto Inc's infrastructure, so the services evaluating the test results are autonomously managed by the three layered service infrastructure. The infrastructure layers are negotiation, brokering, and deployment. These layers adapt the Auto Inc's infrastructure to fit the current needs. In case the adaptation is not possible within the boundaries of Auto Inc. the infrastructure layers could introduce external Cloud computing resources for the actual demands.

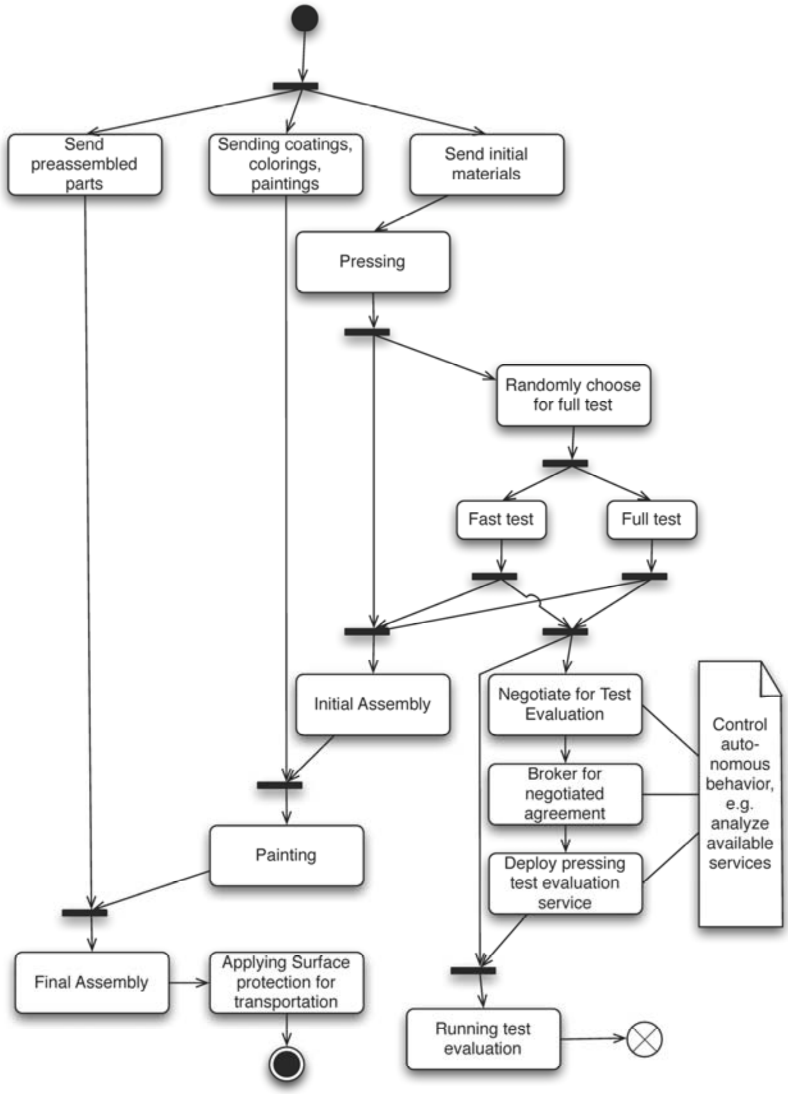

Figure 13: Detailed View of the Testing of Pressing Manufacturing Phase

1. When a new testing evaluation request is formulated the testing process initiates meta-negotiation to propagate/translate the request details towards lower levels. Before using a service, the service consumer and the service provider have to establish an electronic contract defining the terms of use. Thus, they have to negotiate the detailed terms of contract, e.g. the execution time of the service The meta-negotiation service applies self-management during the negotiation bootrstrapping procedure as follows.
   1) All candidate services are selected, where negotiation is possible or bootstrapping is required.
   2) The knowledge base is queried and potential bootstrapping strategies are found (e.g. in order to bootstrap between WSLA and WS-Agreement). In case of missing bootstrapping strategies users can define new strategies in a semi-automatic way.
   3) Finally, the negotiation is started by utilizing appropriate bootstrapping strategies.

2. The brokering services further process the test evaluation request. Brokers are the basic services that are responsible for finding the requested services with the help of a deployer service. This task requires various activities, such as service discovery, matchmaking and interactions with information systems and service registries. The autonomic behvavior of meta-brokers represented here includes:
   1) Tracking the statuses of all interconnected brokers by the IS Agent component of the Meta-Broker. The Matchmaker component also incorporates a feedback-based

solution to keep track of the performances of the brokers. The Information Collector provides broker availability and performance results based on the tracked data.

2) In case of incoming service request arrives the MatchMaker component determines the ranking of each broker according to their performance data. In case of a broker failure the ranks are recalculated and the failed broker is skipped.
3) Finally the broker with the highest rank is selected for handling the invocation.

3. Services with self-management interfaces could identify erroneous situations that could be solved by deploying an identical service on another site. We call this autonomous technique the self-initiated deployment, and in the following we describe its autonomous behavior:
    1) The Automatic Service Deployment (ASD) monitors the occurance of critical situations. First of all it looks for service instances that became defunct because they cannot modify themselves so that they can serve future requests properly. Secondly, a service instance could also get overloaded on such an extent that the underlying resources cannot handle more requests.
    2) In critical situations the ASD first decides whether the service initiated deployment is required (because the service was overloaded) or replication is necessary (because the service became defunct). In both cases the ASD identifies the service's virtual appliance (or master copy) to be deployed, then in the latter case the ASD also prepares for state transfer of the service before it is actually decommissioned.
    3) The ASD generates a deployment job for the service broker layer. This job refers to the sevice to be deployed and the state the deployment servic needs to resume. As a result the test evaluation service can serve the testing process’s request.
    4) OPTIONAL behavior in case a service is decommissioned: A service proxy is placed on the computing resource instead of the decommissioned service instance. This proxy forwards the remaining service requests to the newly deployed service. The proxy decommissions itself when the frequency of the service requests to the proxy decreases under a predefined value.

4. Finally after the requested test evaluation service is identified (or even deployed) the test process can invoke the evaluation service and retrieve the results to provide feedback for the production and design processes.

3.8 Journey planning scenario

Incorporating knowledge about end users in the engineering of SBAs is important for the development of applications suitable for use in varied, evolving environments. Related challenges include considering users’ diverse needs, skills and abilities, and translating them into corresponding sets of operations and qualities at the level of the SBA for a good user experience. This scenario will illustrate the influence of end users’ characteristics and their changing requirements in their interaction with a journey planning service.

Table 16: Journey Planning Scenario Template

Name	Journey planning
Revision	1.0
Description	Describes a citizen's use of e-Government's journey planning service
Authors	CITY
Source	
Scenario Type	Interaction
Abstraction level	Type scenario.
Scenario usage	Positive scenario.
Scenario Steps	See Table 17 and Figure 14
Case Study	E-Government
Goal	Statewide provision of online services
Actors	User, Public Body
Additional Materials	TIS_BG_1

The scenario refines the E-Government case study ([7]) that pertains to citizens accessing government services online. The actors involved are an end user requesting and using a governmental journey planning service, and a public body providing the service. As outlined in Table 17, the user sends a journey planning request with details about the start, end point and travel preferences for his journey. He receives personalised suggestions of routes to follow which he can query for additional details; relevant travel alerts; and dynamic re-mapping of his route as needed. The information communicated takes into account the user's physical abilities (e.g. accessibility of the proposed route options regarding his mobility), his current environment (e.g. a crowded public place such as a train carriage vs. a private office), and his technological platform of choice (e.g. a phone vs. a laptop computer).

Table 17: Structured Textual Description of the Journey Planning Scenario

Normal Course
1) The user requests the eGov route planning service
2) The user confirms his current location as the starting point
3) The user inputs his destination address
4) The user inputs his travel preferences
5) The user submits his query
6) The system displays a route
7) The user walks towards the underground station
8) The user verifies his direction on the area map on his device
9) The user arrives at the underground station
10) The user checks the earliest train to destination on his device
11) The user gets on the train
12) The user receives a notification of delays on the line and new estimated times of arrival
13) The user requests the computation of an alternative route to his destination

14) The system displays alternative routes
15) The user changes train at the next station and follows the alternative route
16) The user arrives at destination

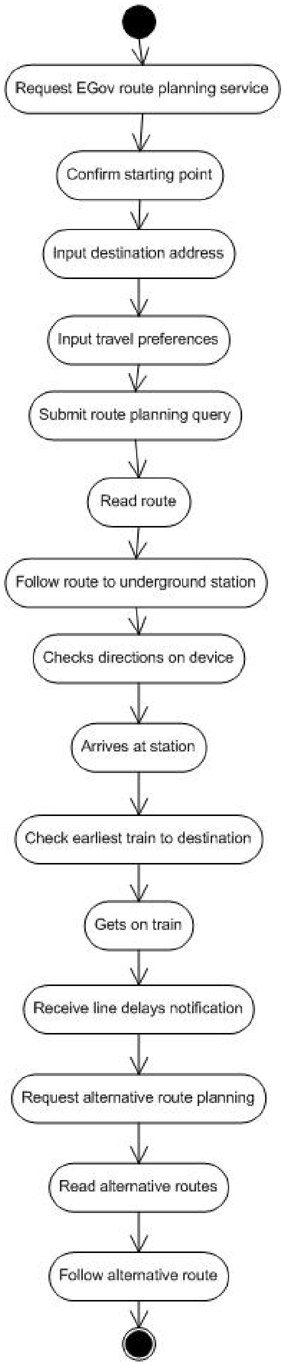

Figure 14: Journey Planning Scenario Activity Diagram

3.9 Self monitoring of blood glucose-SMBG (E-Health)

This scenario extends the eHealth case study presented in [7] that describes the management of a diagnostic workflow. It relates to the goal of easier planning of examinations and treatment for a medical condition (diabetes) through the monitoring of blood glucose. The actors involved are a patient, a eHealth organisation representing the patient's registered health centre, and "other medical staff" including nurses and GPs at the patient's health centre.

The SBMG application is made available on a health device, and issues a reminder to the patient to perform self-monitoring of blood glucose level. The monitoring procedure can be outlined as an aide memoire to the patient. The patient collects a blood sample on a testing strip and inserts it into the health device for reading and computation of his blood glucose level, which is recorded along with the time of the reading. The application subsequently recommends a course of action if and as appropriate, taking into account the patient's health data and using a diagnosis service. Finally, as the patient synchronises the data on his personal health device with that kept by his eHealth organisation, an automated alert is sent to medical staff there if the blood glucose readings cause concern for the patient's health.

The scenario will be used to explore the application of task models in the engineering of SBAs, notably during service composition, matching abstract services identified (e.g. calculate blood glucose level; send automated alert) to more generic role task models for design-time composition.

Table 18: Scenario Template for Patient Lookup

Name	Self-monitoring of blood glucose (SMBG)
Revision	1.0
Description	Describes a patient's use of an e-health system to self-monitor blood glucose level
Authors	CITY
Source	
Scenario Type	Interaction scenario
Abstraction level	Type scenario.
Scenario usage	Positive scenario.
Scenario Steps	See Table 19 and Figure 15
Case Study	eHealth
Goal	Easier planning of examinations and treatments
Actors	Patients, Other medical staff
Additional Materials	EHEALTH_BG_02

Table 19: Structured Textual Description of the Patient Lookup Scenario

Normal Course	Alternative Course
1) The user receives a reminder to monitor their blood glucose level	
2) The user opts to carry out the monitoring procedure	
3) The procedure is outlined for the user	
4) The user deposits a sample of blood on a testing strip	
5) The user inserts the strip into the device	
6) The device computes the amount of glucose in the sample	[unsatisfactory sample] 6.1.1.) The device alerts the user to redo the sampling ➔ do step 4
7) The device records the blood glucose level and the time of the day	
8) The device displays the blood glucose reading and the recommended course of action.	
9) The user syncs the data with his computer	
10) The user syncs his data with his health center's data repository	[the reading is out of the range of satisfactory values]: 10.1.1.) An automated alert is sent to the Health Center

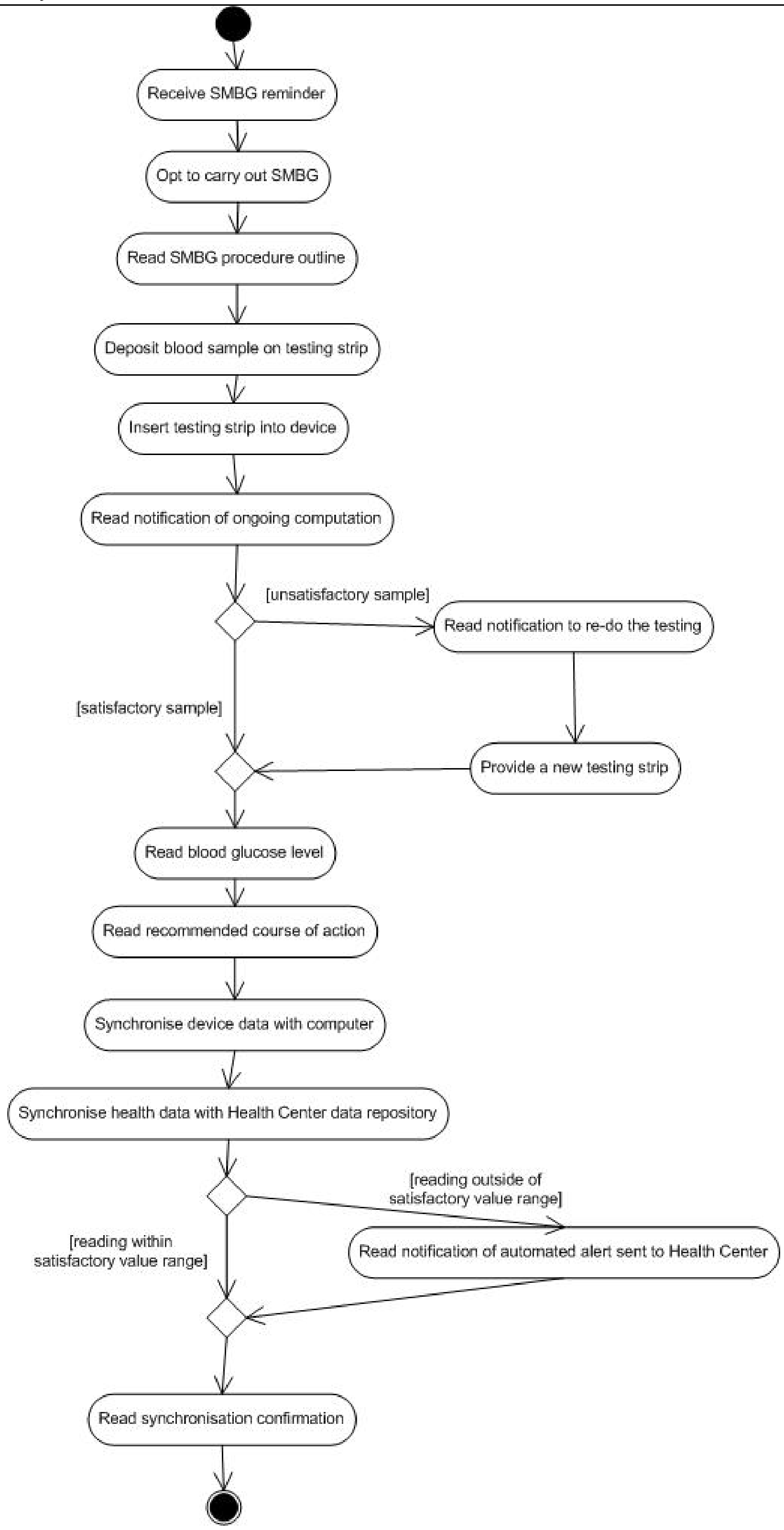

Figure 15: Patient Lookup Scenario Activity Diagram

3.10 Scenarios of grid-based assembly simulation in manufacturing process

The SBA is represented by the "manufacturing" business process (Figure 16). The goal of the process is to design and simulate the new models and move to mass production after simulation verification.

Activities are realized by the appropriate software services provided by the partners in ASN. The underlying infrastructure comprises computational resources equipped with sophisticated resource management and load balancing mechanisms.

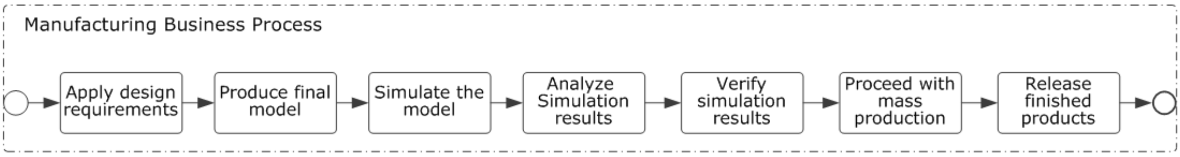

Figure 16: Grid-based Simulation Scenario

The common template for the set of scenarios is represented in Table 20.

Table 20: Grid-based Simulation Scenario Template

ID	Auto-Crosslayer-1
Revision	1.0
Name	Grid-based simulation scenario
Description	Exemplifies the wrong adaptation and monitoring when there is lack of cross-layer aspects
Authors	Asli Zengin, Raman Kazhamiakin, Marco Pistore
Source	"Cross-layer Adaptation and Monitoring of Service-Based Applications" to be submitted to MONA+ 2009
Scenario Type	Internal scenario
Abstraction level	Mixed scenario
Scenario usage	Negative scenario
Scenario Steps	See Table 21, Table 22 and Table 23
Case Study	Automotive case study; IBM_SC_01, IBM_SC_04
Goal	Just-in-time production
Actors	Auto Inc. Manufacturing Factory
Additional Materials	-

3.10.1 Diagnosis Problem

While the process is executed, the "analyze simulation results" activity takes abnormally much more time than it is expected; i.e., the composition PPM is violated, which is detected by the appropriate monitor at the SCC layer. As a result, in order to satisfy the adaptation requirement to compensate the PPM violation, the service is substituted with another service and the analysis task is performed again.

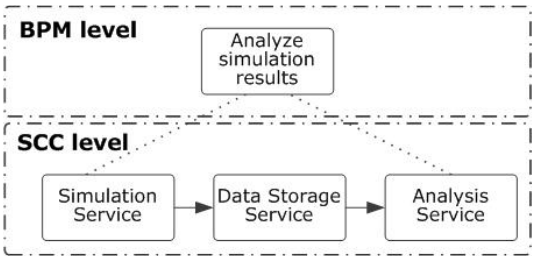

Figure 17: Overview of the Diagnosis Problem

Table 21: Detailed Description of the Diagnosis Problem Scenario

Normal Course	Alternative Course 1
1) Monitor SBA: at SCC layer, analysis results for simulations cannot be produced.	[apply cross-layer monitoring] 1.1.1) Cross-layer diagnosis of monitored event: at SI layer, network problem between data and computing resources.
2) Determine adaptation requirement: stabilize analysis service in SCC.	1.1.2) Determine adaptation requirement: stabilize network in SI.
3) Determine adaptation strategy: replace service.	1.1.3) Determine adaptation strategy: move to back-up network

3.10.2 Adaptation Effectiveness

At BPM layer, it has been monitored that the KPI value for average duration of simulations is not met because the simulation runs take too much time. The adaptation requirement is to compensate KPI violation by reducing the total simulation time. To achieve that, it is decided to parallelize simulation tests at service level by making an agreement with a new simulation service provider and enabling the concurrent run of these two different simulation services.

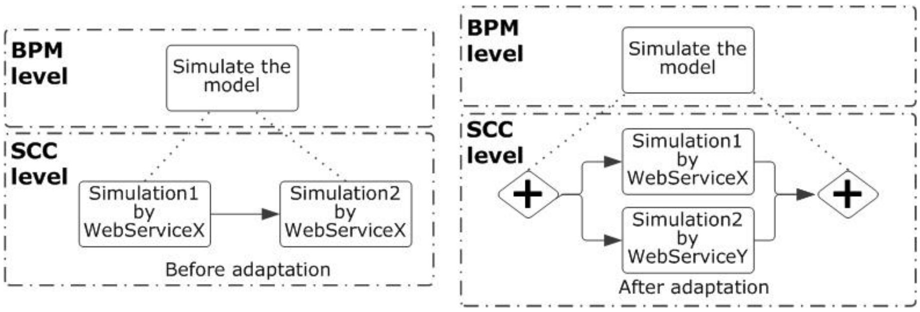

Figure 18: Overview of the Adaptation Effectiveness

Table 22: Detailed Description of Grid-based Simulation Scenario

Normal Course	Alternative Course 1
1) Monitor SBA: KPI violation, simulation runs take too much time.	
2) Determine adaptation requirement: compensate KPI violation.	
3) Determine adaptation strategy: in SCC, add a new simulation service in parallel.	[apply cross-layer adaptation] 3.1.1) Check cross-layer effectiveness requirements: new service should use different (additional) infrastructural resources.
4) Realize adaptation.	3.1.2) Realize adaptation.
5) Monitor SBA: simulation runs still take too much time because new service uses the same computation resources at SI level. (adaptation useless)	3.1.3) Monitor SBA: simulation runs ok, KPI ok.

3.10.3 Adaptation Compatibility

At the level of service infrastructure, it has been monitored that the required QoS value of the simulation service is not met due to the unavailability of some of the storage resources (they have reached full capacity utilization being unable to store new simulation data). The adaptation requirement posed at the SI layer in this case is to compensate the QoS degradation (SLA noncompliance). It is achieved by the following adaptation strategy: perform load balancing of the storage resources (transfer excessive data to additional resources).

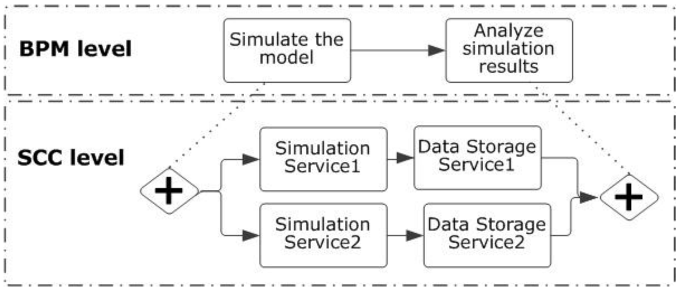

Figure 19: Overview of the Adaptation Compatibility

Table 23: Detailed Description of the Adaptation Compatibility Scenario

Normal Course	Alternative Course 1
1) Monitor SBA: at SI level, QoS violation due to shortage of some storage resources.	
2) Determine adaptation requirement: compensate QoS degrade.	
3) Determine adaptation strategy: in SI apply load balancing among storage resources.	[apply cross-layer adaptation] 3.1.1) Check cross-layer compatibility requirements: load balancing is not allowed due to a business rule at BPM.
4) Realize adaptation.	3.1.2) Investigate alternative adaptation.
5) Monitor SBA: violation of privacy at BPM level due to the business rule “simulation data provided by each service must be kept on different data server”. (adaptation harmful)	➔ End of scenario

3.11 Plan and Purchase Materials Scenario

In this scenario the “Plan and purchase material from suppliers” business process is considered. The process is correlated with manufacturing process; the goal is to acquire the required components before moving to the manufacturing phase. The “decide on supplier” business activity is provided by a service that keeps up-to-date information about available suppliers and their offerings, and discovers, and selects the most appropriate supplier based on the purchase specification of the material. The “delivery of material” business activity has to be performed by a delivery service, probably involving some other services. The SI layer is realized by the appropriate service execution platform, e.g., composition engines, service bus, database, etc.

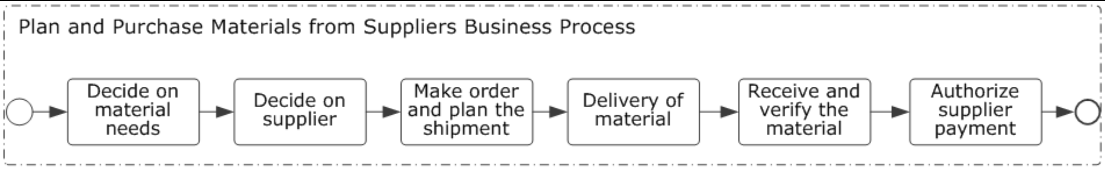

Figure 20: Overview of the Plan and Purchase Materials Scenario

At the BPM layer, it has been monitored that the logistics provider company, which is responsible for the delivery of material, does not comply with the SLA contract. The adaptation requirement is to compensate the SLA violation, and can be achieved by switching to a new logistics provider available in the ASN. This, in particular, requires the negotiation and agreement with a partner with matching service offerings. During the adaptation action, several process instances might have already been started for some material and components. Change of the logistics provider will indeed affect these instances as the corresponding activity ("delivery of material") has to be performed by the new one. Indeed, it is also necessary to adapt at the SCC layer by changing the composition instance accordingly: it is needed to bind to the new services corresponding to the new provider, align with the new interface and data formats if they are different for the new provider, perform some compensation actions for the old provider if the delivery procedure has already been triggered, etc. This may also require adaptations at the SI layer as the new service may have particular constraints on the low-level protocols or security policies.

Table 24: Plan and Purchase Materials Scenario Template

ID	Auto-Crosslayer-4
Revision	1.0
Name	plan and purchase materials
Description	Exemplifies an adaptation strategy that is incomplete because there is lack of cross-layer approach
Authors	Asli Zengin, Raman Kazhamiakin, Marco Pistore
Source	"Cross-layer Adaptation and Monitoring of Service-Based Applications" to be submitted to MONA+ 2009
Scenario Type	Internal scenario
Abstraction level	Mixed scenario
Scenario usage	Negative scenario
Scenario Steps	See Table 25
Case Study	Automotive case study; IBM_SC_01, IBM_SC_05
Goal	Optimize transportation (IBM_BG_02) Efficient material sourcing (IBM_BG_03)
Actors	Auto Inc Manufacturing Factory, Auto Inc EU Headquarter, Supplier, Logistics Provider
Additional Materials	Refines S1.1 – Schedule Product Deliveries Activity in Placing Purchase Orders & Schedule Products Delivery Business Process (IBM_SC_05). Shows cross-layer adaptation need on a scenario on the refined model.

Table 25: Plan and Purchase Materials Scenario Description

Normal Course	Alternative Course 1
1) Monitor SBA: delivery company has not delivered the material on time; SLA incompliance	
2) Determine adaptation requirement: compensate QoS degrade.	
3) Determine adaptation strategy: at BPM level, change delivery company and sign the new SLA contract with the new company.	[apply cross-layer adaptation] 3.1.1) Check cross-layer integrity requirements: services of the new delivery company should be updated in all the running processes via dynamic service binding.
4) Realize adaptation.	3.1.2) Realize relevant adaptations at BPM, SCC and SI levels.
5) Monitor SBA: Ongoing processes, which have started before the adaptation realization, still use the old delivery company at SCC and SI levels.	3.1.3) Monitor SBA: Ongoing processes are ok, not negatively affected by the adaptation.

3.12 Automotive process monitoring scenario

In this scenario we consider the business process defined in the automotive case study, where several organizations cooperate in order to produce automobile (see Figure 4.3 of the deliverable CD-IA-2.2.2). According to the business goals expressed for this case study, the company needs to monitor the business process in order to calculate the related KPIs.

In this case, we are not proposing a single scenario but a class of scenario starting from the consideration that in the business process, not all the information exchange automatically occur, but in some case manual communications happen; for instance, using fax, ordinary mail, phone calls and so on. So, in some cases employees are in charge of manually inserting data in the related information systems.

This situation usually occurs when considering business processes involving the supply-chain like in the case of the automotive case study. It also might happen when some internal activities are outsourced.

Since, the data that can be potentially used for calculating the KPIs are the ones that are exchanged by the organizations, in some cases, the provision of all measurement data needed for KPI calculation is not feasible or not possible. As a consequence, in practice not all process activities are monitored or monitorable and thus the exact calculation of a KPI might not be possible.

Table 26: Template of the Automotive Process Monitoring Scenario

ID	Automotive_process_monitoring
Name	Automotive Process Monitoring
Description	Identify how to monitor KPI in case of service based business processes
Authors	Cinzia Cappiello, Kyriakos Kritikos, Pierluigi Plebani, Branimir Wetzstein
Source	N/A so far

Scenario Type	Internal scenario
Abstraction level	Type scenario.
Scenario usage	Positive scenario.
Scenario Steps	See Figure 21
Case Study	IBM_SC_01
Goal	IBM_BG_01, IBM_BG_02, IBM_BG_03
Actors	Suppliers, Automotive Company, Customers
Additional Materials	None

Since we are consider a class of scenarios starting from the same business process, hereafter we introduce how a scenario is defined.

Starting from the execution process defined for the automotive case study, we assume that the information systems belonging to the actors in the supply chain are not directly integrated. As a consequence, suppliers and retailers communicate with the Auto Inc. by using traditional channels, i.e, fax, phone calls and documents. So, the information regarding the transactions with these actor needs to be manually transferred from one information system to another one (e.g., data entry). In addition, we also assume that the logistics activities are performed by an external department. Thus, the Auto Inc is no longer aware about how the process internal to the logistic works. From the Auto Inc. perspective, the logistic is seen as a black box.

Figure 21 shows this configuration by revising the execution process described in the Figure 4.3 of the deliverable CD-IA-2.2.2.

It is worth noting that the configuration here described is only one of the possible configurations. So, depending on the KPI to be analyzed, different configurations of monitorable/not-monitorable messages could be taken into account.

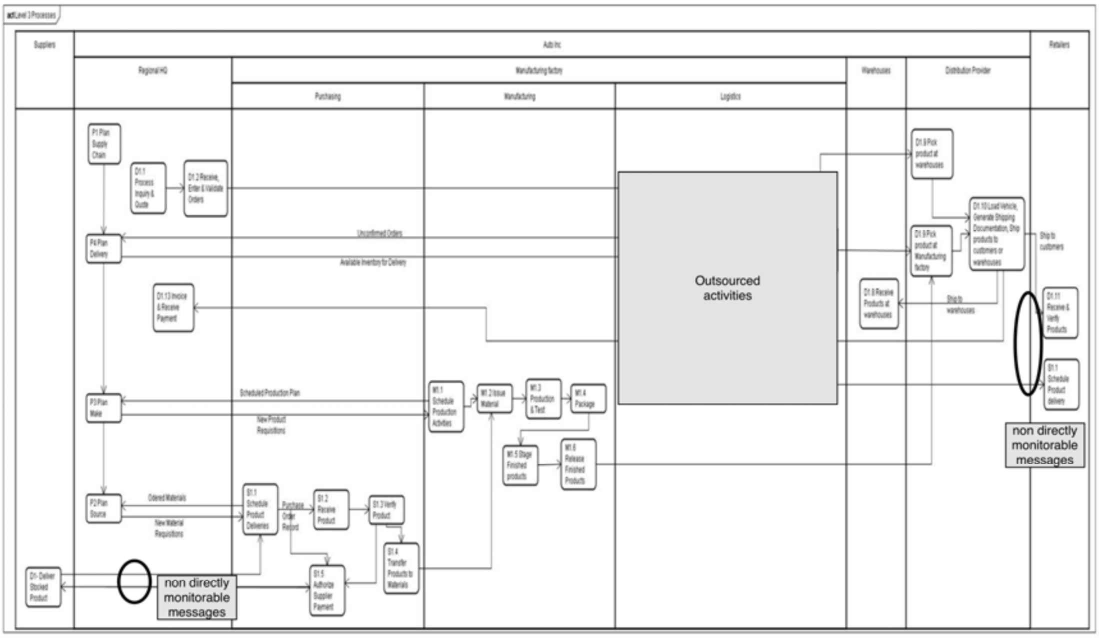

Figure 21: Possible Configuration of Auto Inc. Execution Process (see Figure 4.3 CD-IA-2.2.2)

3.13 Collaborative Transport Chain Control

During the distribution phase of wine production, the temperature variation must meet some strict requirements (i.e., no wide fluctuations and it must be kept within a specific range; see step "control temperature during distribution" of scenario WINDERY-S-DS). In this scenario, the wine is hence controlled by wireless sensor technology, such as RFID.

Table 27: Collaborative Transport Chain Control Scenario Template

ID	WINERY-S-1
Revision	1.0
Name	Collaborative Transport Chain Control
Description	Describes the reaction to situations in which monitored values differ from the estimated range while transporting the wine.
Authors	UniHH
Source	
Scenario Type	Internal scenario
Abstraction Level	Type scenario
Scenario usage	Positive scenario
Scenario Steps	See Figure 22
Case Study	Wine case study; WINERY-S-DS
Goal	WINERY-S-BG3, WINERY-S-DA1, WINERY-S-DA4
Actors	Quality Manager, Delivery Company, Retailer
Additional Material	

When noticing a deviation of the measured values from the estimated range (malfunction), a predefined process (see Figure 22) reacts to the situation in acquiring the current position, calculating the time of arrival and – if the time until arrival is considered too long – redirecting the container, e.g. to a cold storage, with informing the retailer. While due to performance restrictions wireless sensors are not able to execute the process itself, already existing devices and resources in the mobile vicinity should support the execution of the process.

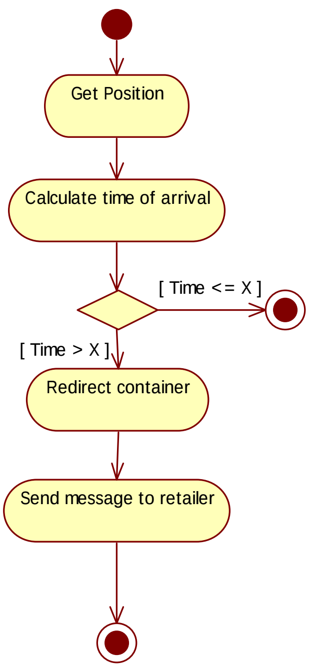

Figure 22: Collaborative Transport Chain Control Scenario Diagram

3.14 Handling occurrences of harmful animals scenario

This scenario addresses the handling of occurrences of harmful animals in the vineyard in case of organic cultivation. To achieve an organic certification, usage of chemical pest control has to be strictly restricted. Instead, biological pest control is used to fight harmful animals.

Table 28: Harmful Animals Scenario Template

ID	WINERY-S-2
Revision	1.0
Name	Handling occurrences of harmful animals (pest)
Description	Describes the reaction to occurrences of harmful animals in the vineyard
Authors	UniHH
Source	will be published
Scenario Type	mixed scenario
Abstraction Level	Type scenario
Scenario usage	Positive scenario
Scenario Steps	see description and figure below
Case Study	Wine case study; WINERY-S-CH-1
Goal	WINERY-S-BG2, WINERY-S-BG3, WINERY-S-DA1, WINERY-S-DA2
Actors	Agronomist, Oenologist, Wine Grower, Quality Manager
Additional Material	Nonte

If an affection has been detected, the oenologist determines the pest and searches (e.g. with help of a knowledge database) for useful creatures to treat the pest. If such an animal is found, an according order is placed. Otherwise, the wine grower is informed to destroy the affected plants in order to avoid spread of the pest. In parallel to these (mainly) manual actions, additional environmental data (temperature, humidity, wind, etc.) and the exact location are automatically gathered and an alarm is sent to the quality manager. After all, the incident has to be documented with all collected information.

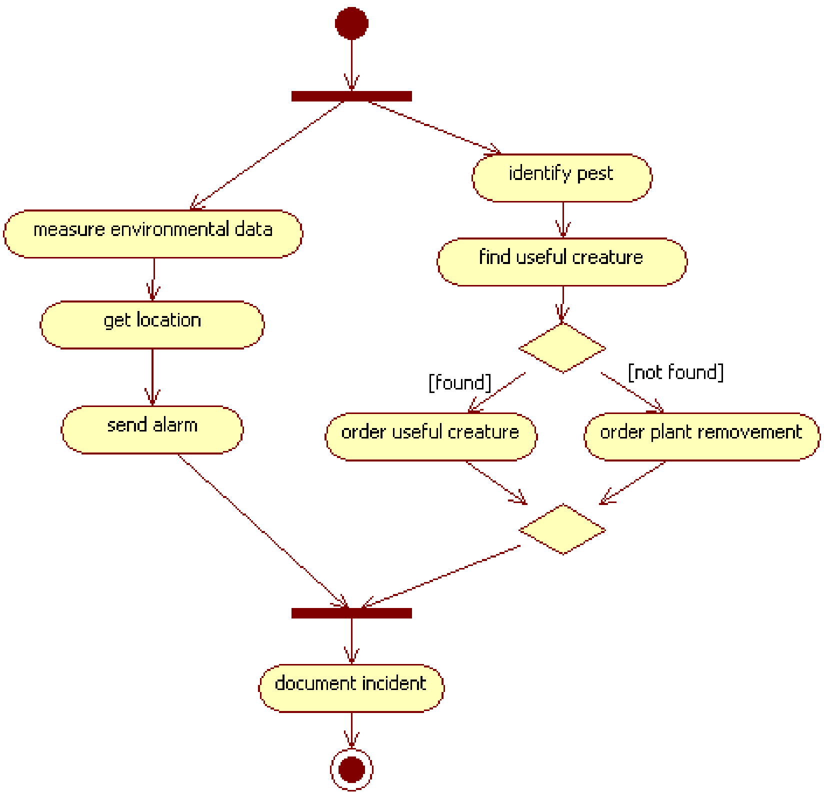

Figure 23: Harmful Animlas Scenario Activity Diagram

3.15 Compensated Group Reservation

Finding techniques to automatically generate the expected QoS for a service composition, which presents itself as a service, is a non-trivial but important task in order to automate the creation and self-management of SOC-based systems. Several challenges stem from this requirement. One is to actually traverse the structure of every service composition, or service composition candidate, so that the QoS characteristics from each of the invoked services is taken into account to compute the actual QoS of the composition.

Given that the contents of the actual message contents can influence the runtime behavior of a service (and therefore of a system composition where this service appears –e.g., reserving a hotel room for one person is, from the point of view of spent resources, not the same as reserving for one hundred, since more messages are sent, more bandwidth is spent, more database transactions can be performed, etc.), in this scenario we will present an approach to automatically deduce QoS expressions which take into account the runtime data which is processed by some composition. More precisely, we will aim at working out functions which are upper and lower bounds of the possible QoS values.

Taking actual data into account when generating QoS expressions for service compositions opens up a series of possibilities which are out of reach for the case of probabilistically determined QoS.

We will illustrate this claim with this scenario that addresses the frequent situation where several items in a group have to be reserved before a processing activity can commence. Although items in the group are reserved individually, the reservation of the whole lot is successful if and only if all items have been individually reserved. Partial reservations are not allowed: if some item reservation fails, previous item reservations must be cancelled. To ensure the above mentioned consistency requirement, this scenario has to take into account explicit compensation of individual reservations as part of its business logic.

Typical examples are: making a reservation of homogeneous resources such as rooms for a group of passengers, or booking production of heterogeneous car parts and accessories necessary for assembly of a production lot in automotive industry.

Table 29: Compensated Group Reservation Scenario Template

ID	Group_Reservation_01
Revision	1.0
Name	Compensated Group Reservation
Description	Reserve all elements in a group (a list part descriptions and quantities), making sure that no partial reservations are made
Authors	Dragan Ivanović, Manuel Carro, Irena Trajkovska
Source	UPM Technical Report CLIP3/2009.0
Scenario Type	Interaction Scenario
Abstraction Level	Type Scenario
Scenario Usage	Positive (with compensation)
Scenario Steps	See Table 30, Figure 24
Case Study	Automotive Industry
Goals	IBM_BG_03
Actors	Company Headquarters (as Client), Local Branch (as Agent), Local Suppliers (as Provider)
Additional Materials	None

The sequence diagram for this scenario is presented on Figure 24.Participants: (i) Provider requesting group reservation; (ii) Agent performing compensated group reservation; (iii) Provider for individual parts (group items).

Table 30: Description of the Compensated Group Reservation Scenario

Normal Course	Alternative Course 1
1) Agent receives from Client a group of items that need to be reserved for the lot processing to commence.	3a) In step 3, Provider fails to reserve the requested item.
2) Agent requests Provider to reserve a single item from the group (part type and quantity) and waits for reply.	3a.1) Agent asks Provider to cancel an already reserved item.
3) Provider confirms successful item reservation. If the item list is not exhausted, Agent iterates to the step 2.	3a.3) If there are more uncanceled item reservations, Agent interates to the step 3a.
4) Agent signals successful group reservation to Client, and returns the list of item reservations.	3a.3) Agent signals failure to Client.

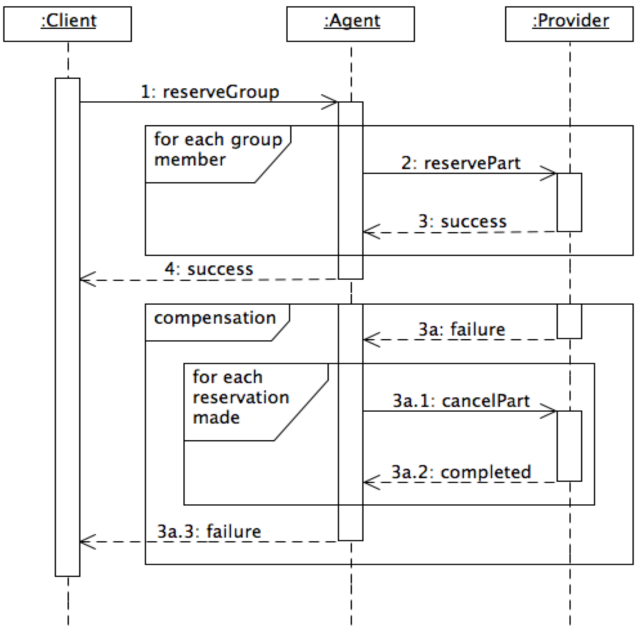

Figure 24: Sequence Diagram for the Compensated Group Reservation Scenario

3.16 Automotive Purchase Order Processing Scenario

The supply chain operations reference model (SCOR) provides abstract guidelines for building the Supply Chain Case Study, i.e. the abstract level 3 activities (see [7], scenario IBM_SC_01, Figure 4.9). SCOR lets the level 4 process implementations in users' hands. This purchase order processing scenario is an example how to realize SCOR level 3 activities using SOA-based processes for an enterprise in the automobile industry called Automobile Incorporation (aka AutoInc). AutoInc contains different business units, e.g. Sales, Logistics, Manufacturing, etc, and collaborates also with other partners like suppliers, banks, carriers, etc.

Table 31: Automotive Purchase Order Processing Scenario Template

ID	Purchase_Order_Processing_BPM
Name	Purchase Order Processing Scenario
Description	Describes how a purchase order from an individual customer or a retailer is processed by an automobile incorporation (Auto Inc.). For more detailed description see the previous section.
Authors	Tilburg
Source	
Scenario Type	internal scenario
Abstraction level	type scenario
Scenario usage	positive scenario
Scenario Steps	see description below
Case Study	Automotive, IBM_SC_01 (Main Supply Chain Process)
Goal	Process a customer/retailer purchase order from receiving the order until shipment and invoicing
Actors	Customer (Individual Customer/Retailer); Different actors (business units) within the Auto Inc. :Sales, Customer Relationship Management (CRM), Enterprise Resource Planning (ERP), Logistics; external Shipper (Distribution Providers), Bank
Additional Materials	Refines the steps D1.2, D1.3, D1.4, D1.5, D1.6, D1.7, D1.9, D1.10 and D1.13 within the “Delivered-Stocked-Products” activity in IBM_SC_01.

Figure 25 and Figure 26 visualize the entire order processing scenario modelled in BPMN; Figure 27 demonstrates the activity diagram of the scenario.

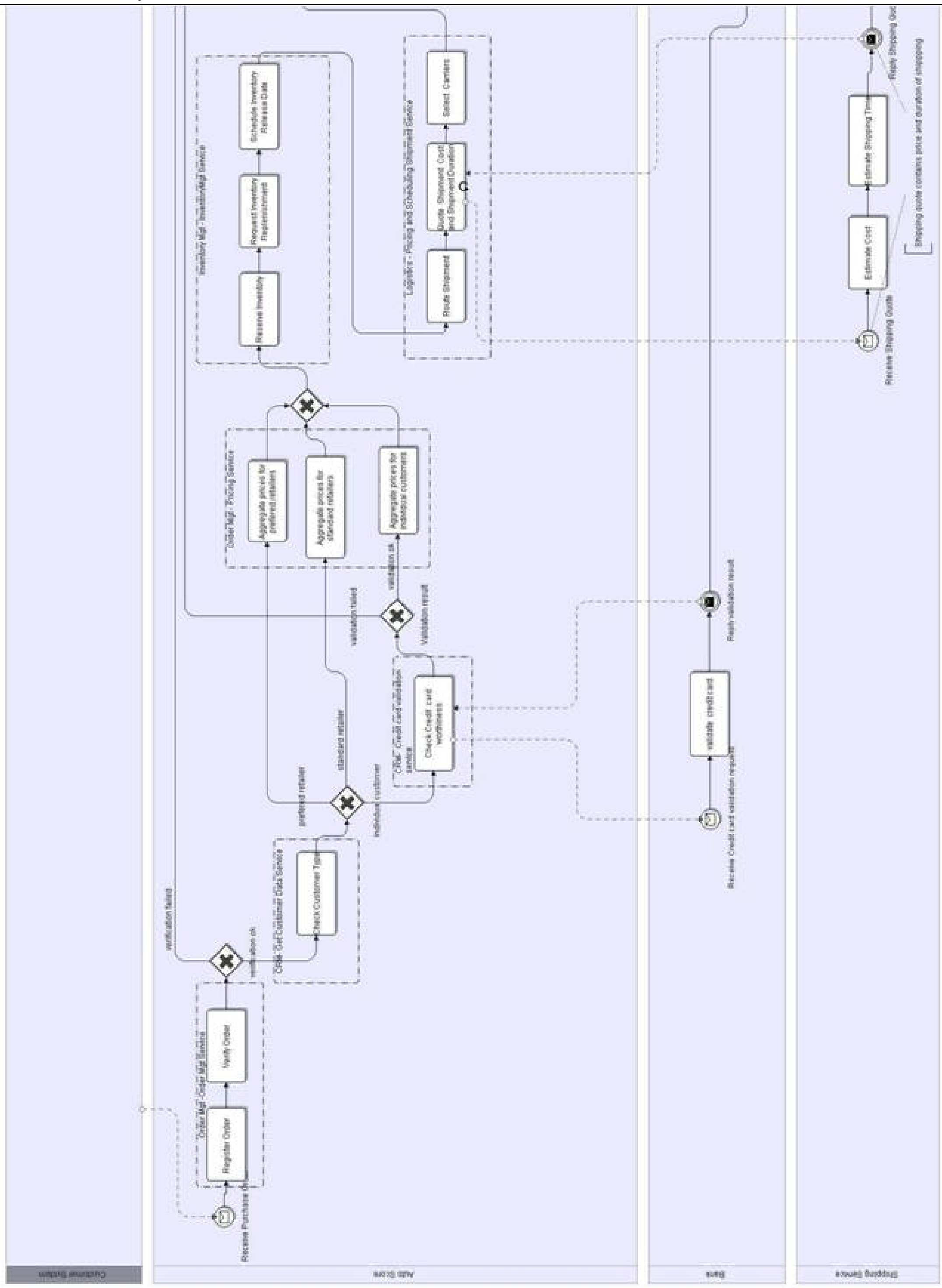

Figure 25: Automotive Purchase Order Processing Scenario (Part 1)

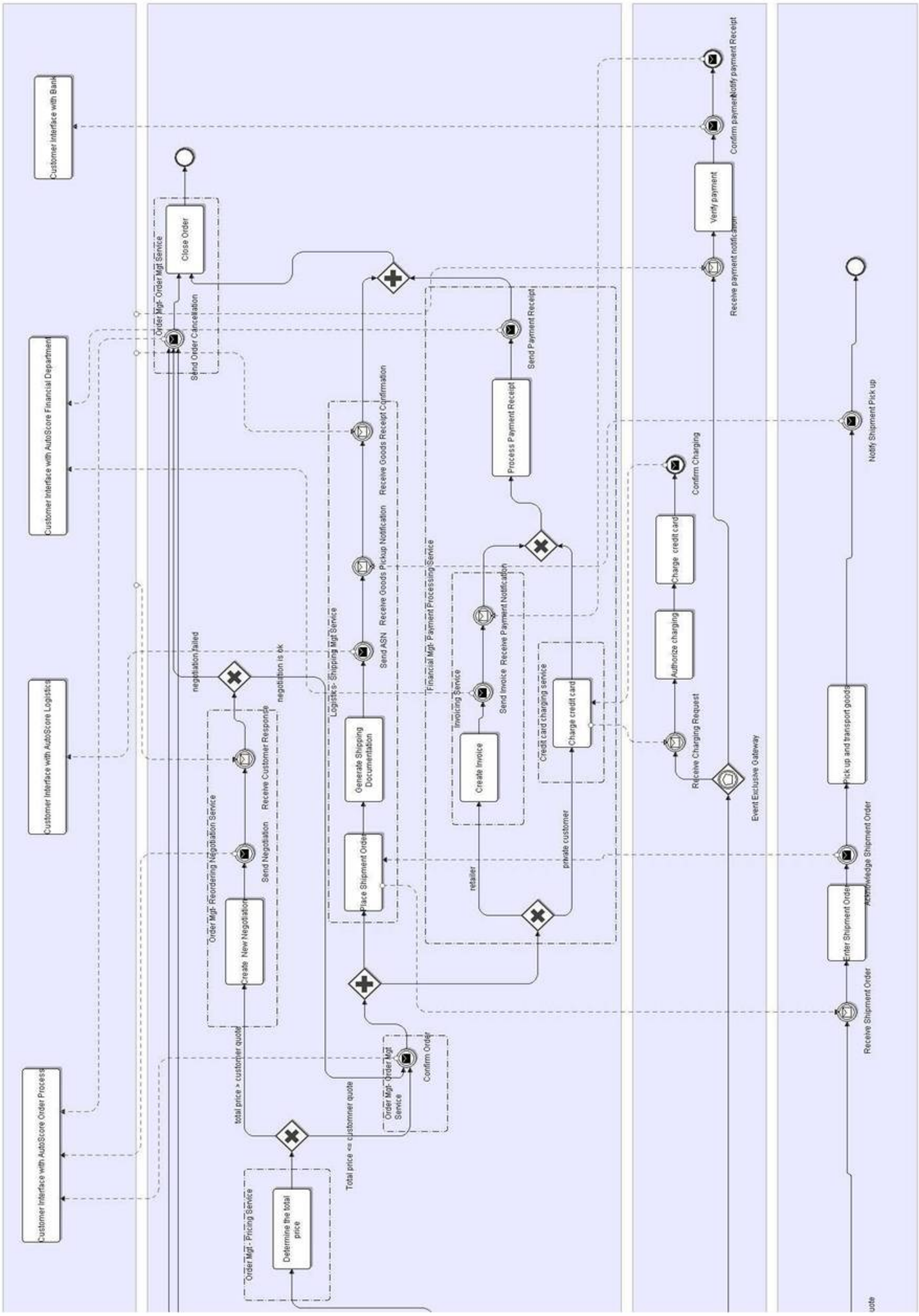

Figure 26: Automotive Purchare Order Processing Scenario (Part 2)

Table 32: Automotive Purchase Order Scenario Structured Description

Normal Course	Alternative Course 1	Alternative Course 2	Alternative Course 3
1) Receive Purchase Order			
2) Register Purchase Order			
3) Verify Purchase Order	[verification fails] 3.1) Reject Order [go to step 11]		
4) Check customer type	[type == preferred retailer] 4.1) Aggregate prices for preferred retailers	[type == standard retailer] 4.2) Aggregate prices for standard retailers	[type == individual customer] 4.3) Check Credit card Worthiness [credit card check fails] 4.3.1) go to step 3.1 [credit card check succeeds] 4.3.2) Aggregate prices for individual customers
5) Plan Inventory Release[1]			
6) Plan Shipment[2]			
7) Finalize the total price of order	[final price > quote] 7.1) Reorder Negotiation[3]	[negotiation is not ok] 7.1.1) Go to step 3.1	
8) Confirm Order	[customer = retailer] 8.1) Invoice customer[4]	[customer = individual customer] 8.2) Charge credit card	
9) Process Payment Receipt			
10) Ship Products[5,6]			
11) Close Order			

[1] This step includes "Reserve Inventory", "Request Inventory Replenishment", and "Schedule inventory release"
[2] This step includes "Route Shipment", "Quote Shipment Cost and Duration", and "Select Carriers"
[3] This step includes "Create new negotiation", "Send negotiation", and "Receive Customer Response"
[4] This step includes "Create Invoice", "Send Invoice" and "Receive Payment Notification"
[5] This step includes "Place Shipment Order", "Generating Shipping Documentation", "Send ASN", "Receive Goods Pick-Up Notification" and "Receive Goods Receipt Confirmation"
[6] This step is executed simultaneously with step 8.1 or 8.2

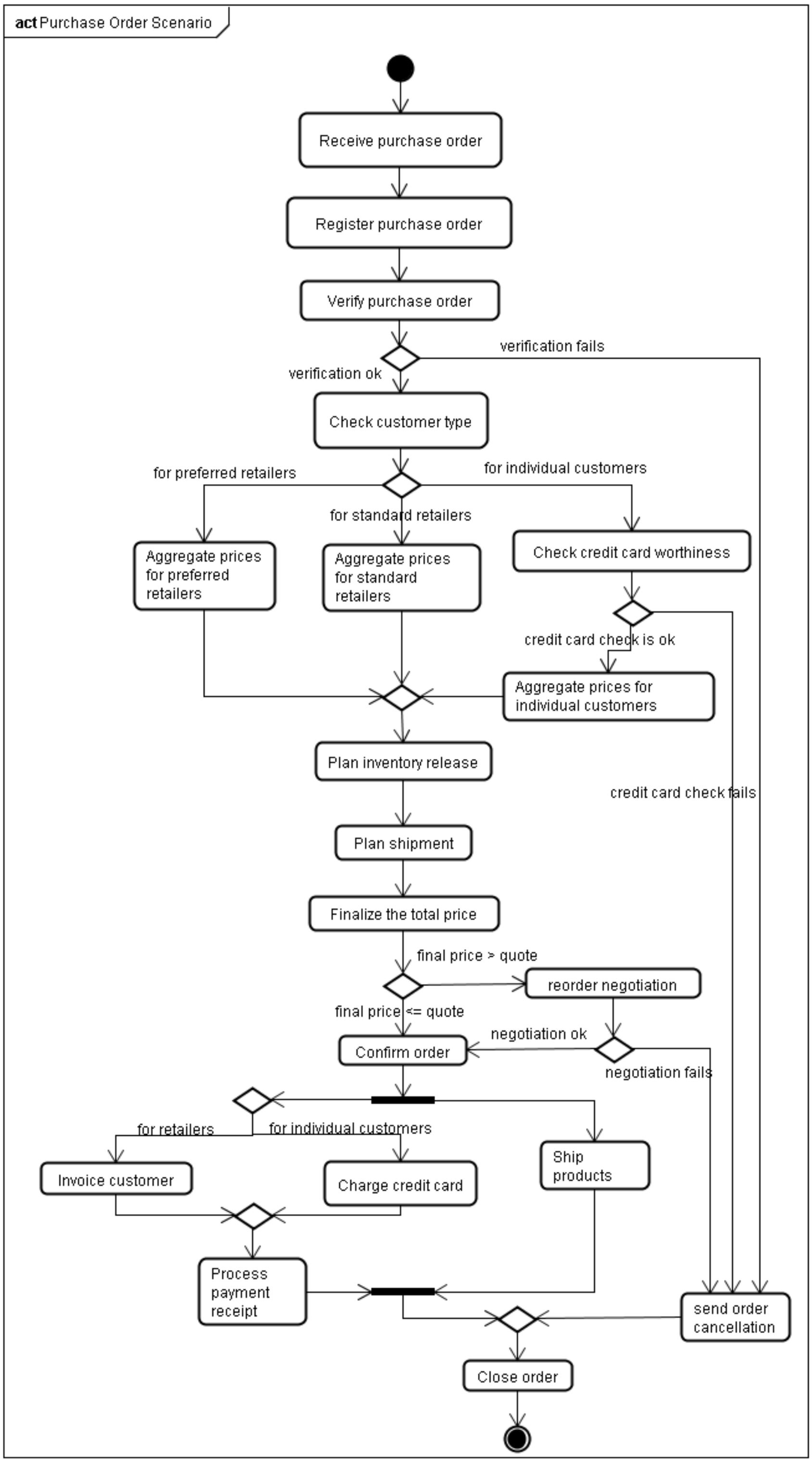

Figure 27: Activity Model of Auto Inc. in the Purchase Order Processing Scenario

4 Scenario Validation Plans

In order to validate the S-Cube Integrated Research Framework and the research results within the framework it is necessary to define and realize the scenario validation plan. The plan should include the information about the organizational settings for the validation, the results of the experiments and their explanation. It will be extremely beneficial for the S-Cube project not to report successful validations but also failed ones. Failed validations should be used to learn and, more importantly, not to repeat such a validation within the project.

In the following sections we present the key elements that such a plan should report as well as the example of how such a validation plan may look like when applied to one of the scenarios described in Chapter 3.

4.1 Validation Plan Template

Once the validation scenario is selected/created a validation plan should be developed, which clearly states the validation goal, the validation object, its relation to the integration framework, the scenarios used for this validation, the organisational setting (if any), a brief description about how the validation was performed, the results of the validation and experiences gained in the form of Dos and Don'ts (see Table 33). These elements are described below:

- *Validation Goals*: Validation goals should clearly describe the purpose of your validation. Validation goals may include a demonstration of the feasibility of an approach; a comparison of two approaches; a formal proof of correctness; a formal proof of the efficiency of an approach.
- *Validation Object*: The validation object is the approach/technique/method, which is validated.
- *Reference to the integration framework*: The validation object should be well related to the integration framework. This link should be established by relating the validation object to the "three pictures" of the framework. Consequently, the validation object should be related to the framework element or framework elements in case of a cross-cutting technique; to the life cycle model and to the architectural model. If you feel that no link can be established in a natural way, please contact the IA-3.2 workpackage leader to discuss whether the framework needs a modification.
- *Scenarios*: This part contains the unique identifiers of the validation scenario(s) used. If necessary, additional information can be included on how these scenarios are used and/or to describe any unusual use of the scenario.
- *Validation Type*: The validation type or research method describes the technique used for validation. These techniques may range from empirical ones such as questionnaires, case studies, field studies, action research, and laboratory experiments to formal methods such as verification, formal proofs or prototyping. If a validation type is not frequently used in the S-Cube reference disciplines, it should be explained, e. g., by citing appropriate literature describing the validation type.
- *Organisational setting*: The organisational setting describes the environment (in its broadest sense) in which the validation was performed. This description differs heavily according to the validation type. For laboratory experiments, the design of the experiment, the experimental groups, the participants and the materials should be described. For action research, the company, its departments and the people involved should be described as well as the influence of the researcher on the validation. For tool experiments, the tools used, the infrastructures and the computational platform (power of the processor, operating system, amount of main memory etc.) should be described. In case of prototyping the software fragments re-used should be described together with the abilities and limitations of this prototype.

- *Description*: The description section covers how the validation was performed. It should document the major steps to derive the validation result. In case of empirical research methods it should document how the experiment, case study or action research was performed. In case of prototyping the description should include for instance, the design of the prototype.
- *Results*: This section should document all relevant results achieved during the validation. It should contain the raw data, the statistical analysis, graphs/figures to visualise this data, the interpretation, the threads to validity (if applicable) as well as known limitations of the validation study.
- *Experiences*: Finally we are also interested in collecting experience reports in IA-3.2. Any information, which seems relevant to repeat this kind of validation should be reported most preferably in form of a Dos and Don’ts list. This information include experiences with the particular research method, experiences in a particular organisational setting, experiences with a particular scenario and experiences with the validation object.

Table 33: Validation Plan Template

ID	Unique Identifier of the Validation Plan
Name	Name of the Validation Plan
Validation goals	Specify briefly what you intend to show with your validation and how this contributes to the goals and vision of the S-Cube project.
Validation Object	Approach, Technique or Method to be validated.
Reference to the Integration Framework	Relate the Approach, Method, Technique validated to the Integration Framework including: • Framework element(s) addressed. • Position in the lifecycle model • Position in the architecture model.
Scenarios	The IDs of the involved scenarios.
Validation type (research method)	E. g. Empirical, experimental, prototype, conceptual proof, etc.
Organisational Setting	Please explain the environment in which the validation was performed. This could for instance be an industrial setting (in this case explain the company, department, type of task under investigation), an empirical experiment (in this case describe the experimental groups, the participants, the materials used in the experiment) or a technical experiment (in this case explain the infrastructure and tools used, the communication channels etc.).
Description	Some information about the validation should be useful, for example for an experimental validation, the provided inputs could be reported.
Results	Report all your validation results, e. g. data sets acquired, statistical analysis of the data, tools and prototypes produced, interpretation of the results etc.
Experiences	Report your experiences gained when performing the particular validation type (research method). What were the major obstacles/problems when applying this type of validation? Why did you choose this type? What were, in you opinion, the major strengths of this approach?

4.2 Validation Plan Examples

Here we present the possible validation plan applied to one of the scenarios described by the S-Cube partners. This is a tentative representation: the concrete validation of the scenarios will be performed in the later phases of the project and will be reported in the upcoming deliverables of the S-Cube integration activity IA-3.2. Given that, the template is incomplete; its goal is merely to demonstrate the application of the presented approach to the materials identified by partners.

4.2.1 Validation Plan for Grid-based Simulation Scenarios

The following plan presents a potential validation of the scenario represented in Section 3.10.

Table 34: Exemplary Template of the Validation Plan for Grid-based Simulation Scenarios

ID	Validate-Crosslayer_Framework-1
Name	Validation of Cross-layer Monitoring and Adaptation Framework for a Wrong Diagnosis Case during SBA Monitoring
Validation goals	This validation demonstrates the necessity of cross-layer monitoring in a scenario where adaptation requirement is wrongly identified due to the lack of complete and correct diagnosis of the monitoring data. It aims to address the research challenge "cross-layer monitoring and adaptation", mentioned in S-Cube's global research vision.
Validation Object	Cross-layer Monitoring and Adaptation Framework
Reference to the Integration Framework	• Framework elements addressed: Service Adaptation and Monitoring (SAM) • Position in the lifecycle model: Identify Adaptation Needs • Position in the architecture model: Adaptation Engine
Scenarios	Auto-Crosslayer-1
Validation type (research method)	Empirical, case study
Organisational Setting	Prototyping + technical experiment: The prototype of the cross-layer monitoring engine is tested via a case study.
Description	For doing the technical experiment, the SBA that is used for the case study is implemented on a BPEL engine and the prototype of the cross-layer monitoring engine is integrated with the BPEL engine. To compare the proposed engine with the traditional monitoring engines, another experimental set-up is prepared where a well-known state-of-the-art monitoring engine is picked up for the case study. At next step, the necessary monitoring data are given as input to the both engines and the results are observed.
Results	NA (will be reported when the results are obtained)
Experiences	NA (will be reported when the results are obtained)

4.2.2 Validation Plan for Collaborative Transport Chain Control Scenario

Table 35: Validation Plan for the Collaborative Transport Chain Control Scenario

ID	
Name	Validation of process migration
Validation goals	The goal of the validation is to show that the probability of the successful execution of a service-based process (~ service composition) in mobile environments can be enhanced by the concept of process migration
Validation object	Process migration
Reference to the	The validation object contributes to the logical run-time architecture of

Integration Framework	WP-JRA-2.2, i.e., to the element in life-cycle: operation & management (service composition execution), in providing a mechanism for cooperative execution of service compositions in mobile environments.
Scenarios	WINERY-S-1
Validation type	Formal proof with analytical assumptions which has to be confirmed by practical experiments with the scenario executed on a prototype
Organizational Setting	The prototypical evaluation was executed with a prototype of DEMAC (Distributed Environment for Mobility-Aware Computing) mobile process engine and context management system, a middleware which realizes the concept of process migration.
Description	Figure 31 shows the exemplary network infrastructure of a transshipment centre for container traffic, where a freezer container is monitored by a wireless sensor. In case of a malfunction of the cooling system, the wireless sensor instantiates a predefined mobile process template which specifies reactions to the detected situation. The resulting process instance is depicted in Figure 32: Mobile process example, showing a selected set of abstract activities and their input/output data: First, the current position of the container has to be acquired (*Get Position*). Second, the estimated time of arrival has to be calculated in order to decide whether the cargo will thaw until the container arrives (*Calculate Time of Arrival*). If the time until arrival is considered too long (*Time > X*), the container must be redirected, e.g. to a cold storage (*Redirect Container*). Furthermore, a message has to be generated to inform maintenance support where to find the defect container (*Send Message*). The last two activities of the process are realized as a transaction because the engineer will probably not be able to find the container without knowledge about its new destination. Therefore, the redirection has to be undone if the message cannot be sent within a specified deadline. Furthermore, the message's information must be transferred encrypted – which is attached as a non-functional requirement. Due to performance restrictions, the wireless sensor is not able to execute the process itself. As the process will therefore leave the sensor's sphere of control, it attaches a management descriptor which holds rules about its recovery, monitoring and logging requirements. In this use case, the management descriptor specifies that process execution should be monitored by a backup-device and that, in any case of irregularity, the process should be restarted by this device. Furthermore, process participants, failing devices and recovery actions should be logged and failing devices should be excluded from further process execution. If applicable, a context-based look-ahead procedure should be used to find the most appropriate migration path in order to avoid unnecessary migrations. A possible execution path of the mobile process is shown by the numbered arrows in Figure 31: Transport logistic network. The wireless sensor is not able to calculate a temporarily optimal execution strategy for the process. Therefore, it migrates the process to an arbitrary other device in its communication range, in this case to wireless Controller A (step 1). But Controller A has a malfunction and is not able to call any other service to execute the process. A timeout indicates its failure and the process is restarted. As this incident is also logged, the failing device is avoided during upcoming migrations and, consequently, in the second attempt Controller B is selected (step 2). As this controller is a stationary and quite powerful device, it is able to call a nearby GPS service to

	collect data about its current position as well as to calculate the estimated time to arrive at the container's destination (steps 3 and 4). Furthermore, it can decide about the necessity of redirection and uses its own local service to unload the container. However, as it is not connected to the Internet, it has to use an intermediary device to call an appropriate e-mail service. The message is therefore encrypted as described above. Furthermore, as the use of the network (e.g. UMTS) causes telephone charges, participant and payment details can be logged to the mobile process and can be refunded later. **1. Formal proof of advantage of process migration** To discuss the advantage of process migration, we examine a stochastic model, where *p* denotes the probability of a single device being capable of executing the current task (locally or remotely), *q* denoting the probability of process migration and *n* representing the number of hops caused by the migration. The successful execution probability for a migrating process can be calculated as a converging geometric series of the likelihood of successful sub-task execution anywhere in the mobile vicinity (cp. Figure 28). **2. Experimental evaluation with a prototype** Beside the formal proof, we evaluated the applicability of the generic context model and process management system with a prototype implementation realized in the DEMAC (Distributed Environment for Mobility-Aware Computing) project (cp. Kunze et al. 2008, pp. 467-469). The evaluation includes an experiment to determine the probability of successful execution. The environment for the experimental series consists of a simple process with one single activity, six heterogeneous devices with two devices having the capability to execute the processes' activity, and four devices unable to do so. Because sender and receiver of the mobile process cannot be the same, there are 5 possibilities for each process to migrate from one device to another. This leads to an execution probability of *p*=40% within the entire system. To test the behavior of the prototype under load, several test runs have to be carried out, each including 100 processes.
Results	**1. Formal proof of advantage of process migration** Some exemplary values calculated are presented in Figure 29, showing the probabilities of successful process execution with exemplary migration probabilities of *q*=0%, *q*=20% *q*=60% and *q*=88%, while *p* is assumed to be constantly equal to 40%. As to see, the estimated probability of a successful execution increases considerably already after a few hops, especially if there is a high heterogeneity and thus a high migration probability. **2. Experimental evaluation with a prototype** Figure 30 shows the average number of hops resulting from migrations necessary to execute the process successfully compared to the expected analytical value. The analysis of the experiments further shows that only a few hops suffice to increase the probability of successful execution to levels more than twice as high. The estimated probability and the applicability of the presented concept can therefore also be confirmed by practical experimentation.
Experiences	

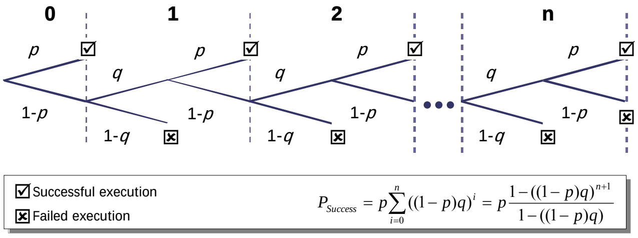

Figure 28: Probability tree of successful mobile process execution (Kunze et al. 2008, p. 461)

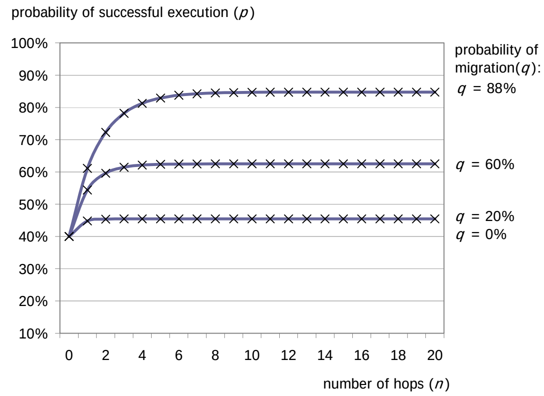

Figure 29: Execution probability of process migration variants (p=40%)

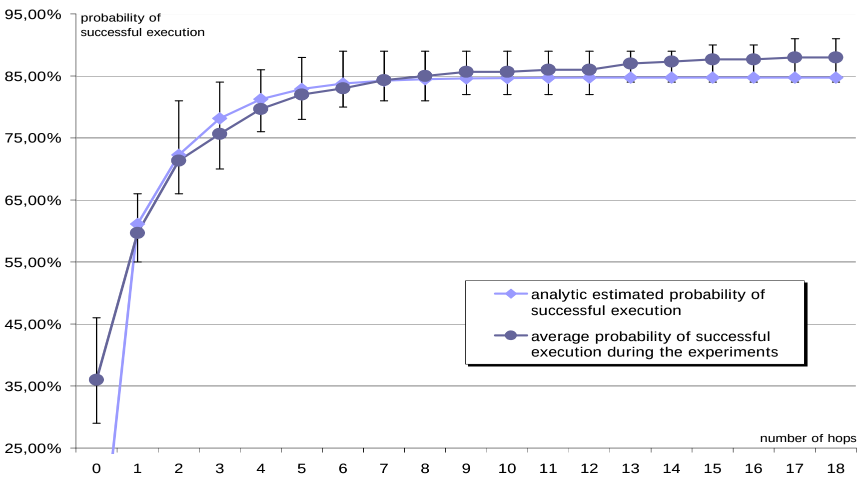

Figure 30: Results of the experimental evaluation (Kunze et al. 2008, p. 469)

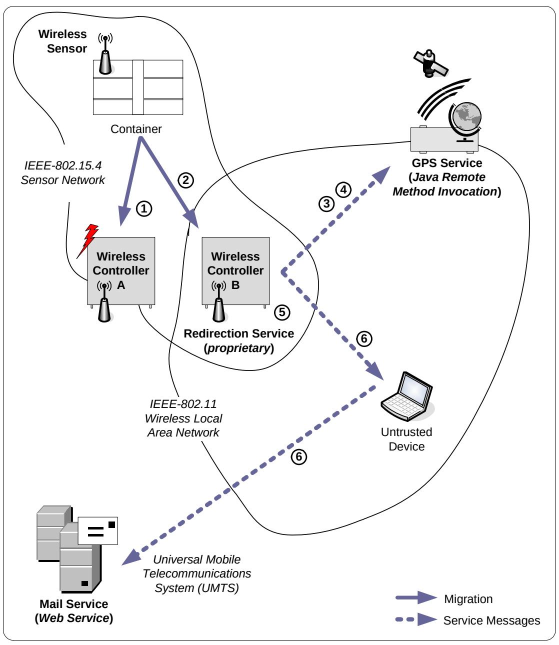

Figure 31: Transport logistic network

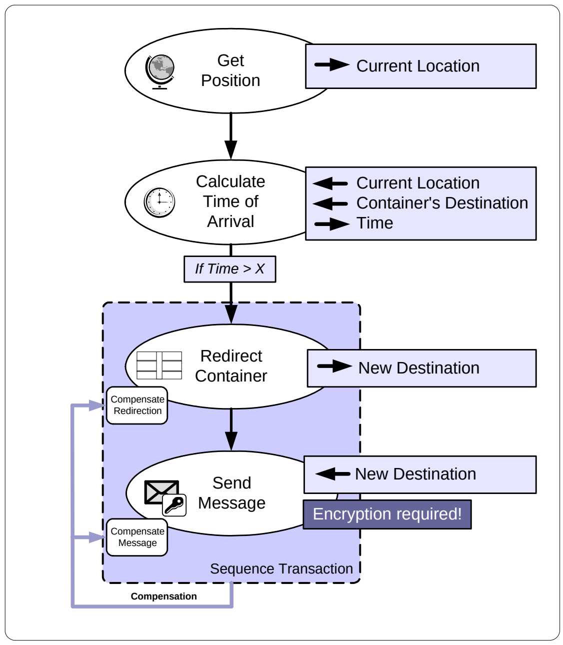

Figure 32: Mobile process example

5 Conclusions

In this deliverable, we have reported the guidelines for documenting scenarios including their link to the IA-2.2 case studies. These scenarios are will be used to validate the S-Cube research framework and to support the validation of the future research results. In addition, to the guidelines we have collected an initial set of scenarios based on the existing S-Cube materials produced in the JRA-1 and JRA-2 activities. This initial set of scenarios will be extended throughout the entire life of the S-Cube project. Furthermore, we have demonstrated by the means of one example how the scenarios will be used to validate S-Cube results.

The next step in work-package IA-3.2 is to exploit the collected scenario to perform the validation of the integrated framework (Deliverable CD-IA-3.2.2, month 24). This will provide suggestions for improvements to the definition of the integration framework and to the research activities in JRA-1 and JRA-2. It will also provide feedback to the validation activities themselves, e. g., in terms of coverage of the integrated framework by the collected scenarios; this feedback will be used to drive the future activities concerning the collection of scenarios and the definition of validation plans.

Finally, as the collection of the validation scenario – and, more in general, the validation of the integrated framework – is an activity that will occur throughout the full life of the project, we are planning to make the scenarios available through the S-Cube web portal. The structured description of the scenarios will help S-Cube members to find scenarios for their individual validation tasks. This way, scenarios and results become a live corpus, which is extended and revised in a continuous way, as part of the research activities undertaken by the partners within S-Cube.

References

[1] C. Rolland, C. B. Achour, C. Cauvet, J. Ralyté, A. Sutcliffe, N. Maiden, M. Jarke, P. Haumer, K. Pohl, E. Dubois, and P. Heymans, "A Proposal for a Scenario Classification Framework," *Requirements Engineering,* vol. 3, pp. 23-47, 1998.

[2] K. Pohl, *Requirements Engineering - Grundlagen, Prinzipien, Techniken*, 1st edition ed. Heidelberg: dpunkt.verlag, 2007.

[3] G. Sindre and A. L. Opdahl, "Eliciting security requirements with misuse cases," *Requirements Engineering,* vol. 10, pp. 34-44, 2005.

[4] E. D. Nitto, C. Ghezzi, A. Metzger, M. Papazoglou, and K. Pohl, "A Journey to Highly Dynamic, Self-Adaptive Service-Based Applications," *Automated Software Engineering,* pp. 257-402, 2008.

[5] Object Management Group: "UML 2.0 Superstructure Specification". OMG document: ptc/03-08-02, 2003.

[6] "StarUML - The Open Source UML/MDA Platform". Website: http://staruml.sourceforge.net/en/.

[7] POLIMI; Tilburg; CITY; FBK; UniDue: "Collection of Industrial Best Practices, Scenarios and Business Cases", S-Cube Deliverable CD-IA-2.2.2, 2009.

[8] Gehlert, A.; Hielscher, J.; Danylevych, O.; Karastoyanova, D.: "Online Testing, Requirements Engineering and Service Faults as Drivers for Adapting Service Compositions". In: Karastoyanova, D.; Kazhamiakin, R.; Metzger, A.; Pistore, M. (Eds.): Proceedings of the International Workshop on Service Monitoring, Adaptation and Beyond (MONA+ 2008), December 13, 2008, Madrid, Spain, 2008, p. 39–50.

[9] The S-Cube Team: "S-Cube's Global Research Vision, Version 1.0 – April 2009", 2009

[10] The S-Cube Team: "CD-IA-3.1.2 – Integration Framework Baseline", 2009.