

D1.3.1 Industrial Use Cases for French Consortium

ModelWriter

Text & Model-Synchronized Document Engineering Platform

Project number: ITEA 2 13028

Work Package: WP1

Task: T1.3 – Industrial Use Cases for French Consortium

Edited by:

Anne Monceaux (AIRBUS Group)

Marwa Rostren (Obeo)

Date: 31-Jan-2015

Version: 1.0.0

Apart from the deliverables which are defined as public information in the Project Cooperation Agreement (PCA), unless otherwise specified by the consortium, this document will be treated as strictly confidential.

Document History

Version	Author(s)	Date	Remarks
Version 1.0	Anne Monceaux Marwa Rostren	31/01/2015	Initial release

Table of Contents

DOCUMENT HISTORY	2
1 INTRODUCTION.....	4
1.1 ROLE OF THE DELIVERABLE	4
1.2 STRUCTURE OF THE DOCUMENT	4
1.3 TERMS, ABBREVIATIONS AND DEFINITIONS	4
2 USE CASES DESCRIPTION	5
2.1 UC-FR1- SYNCHRONIZATION BETWEEN MODELS AND DOCUMENTATION (OBEO - SIRIUS PRODUCT).....	6
2.1.1 SIRIUS: The team constraints.....	7
2.1.2 SIRIUS: The software and its source code	7
2.1.3 SIRIUS: The used models and the documents.....	8
2.1.4 SIRIUS: The documentation synchronization examples	8
2.1.5 SIRIUS: The UC synthesis	10
2.2 UC-FR2- ENTERPRISE ARCHITECTURE (OBEO - SMARTEA PRODUCT).....	12
1- SMARTEA: The team constraints	12
2- SMARTEA: The software and its source code.....	13
3- SMARTEA: The used models and the documents	13
4- SMARTEA: The use case Examples.....	15
5- SMARTEA: The UC synthesis.....	17
2.3 UC-FR3- TITLE (SUBJECT OF A REQUEST CHANGE)	19
2.4 UC-FR4- SYNCHRONIZATION OF REGULATION DOCUMENTATION WITH A DESIGN RULE REPOSITORY (OBEO + AIRBUS GROUP + LORIA).....	19
2.4.1 The context and teams constraints.....	19
2.4.2 The SIDP documents.....	19
2.4.3 Current SIDP-to-DB scenario.....	20
2.4.4 Refinement of the scope of the use case	22
2.4.5 The UC synthesis.....	22
2.5 UC-FR5- PRODUCTION OF A CONTEXT SPECIFIC DESIGN DOCUMENT (OBEO + AIRBUS GROUP + LORIA).....	23
2.5.1 The context and teams constraints.....	23
2.5.2 Representing the design task context	24
2.5.3 Refined scope of the use case.....	24
3 CONCLUSIONS & WAY FORWARD	26
ANNEX 1.....	26

1 Introduction

1.1 Role of the deliverable

This document is the first version of the description of the use cases proposed by the French consortium. It may be up-dated depending on the further details and requirements we get from our industrial use case providers.

1.2 Structure of the document

This document is organized as follows:

- Chapter 1 introduces the document.
- Chapter 2 describes for each use case: the scope and motivation, the approach and the available resources (corpora).
- Annex 1 lists for each use case the annex documents and associated data deliverables, so called “corpora”:
 - o D1.2.2 Public corpora, and
 - o D1.2.3 Private Corpora.

1.3 Terms, abbreviations and definitions

Abbreviation	Definition
A/C	Aircraft
ATA	Air Transport Association
DB	Data base
DP	Design Principles
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
SIDP	System Installation Design Principle

2 Use cases description

The use cases are provided by Obeo and Airbus Group companies.

OBEO addresses the work package WP1 by providing at the first time its own critical need to gain time and quality of its own products like the Sirius product's documentations; at the second time, OBEO describes and defines its customers need to synchronize and collaborate to update their own documentations while using the OBEO's SMARTEA solution.

In the first and second sections, OBEO describes and defines Sirius (UC-FR1) and SMARTEA (UC-FR2) products, its existing documentation life cycles and its requirements for each.

The definition of the foreseen UC-FR3 is still pending – we will decide how to address it in a second iteration of this document.

The two Use Cases UC-FR4 and UC-FR5 proposed by AIRBUS Group are respectively described in sections 2.4 and 2.5. These two use cases are closely connected: they both focus on specific kinds of documents called SIDP (e.g. System Installation Design Principles) and on the models that these documents are using or referring to. We call “rules” the installation design principles currently described in an unstructured way inside the source documents.

The connection between the two cases is illustrated by Figure 0 1 below. Basically UC-FR4 focuses on analysing the SIDP documents and identifying rules elements inside to help the building of a rule database. The second UC-FR5 uses the formalized rules together with the formalized “visual” model elements to identify the information that is relevant for a given design context. The design context will be represented as exemplary queries that may involve reasoning.

The overall driving need for these two Use Cases is to reduce the time and the burden for the designers to consult a large set of regulation documents in order to retrieve design rules. Due to reasons such as technology push, process changes, etc., an increasing number of different regulation documents are issued by different stakeholders. They contain a high number of informal rules and the designers have difficulties to follow the information cascade and retrieve or rebuild the right information. This situation results in time waste, suboptimal designs and higher risks of error.

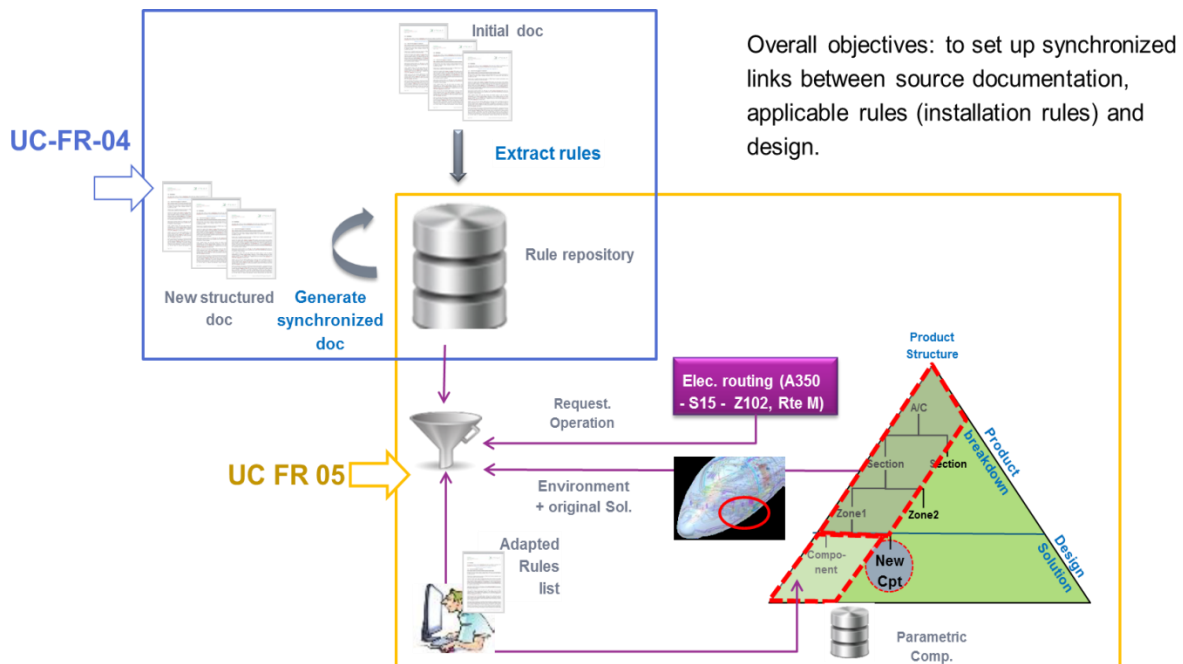


Figure 2-1 UC-FR-04 + UC-FR-05 common context and interface

2.1 UC-FR1- Synchronization between Models and Documentation (OBE0 - Sirius Product)

OBE0's activity is based on its own software's like Sirius (<http://eclipse.org/sirius/>). Sirius documentation is the first important Use Case for OBE0 to prove the validity of the "Text & Model – Synchronized Document Engineering" approach of ModelWriter.

In this software, models are directly used to generate a part of the product code. The Sirius team has a seriously hard task to properly update the documentation to keep it as much as possible synchronous when the product evolves:

- The specification,
- The Web page documentation content,
- The Developer Manual
- The Specifier Manual
- The User Manual, tutorials and Cheat-sheets
- The release Notes
- The Javadoc

Very often, this documentation got completely asynchronous because of adding, deleting, changing or replacing concepts in models or in source code. Systematically, for each issue and each release, developers and architects did not trust the existing documentation content and spent a lot of time on searching the asynchronous parts, updating and re-writing it.

Today, to keep the documentation as much as possible synchronous with the current product state, the Sirius team spends time to analyse new tasks impacts on the existing documents. Therefore the documentation synchronization is done manually and is actually considered as a part of their workflow; their task's schedules include specific time to locate and to update the

concerned impacted parts for every issue but that specific time is not really accurate and depends on every task impacts. Consequently, the team is not able to evaluate a real ratio of the time spent to synchronize the documentation.

The team has set up several optimization measures to reduce the time spent on synchronizing the documentation. They have centralized the complete documentation in a single eclipse project to facilitate its maintenance. The current documentation covers all the Sirius's features but the team is depriving the creation of new and more focused one to avoid expensive duplications.

Consequently, the current documentation is:

- Too long to be considered as start-up documentation for users.
- Not detailed enough to be considered as a complete reference for developers and for contributors; it does not contain technical tricks and does not contain all the best practices. The team members exchange this information using skype or during their internal meetings. Inevitably, this information are lost or forgotten after a while but the team does not have the time today to write and especially they do not have the time to maintain specific documentation containing this kind of information.

In addition, today the existing Javadoc is not trusty; it was not updated since a while.

The OBEO aim objective can be expressed by proving that the integration of ModelWriter Tool to manage Sirius's documentation allows gaining in productivity and in documentation consistency since the documentation should not contain outdated information. Thanks to ModelWriter, the Sirius team should never have to re-write the documentation because of inconsistencies and will focus on documenting only asynchronous parts after implementing improvements or new features. The Sirius team should be able to write all documentation they need with any granularity level they want regardless duplications.

2.1.1 SIRIUS: The team constraints

- We need to keep documenting Sirius in an eclipse editor (using Mylyn textile).
- We need to keep the same documentation engine working (ant scripts, toc generators, etc.).
- We need a synchronization tool assist to facilitate the documentation management and follow the product life cycle.
- We won't any dependency to non-eclipse application, non-open source tool.
- We won't any explicit links to the model(s) in the documentation body. It makes documentation unreadable.
- If the ModelWriter is a tool like Check-Style, we need to filter warnings, errors and keep the filter information ON (keep filter information).

2.1.2 SIRIUS: The software and its source code

Sirius is open source software. It can be downloaded for free using the next link: <http://eclipse.org/sirius/download.html> or by installing the last free version of OBEO Designer: <http://www.obeodesigner.com/>

The source code is available by cloning the repository:

<http://git.eclipse.org/gitroot/sirius/org.eclipse.sirius.git>

2.1.3 SIRIUS: The used models and the documents

The Sirius models are dispatched in several *.ecore models listed below:

- diagram.ecore – used to manage diagrams representations concepts. This model can be found in the “model” folder in the “org.eclipse.sirius.diagram” plugin.
- layoutdata.ecore – used to manage layoutdata concepts. This model can be found in the “model” folder in the “org.eclipse.sirius.diagram.layoutdata” plugin.
- sequence.ecore – used to manage sequences representations concepts. This model can be found in the “model” folder in the “org.eclipse.sirius.diagram.sequence” plugin.
- table.ecore – used to manage tables representations concepts. This model can be found in the “model” folder in the “org.eclipse.sirius.table” plugin.
- tree.ecore – used to manage trees representations concepts. This model can be found in the “model” folder in the “org.eclipse.sirius.tree” plugin.
- basicfamily.ecore – used to explain scenarios in the startup tutorial. This model can be found in the “model” folder in the “org.eclipse.sirius.sample.basicfamily” plugin.

Any change in one or more of these models provides asynchronous parts in the documentation.

Note that for the Sirius product, the software source code can be considered as the real model. Its modification provides asynchronous parts in the documentation too and Sirius Team needs today to check manually the documentation after every change to keep it updated.

The Sirius software’s documentation is centralized in the “**org.eclipse.sirius.doc**” plugin available by cloning the same repository as for the source code:

<http://git.eclipse.org/gitroot/sirius/org.eclipse.sirius.git>.

This documentation does not cover all the models artefacts but it covers the main concepts, scenarios and features.

To check all modifications done by Sirius Team on the existing documentation before every release you can compare the plugin status to the last release tag status.

2.1.4 SIRIUS: The documentation synchronization examples

1. Add a New Feature

An example of documentation update caused by adding a new feature: readers can refer to the next commit.

```
commit 384fb01d4306d2709dd3c40d7b19ad2a2bb29ca9
Author: Laurent Redor <laurent.redor@obeo.fr> 2014-08-05 10:38:21
Committer: Laurent Redor <laurent.redor@obeo.fr> 2014-08-22 09:32:51
Parent:      49befdb3fb46deb246b5f7331764b11116895c57      ([442231] Have
InvalidPermissionCommand/LockedInstanceException manages several EObjects)
```



```

Child: 707fcaa0430a16afb22db2c80ccfc1bbea0ad4c8 ([427872]
AbstractDeleteDRepresentationElementTask and subclasses cleanup)
Branches: master, origin/bug/cleanupTests, origin/master,
origin/tests.performance, origin/v2.0.x

[441090] Resize a container without modifying the contained elements
location

* SiriusResizeTracker: Override of {@link ResizeTracker} to allow a
resize that also moves all children.
* AirResizableEditPolicy:
** Use our own ResizeTracker to set the "flag"
SiriusResizeTracker.FIX_CHILDREN_KEY when the corresponding shortcut is
pressed.
** Adapt buildResizeCommand to also resize children (according to
SiriusResizeTracker.FIX_CHILDREN_KEY)
* MoveViewOperation (and InstanceRoleResizableEditPolicy) : Move the
existing MoveViewOperation from sequence to diagram.

Bug: 441090
Change-Id: I5f603e7c1aa5abe74e49b6c5c2325e2efe86801d
Signed-off-by: Laurent Redor <laurent.redor@obeo.fr>

```

The commit concerns a new feature: “**Resize a container without modifying the contained elements**”. By consulting the commit changes, the reader can notice that the same commit contains code changes and documentation updates. The “**SiriusResizeTracker.java**” class was created and used to manage the container’s resize. This new implementation makes asynchronous the current documentation. That’s why we invite you to focus on:

- The “**org.eclipse.sirius.doc/doc/user/diagrams/Diagrams.textile**” file in which the new behavior is described in a new section “**Resizing elements**”.
- The “**org.eclipse.sirius.doc/specs/proposal/441090.textile**” file in which the specification of the improvement is detailed.

2. Improve an Existing Feature

An example of documentation update caused by improving an existing feature: readers can refer to the next commit.

```

commit 840ebe0d0e69111d507b174088f0cf0d35145c5b
Author: Laurent Redor <laurent.redor@obeo.fr> 2014-07-02 10:29:57
Committer: Laurent Redor <laurent.redor@obeo.fr> 2014-07-04 11:41:54
Parent: 9893193aa139c428d72acf9589e1b2a4f49aae9e ([437528] Regen the spec
html file.)
Child: f52c1a387b1e8e0c5476676437738d5b62eb4865 (Merge branch
'bug/435507_SnapToGridForCreation')
Branches: master, origin/bug/cleanupTests, origin/gsoc2014, origin/master,
origin/performances, origin/tests.performance, origin/v2.0.x

[438691] Add "touched" mode for selection with rectangle

```

Currently, you can select several elements by using a selection rectangle. To be selected, an element must be completely contained by the selection rectangle.

This commit adds a new mode ("touched" mode), in which to be selected, an element must intersect the selection rectangle.

This new mode will be activated when the user selects elements from right to left. The current mode remains when the user selects elements from left to right.

[Bug: 438691](#)

Change-Id: I3f376bc9b2292cf56835338c8a3ae9a0140bc74d

Signed-off-by: Laurent Redor <laurent.redor@obeo.fr>

The commit concerns improving an existing feature: **"Selection with rectangle"**. By consulting the commit changes, the reader can notice that the same commit contains code changes and documentation updates. The **"RubberbandSelectionTool.java"** class was modified and used to manage the **"touched"** mode. This new implementation makes asynchronous the current documentation. That's why we invite you to focus on the **"org.eclipse.sirius.doc/doc/user/diagrams/Diagrams.textile"** file in which the behaviour of **"rectangle's selection"** explanation is replaced by the new behaviour after the change.

2.1.5 SIRIUS: The UC synthesis

Use Case ID:	UC-FR1		
Use Case Name:	Synchronization between Models and Documentation		
Created By:	OBEO	Last Updated By:	OBEO
Date Created:	30/01/2015	Date Last Updated:	31/01/2015

Primary actor:	Sirius Team
Secondary actors:	Sirius Contributors
Description:	<p>Today, the actor must analyze new tasks impacts on the existing documents. Documentation synchronization is done manually. Documentation synchronization is actually considered as a part of the Sirius tasks workflow. Task's schedules must include specific time to locate and to update the concerned impacted parts for every issue. This specific time depends on every task impacts and is not really accurate. The complete documentation is centralized in a single eclipse project to facilitate its maintenance.</p> <p>Limitations:</p> <ul style="list-style-type: none"> - The current documentation covers all The Sirius's features but the team is depriving the creation of new and more focused one to avoid expensive duplications.

	<ul style="list-style-type: none"> - The current documentation is too long to be considered as startup documentation for users. - The current documentation is Not detailed enough to be considered as a complete reference for developers and for contributors; it does not contain technical tricks and does not contain all the best practices. The team members exchange this information using skype or during their internal meetings. Inevitably, this information are lost or forgotten after a while but the team does not have the time today to write and especially they do not have the time to maintain specific documentation containing this kind of information.
Preconditions:	<ul style="list-style-type: none"> 1- Introduce a new feature or improve an existing feature in the models or in the source code. 2- Existing documentation contains the improved feature or existing documentation does not contain the new feature or the improved feature. 3- Edit, add or remove a concept from the documentation.
Postconditions:	<ul style="list-style-type: none"> 1- Concerned documentation contains "markers" e.g. like warnings if the concept was improved, error if the concept is deleted. 2- Models and source code must contain markers too e.g. like warnings on each improved concept, errors on each new undocumented concept. 3- The user must be able to deactivate a marker; in this case the linked concept will never be synchronized again in the concerned context.
Normal Course of Events:	<ul style="list-style-type: none"> 1- Introduce a new Feature or improve an existing one. 2- Activate/run the ModelWriter which links models with documentation. 3- Visualize synchronization between the models and the existing linked documentation.
Alternative Paths:	
Exceptions:	
Special Requirements:	<p>Sirius Team Requirements:</p> <ul style="list-style-type: none"> 1- Keep using Mylyn textile for documentation 2- Keep the same documentation engine working (ant scripts, toc generators, etc.) 3- ModelWriter must be a synchronization tool assist to facilitate the documentation management and follow the product life cycle. 4- Won't any dependency to non-eclipse application, non-open source tool. 5- Won't any explicit links to the model(s) in the documentation body. It makes documentation unreadable. 6- If the ModelWriter is a tool like Check-Style, we need to filter warnings, errors and keep the filter information ON during the documentation life cycle (keep filter information).
Assumptions:	The models are either of type: Ecore models or Java code.
Notes and Issues:	<p>Thanks to ModelWriter:</p> <ul style="list-style-type: none"> 1- No need to analyze the new tasks impacts on the existing documents

	2- Documentation synchronization will be done automatically 3- Tasks' schedules will just include time to update the asynchronous and to add new documentation features 4- The team will be free to synchronize or not a concept/feature
--	--

2.2 UC-FR2- Enterprise Architecture (OBEO - SmartEA Product)

The OBEO's SmartEA solution for Enterprise Architecture brings together existing repositories and helping architects to design future architectures. The information systems of major companies and organizations are often extremely complex and made up of hundreds or even thousands of applications, databases, and servers distributed across multiple sites. These information system components are so inextricably linked and interconnected that it becomes extremely difficult to successfully implement changes within the company - be they strategic, organizational or digital - in order to adapt to evolving needs.

Today, SmartEA needs to provide documentation explaining the migration plans and explaining the Impact analysis to help the architect making transformations in a consistent and pragmatic way. In this context, and especially in the migration context, intentions behind decisions are as important as decisions themselves, and therefore so is the documentation. This documentation does not exist yet; today the migration plans are represented by a comparison model which compares the source architecture with the target one; the impact analysis can be obtained by deducing all the related artifacts, representations and references of all the model's elements.

In addition, Smartea Users need to provide their own project's documentation. This documentation is currently provided by using Acceleo Templates Artifacts which must be synchronous with the current state of project models.

The difficulty in the SmartEA context is not just related to the use of a specific kind of models, but in addition, these models and diagrams are stored remotely on a server, they are subject to authentication rules and can be edited in a collaborative mode manner. Thus, ModelWriter must be able to be stored remotely and must allow to collaboratively editing a document.

1- SMARTEA: The team constraints

- We need a new editor to facilitate documentation typing and synchronization with the diagrams, trees, and other models.
- We won't any code references in the body of the documentation.
- We need a collaborative documentation editing.
- We need a notification system about all documentations changes after editing.
- The ModelWriter must be easy to integrate to be a part of the SmartEA project.
- The ModelWriter must be able to be used on remote.
- We won't any models modification basing on documents modification.
- ModelWriter should allow activating/deactivating the synchronization direction "Text → Model" or "Model → Text" so SmartEA architects and developers can deactivate the "Text → model" synchronization option.

2- SMARTEA: The software and its source code

The SmartEA software is not an open source project. Thus the source code will not be shared with the ModelWriter project partners.

A Free trial is available on <http://www.obeosmartea.com/download> to discover the product. This version is limited in time and can be used about 3 months after the installation.

To learn how to use the SmartEA product you can refer to the following link:

<http://www.obeosmartea.com/product/online-demo> or you can refer to the detailed product's documentation available in the help of the SmartEA Rich client (eclipse RCP).

3- SMARTEA: The used models and the documents

All the diagrams, trees, tables and other artifacts presented in SmartEA are models conform to the TOGAF meta-model.

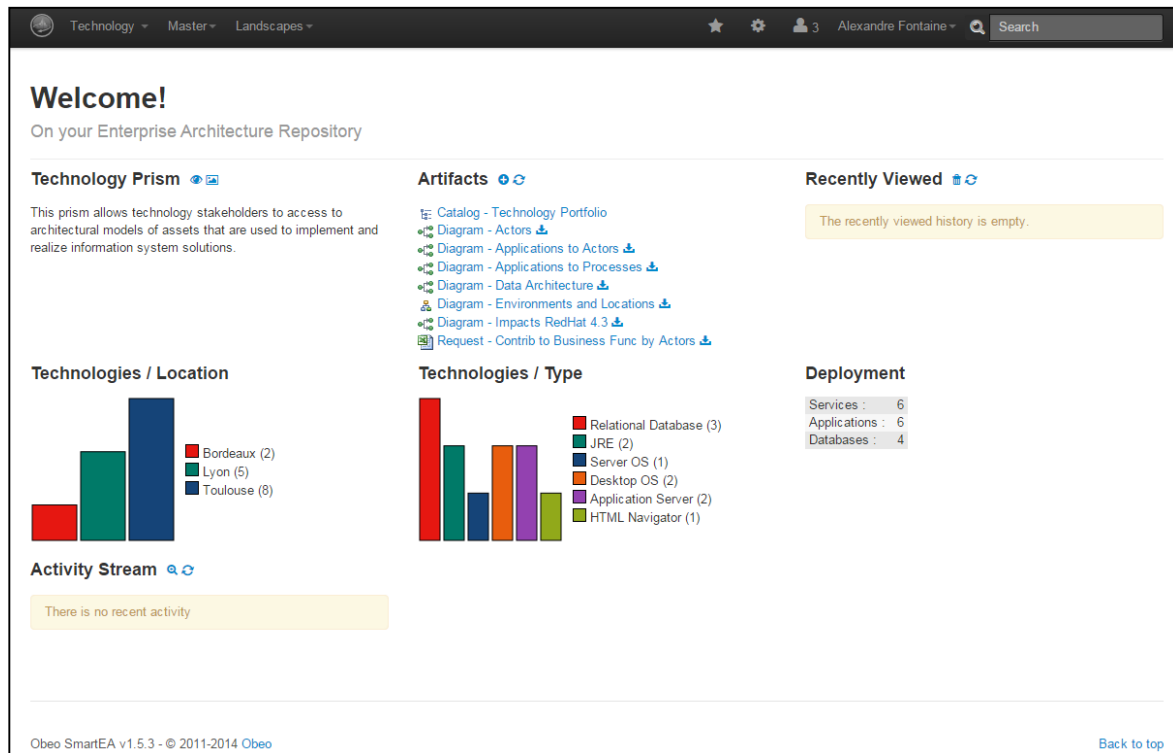
Today, the migration plans are not documented but models are available to indicate all modifications between source and target architectures to deduce the migration plans. The impact analyses are not documented neither but related artefacts, representations and references can be deduced automatically basing on given models.

In the rest of this document, we use the VOY example to illustrate the use case example. It is a Voyage Discount of the Travel Agency example provided by the SmartEA team's members basing on the Travel Discount provided by the CEISAR <http://www.ceisar.fr/>.

We assume that:

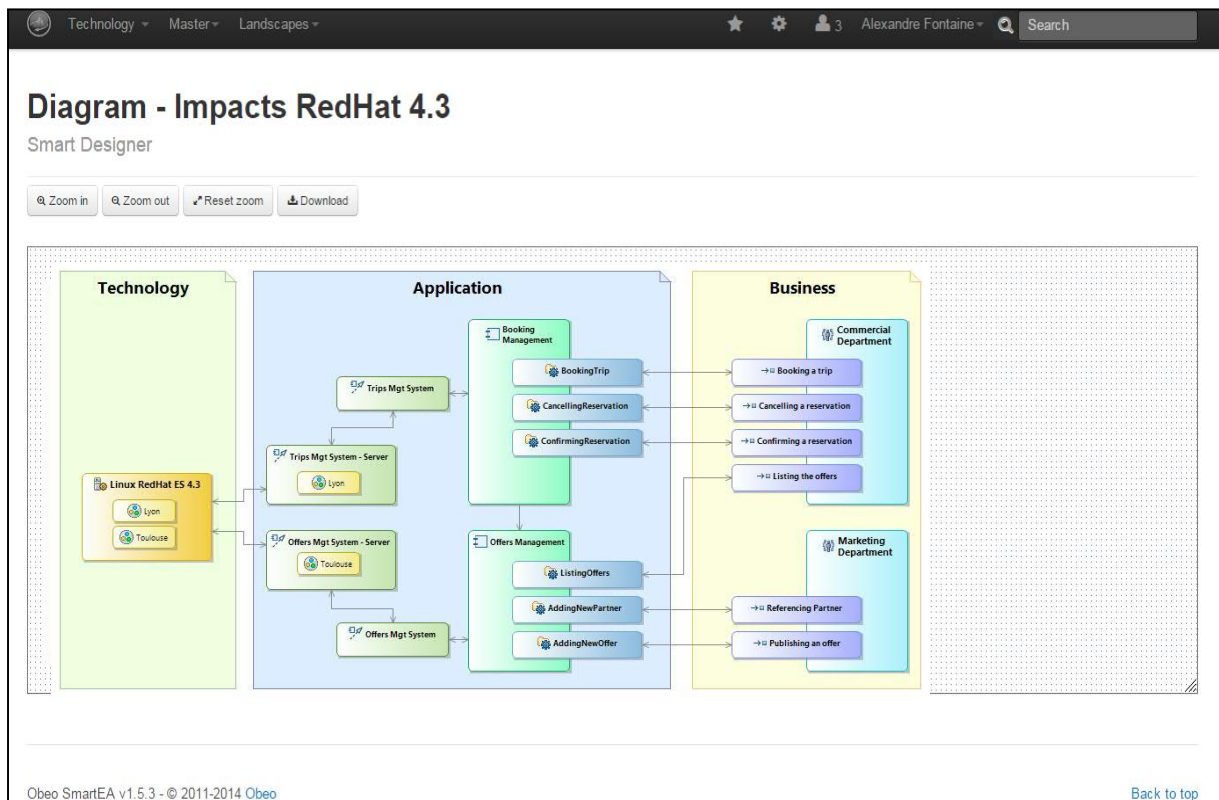
- you have already installed the trial version of SmartEA product,
- you have started the server,
- you are connected to: <http://localhost:9000/> as “afontaine” user, using the password “123”

The welcome page will be displayed as follows:



All the Voyage Discount models artefacts (catalogues, diagrams, trees, etc.) provided by the SmartEA team's members are available and their contents can be displayed by simple click.

e.g., a simple click on the **“Diagram – Impacts RedHat 4.3”** will display the impacts of RedHat 4.3 component diagram as follows:



4- SMARTEA: The use case Examples

This use case concerns the user documentation related to different artefacts and models. It concerns also all migration plans and impact analysis documentations.

Today, customers using SmartEA solution must know how to use ACCELEO modules to write, generate and to update their own documentation which are considered as specific artefacts. This is absolutely not the best solution to integrate documents to the SmartEA project life cycle.

We need to introduce a useful editor to facilitate the documentation typing and the documentation synchronization with the project models and diagrams.

The editor must also be able to manage collaborative features and must ensure users notification.

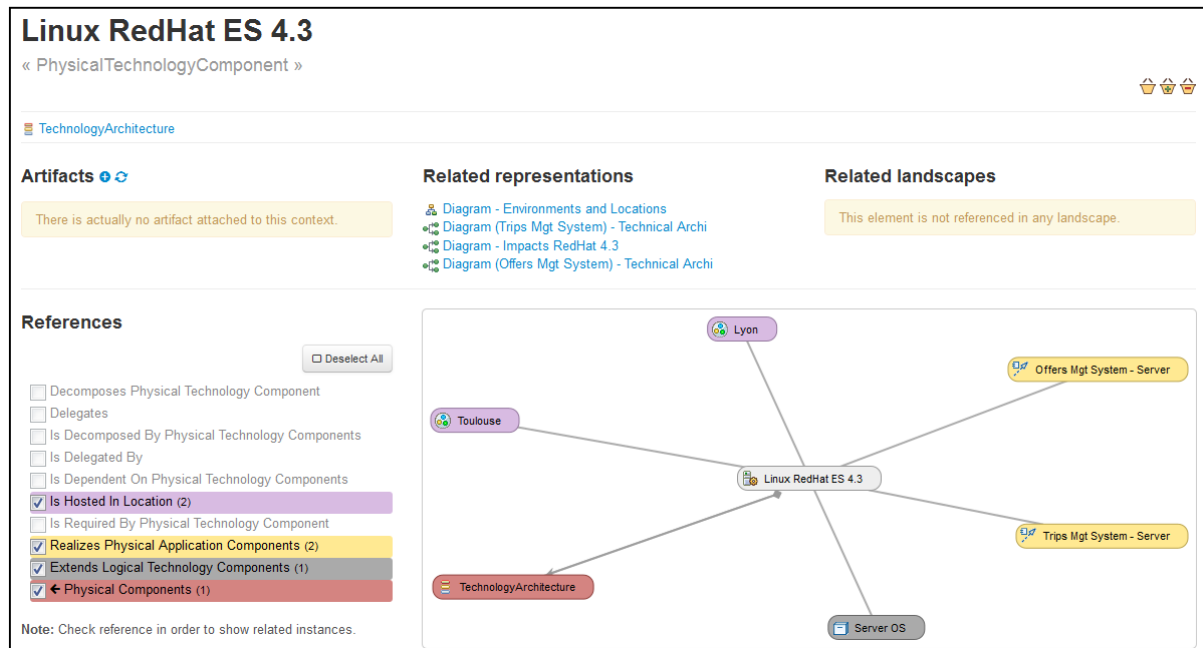
The **Annex 1 (D1.2.1-OBEO-UC-Annex1-SMARTEA.doc)** presents a document example related to the **VOY** Company example. The document contains the general architecture of the Company and a section focusing on the used technology replacement to establish the impact analysis and migration plans.

In SmartEA, the comparison report displays information about the architectures and about the components modifications. Here the Business, the Application and the Technology architectures are concerned by the current modifications.

Deleting the “**Linux RedHat ES 4.3**” component impacts several representations:

- The Environments and Locations diagram
- The Trips Mgt System – Technical Architecture diagram
- The Offers Mgt System – Technical Architecture diagram
- The Impacts RedHat 4.3 diagram

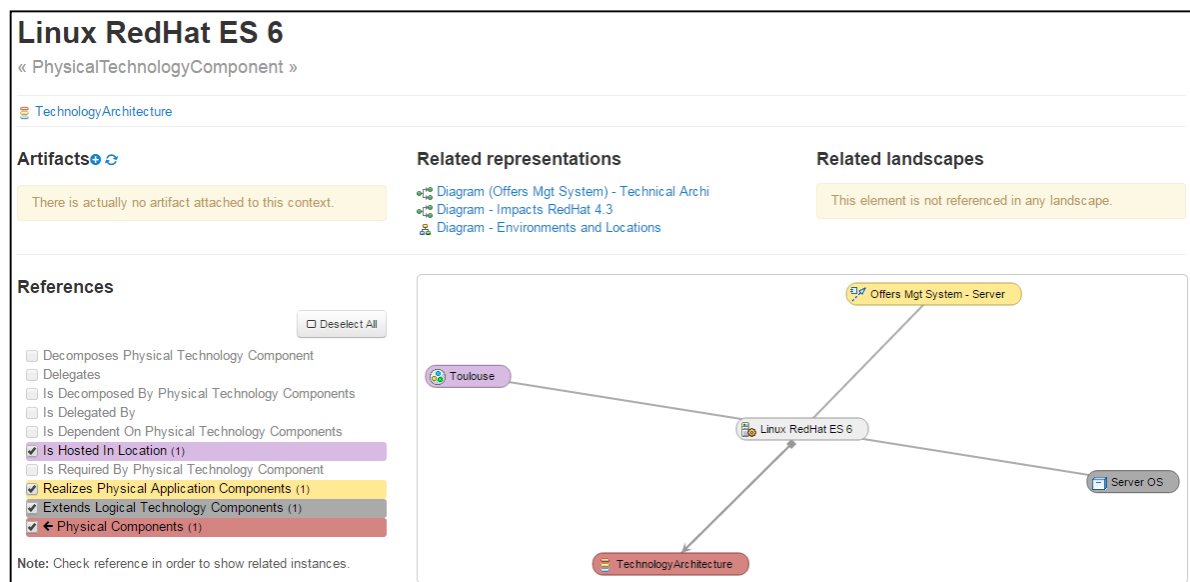
This information can be reached by displaying the related representations by double clicking on the “**Linux RedHat ES 4.3**” component of the tree above.



Adding the “**Linux RedHat ES 6**” component impacts several representations:

- The Environments and Locations diagram
- The Offers Mgt System – Technical Architecture diagram
- The Impacts RedHat 4.3 diagram

This information can be reached by displaying the related representations by double clicking on the “**Linux RedHat ES 6**” component of the tree above.



Combining all of this information together can help to produce the migration plan document to reach the new architecture from the existing one.

Thus, the migration plans can be entirely generated automatically. Thanks to ModelWriter, SmartEA should never have to regenerate the entire migration plans and will only focus on generating asynchronous parts after implementing improvements or new features and artefacts.

5- SMARTEA: The UC synthesis

Use Case ID:	UC-FR2		
Use Case Name:	Enterprise Architecture		
Created By:	OBEO	Last Updated By:	OBEO
Date Created:	30/01/2015	Date Last Updated:	31/01/2015

Primary actor:	SmartEA Users
Secondary actors:	SmartEA Team
Description:	<p>The OBEO's SmartEA solution for Enterprise Architecture brings together existing repositories and helping architects to design future architectures. The information systems of major companies and organizations are often extremely complex and made up of hundreds or even thousands of applications, databases, and servers distributed across multiple sites. These information system components are so inextricably linked and interconnected that it becomes extremely difficult to successfully implement changes within the company - be they strategic, organizational or digital - in order to adapt to evolving needs.</p> <p>Today, Smartea Users need to provide their own project's documentation. This documentation is currently provided automatically using Acceleo Templates Artifacts which must be synchronous with the current state of project models.</p> <p>The actor must have Acceleo skills to write, update and generate documentation. The actor must analyze the models modifications impacts on the Acceleo templates artifacts. The Acceleo artifacts synchronization is done manually. The documentation generation is launched manually. The documentation is generated automatically using the updated Acceleo templates.</p> <p>In addition, SmartEA needs to provide documentation explaining the migration plans and explaining the Impact analysis to help the architect making transformations in a consistent and pragmatic way. In this context, and especially in the migration context, intentions behind decisions are as important as decisions themselves, and therefore so is the documentation. This documentation does not exist yet; today the migration plans and the impact analysis are represented by models.</p>
Preconditions:	<ol style="list-style-type: none"> 1- Edit, add or remove a model 2- Existing documentation contains the improved model concepts or Existing documentation does not contain the new or the improved model concepts 3- Edit, add or remove a concept from the documentation
Postconditions:	<p>As for user documentation:</p> <ol style="list-style-type: none"> 1- Concerned documentation templates must contain "markers" e.g. like warnings if the concept was improved, errors if the concept is deleted 2- Models contain markers too e.g. like warnings on each improved concept, errors on each new undocumented concept 3- The user must be able to deactivate a specific marker, in this case the linked concept will never be synchronized again in the concerned context.

	<p>As for migration plans:</p> <ol style="list-style-type: none"> 1- Concerned migration and impacts analysis documentation must contain "markers " e.g. like warnings if the concept was improved, errors if the concept is deleted 2- Models contain markers too e.g. like warnings on each improved concept, errors on each new undocumented concept 3- The user must be able to decline a marker; in this case the linked concept will never be synchronized again in the concerned context.
Normal Course of Events:	<ol style="list-style-type: none"> 1- Introduce a new model or improve an existing one 2- Activate/run the ModelWriter which links models with documentation 3- Visualize synchronization between the models and the existing linked documentation
Alternative Paths:	
Exceptions:	
Special Requirements:	<p>SmartEA Team Requirements</p> <ol style="list-style-type: none"> 1- Need a new editor to facilitate documentation typing and synchronization with the diagrams, trees, and other models. 2- Won't any code references in the body of the documentation. 3- Won't any model's modification basing on document's modification. 4- Need a collaborative documentation editing. 5- Need a notification system about all documentation's editing changes. 6- The ModelWriter must be easy to integrate to be a part of the SmartEA project. 7- The ModelWriter must be able to be used on remote. 8- ModelWriter should allow activating/deactivating the synchronization direction "Text -> Model" or "Model -> Text" so SmartEA architects and developers can deactivate the "Text -> model" synchronization option.
Assumptions:	<p>The difficulty in the SmartEA context is not just related to the use of a specific kind of models, but in addition, these models and diagrams are stored remotely on a server, they are subject to authentication rules and can be edited in a collaborative mode manner. Thus, ModelWriter must be able to be stored remotely and must allow to collaboratively editing a document.</p>
Notes and Issues:	<p>Thanks to ModelWriter:</p> <ol style="list-style-type: none"> 1- Provide a new editor to replace the use of Acceleo artifacts, 2- Users can write their own migration document basing on migration models and tools 3- Users can define their own impact analysis document basing on impact models and tools 4- The editor content synchronization must be done automatically 5- The actor will be free to accept (activate/deactivate) synchronization results 6- The actor must be able to activate/deactivate Text --> Model synchronization

2.3 UC-FR3- TITLE (Subject of a Request change)

Decision about this use case is pending – To be decided for the next version of this document.

2.4 UC-FR4- Synchronization of regulation documentation with a design rule repository (OBE0 + Airbus Group + LORIA)

2.4.1 The context and teams constraints

In Airbus context, the purpose of a SIDP document is to describe the installation design principles for a system or for a set of systems in a functional area. The system or the functional area is designated by an identifier called ATA. For example ATA38 refers to Water Waste System while ATA92 refers to Electrical and Optical system.

For each aircraft project, a set of SIDP documents must be produced. Indeed, the aim of a SIDP document is to provide optimal and harmonized installation design principles that comply with the system requirements and take into consideration applicable airworthiness regulations, internal procedures, or weight saving objectives, cost of production and maintenance aspects for the whole aircraft. Depending on these specific constraints that of course may vary depending on the aircraft project (installation, certification, environment, etc.) the SIDP documents can propose specific solutions to fulfil the requirements.

Therefore the teams in charge of authoring these documents (e.g. system installation team) shall actually author and maintain many documents.

In addition, the evolution of applicable regulations may impact some of the rules inside one document or inside a set of documents.

This is why System Installation team started the building of a rule database to be able to manage rules individually, and to speed up and to ease the authoring of SIDP documents.

The total number of rules is not precisely known. According to end-users, there exist around 6000 à 10000 rules within all reference documents related to Electrical Installation Design.

2.4.2 The SIDP documents

The SIPD documents follow a common structure: chapters 1 to 6 introduce the intent, the applicable references, the responsibilities, etc. The chapter 7 is called Design Principle and is the core part describing the installations rules. Chapters 8 to 10 records validation signatures, lists of revisions and appendices. Within chapter 7 - Design Principle, the rules and requirements for system installation are documented using text and images as illustrated below Figure 2-2.

“A space provision will be provided in order to insert a heater strip, if required.

To ensure the fixation of foam **ABS0616**, it is recommended to use self adhesive tape **ABS5662B02** along the longitudinal foam slot.

In the same way, this adhesive tape shall surround the insulation at junction positions as shown in [Figure 7-57](#) below. If insulation parts are twisted or bent on assembly, the ends are required to be cut straight. During assembly, cut the insulation to the correct length as required.”

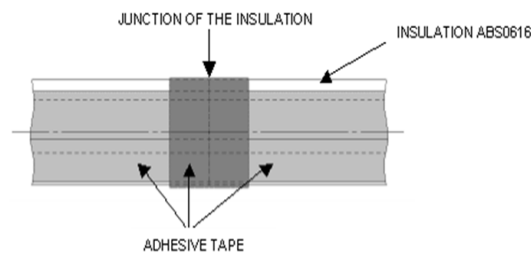


Figure 7-57: Pipe insulation

Figure 2-2 – Example rule

The images are actually today screenshots of models that are otherwise available in numeric and processable format. We call “visual” models the source models for these images; they are mainly tables, diagrams and 2D drawings. Their originating editors may be Excel, UML or SysML or E-R modelers, or 2D-CAD tools.

Thus, one use case for ModelWriter is to allow the synchronized edition of the textual and “visual” model parts of the SIDP documents.

2.4.3 Current SIDP-to-DB scenario

System Installation team started the building of a rule database (SQL) which currently contains a selection of existing installation rules in a semi-structured format. In essence the current SIDP-to-DB scenario manually translates a sentence into an n-ary relation to be stored in the database.

To realize the DB, a first rule data model has been defined as well as an editing environment for these rules. This writing interface is illustrated below Figure 2-3. It shows several frames to enter predefined types of meta-information regarding the rules (Configuration, Context...); and one Rule Content frame to enter the content of a rule in a semi-structured format (corresponding to the data-model):

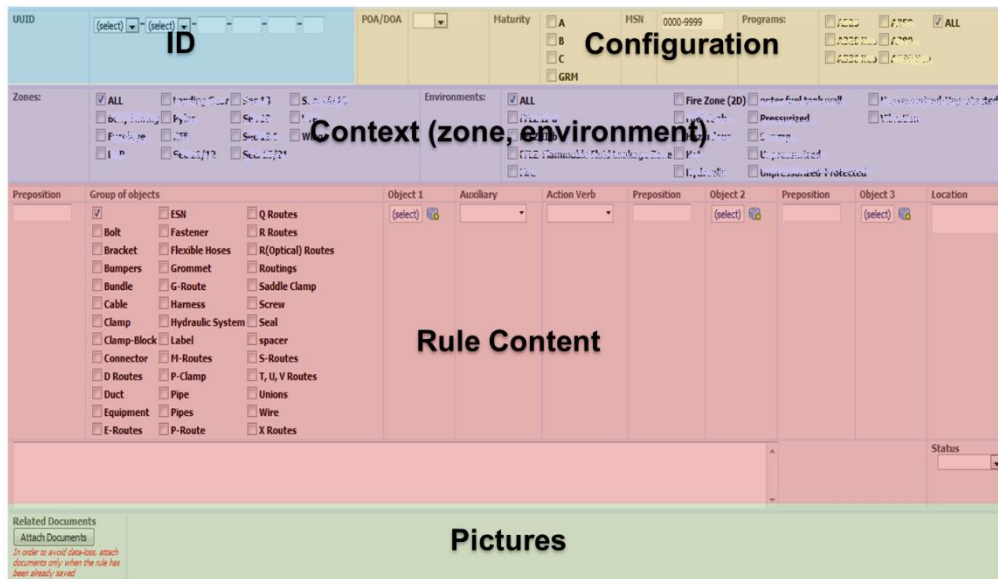


Figure 2-3 – preliminary writing interface for rules

In this semi-structured format:

- some elements of the rule such as for example the “object” are isolated and the allowed vocabulary for it is predefined. For example the Object may be a *bolt*, a *bracket*, etc. and the author shall choose among the available list.
- some writing policy generally applies, for example the use of "shall", "should", "must", "will" and "may" observes guidelines: the word SHALL denotes a mandatory design principle to be applied; while SHOULD denotes a recommendation or advice on implementing a design principle.

Considering this starting situation, other needs were expressed for ModelWriter.

The first need is to help the building of the rule database and ensure its synchronization with the source documents. This requires quite advance semantic analysis of the SIDP content.

- ⇒ Specification for an improved, more relevant, detailed or formal rule data-model or format for the “more formalized rules” can be proposed in the meanwhile
- ⇒ It must be considered that a given rule might be actually documented partially in the text part and partially in a model. For example the text part might identify an equipment (*a given pipe type*) and express some installation condition (*in a wet zone*) while a “visual” model (a table or a 2D drawing) might contain other consequence information (the required distance between the equipment and another object). This is illustrated below Figure 2-5:

Text: Minimum distances between routes shall be in accordance with Table 25
“Visual” model: Table 25

Table 25: Minimum distances between non-

MINIMUM DISTANCES BETWEEN NON-PROTECTED HARNESSES (I)

CATEGORIES	u	x	a	q	z	v	g	s
G, E	§ 2.3.4.2							
X	50	§ 2.3.4.2						
P	50	50	§ 2.3.4.1					
D	100	100	100	§ 2.3.4.1				
M	100	100	100	25	§ 2.3.4.1			
S	150	150	150	75	75	§ 2.3.4.1		
Q	150	150	150	150	150	150	§ 2.3.2	
E	200	200	150	100	100	50	150	

Figure 2-4 – rule content is partially in text and partially in a visual model

⇒ Note: the System installation team objective is that 6 SIDP for A350 shall be put in the DB (over a set of 25 existing documents) – this objective doesn't engage the ModelWriter project of course but we hope that the project will help.

- The second one is to allow the synchronization between the obtained “more formalized rules” contained in the database and the associated “visual” models. This approach may be easier or more efficient than starting from the initial documents. It is expected that the project will consider these alternatives and evaluate possible solutions.

2.4.4 Refinement of the scope of the use case

The use case UC-FR4 will focus on SIDP documents' textual part analysis, and will investigate in how far the manual text-to-DB translation process could be automated. The synchronization mechanism is conceived as a connection between the source texts and the (more) formal representation of the rules extracted from the text.

Yet an alternative course of event is also foreseen. Considering the fact that the current source documents are actually composite ones: they include (today as pictures) snapshots of models of various types (tables, diagrams or 2D drawings) that are in fact edited in numeric format using various modelling tools.

Therefore, we can take advantage of the ModelWriter “basic” annotation capability, to experiment the feasibility to synchronize the models with the text, by identified the model elements “present” in the textual part.

2.4.5 The UC synthesis

Use Case ID:	UC-FR4		
Use Case Name:	Synchronization of regulation documentation with a design rule repository		
Created By:	Airbus Group	Last Updated By:	Airbus Group
Date Created:	30/01/2015	Date Last Updated:	31/01/2015

Primary actor:	System installation author
Secondary actors:	System installation expert
Description:	This use case will explore the use of ModelWriter to translate a document into a repository of design installation rules.

	Legacy documents will have to be parsed, structured and formalized according to a meta-model to be defined. ModelWriter will allow synchronizing the content of the source documents with the formalized rules.
Preconditions/input:	<ol style="list-style-type: none"> 1) A set of SIDP documents is available in a processable format 2) A corpora of existing semi-structured rules is available 3) The modelling editors are connected or can be connected to ModelWriter (connectors to SIRIUS) and the models are represented in ModelWriter (eCore)
Postconditions:	<ol style="list-style-type: none"> 4) The text or elements of a text are converted to a knowledge representation format which can be queried and/or reasoned with (e.g., RDF or OWL) 5) text elements and model elements that match can be automatically synchronized (semantic annotation)
Normal Course of Events:	<ol style="list-style-type: none"> 1. MW analyses the document (text) and build a knowledge representation of it 2. MW manages the synchronization between the document (text) and the formal representation of rules 3. MW allow querying the knowledge representation
Alternative Paths:	<ol style="list-style-type: none"> 1. System installation author edits some rules <ol style="list-style-type: none"> a. in the document (text) b. in a semi-structured way (BD) 2. System installation author edits some models in a modelling editor 3. MW analyses the text and build a knowledge representation of it 4. MW retrieves the models elements that match text elements 4. MW manages the synchronization between text and model and warns in case of changes 5. Visualize synchronization between the models and the existing linked documentation
Exceptions:	
Special Requirements:	6) A specification for an improved and controlled formulation of the design rules in natural language (input documents) may be proposed
Assumptions:	The models are either of type: table, diagram or 2D drawing
Notes and Issues:	

2.5 UC-FR5- Production of a context specific design document (OBEO + Airbus Group + LORIA)

2.5.1 The context and teams constraints

Once authored, the SIDP documents are used by the designers.

SIDP are reference documents for a given aircraft project and their use is mandatory: when developing a new system, designers must find and check all rules and requirements that apply to the various components of that system. The driving need for this use case is to help designers to retrieve rule information when preparing a design task.

Designers spend today a lot of time retrieving all the different information they needed when performing a design task: task to perform (Need), regulations and design principles (Rules) and existing design solutions (Previous design) (Figure 2-3). Because the documentation evolves with time, cross-references across documents and document parts multiply thereby making the checking process very costly and time consuming. Even more, they have difficulties selecting the subset of relevant information applying to a given task.

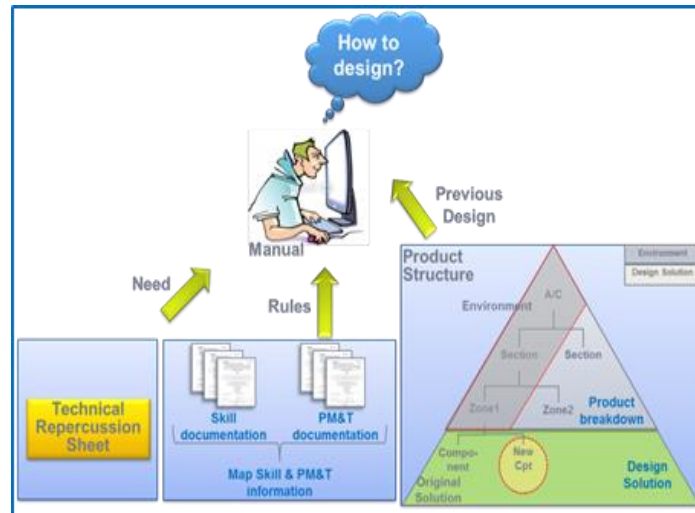


Figure 2-5 – Design context

In that context, a tool that would assist designers and installers retrieving rules relevant to their context would improve productivity and lower design/installation failure rates.

2.5.2 Representing the design task context

In our context, we consider that the elements to describe the design context will have to be converted into search criteria. Possible examples are:

- “Find all rules” related to:
 - A specific route (G, M, X),
 - A specific A/C zone,
 - A specific system (ATA chapter),
 - A type of problem (segregation, electrical protection, minimal distance),
- Find all rules that contains a specific keyword.

The relevant criteria to search and query the rules will have to be established. One approach is to establish them manually. For example, part of them is intended to be captured in the SIPD-to-DB scenario above (§2.4.3) where the rule writing interface foresees the edition of some meta-information about rules, like the characteristics of the zone (spatial area) in which an equipment shall be installed (wet, pressurized, etc..) or some configuration information about the applicable aircraft project.

Therefore, we have identified as an extension for the semantic analysis use case the need to find these criteria that we call “rule conditions” inside the document (text).

2.5.3 Refined scope of the use case

This use case will explore the use of ModelWriter to query both the rules and the visual models together and produce a document (at least a sub-set of rules) that is relevant to a defined design context.

The contextual information includes representations of the targeted product structure zone and impacted components, and a description of the design task. Together with the contextual information model, a design

data repository will be used as source data to produce the derivative document. Meta-models must be defined for representing the contextual information and for generating the output document. Synchronized links between source information and generated document is mandatory.

Use Case ID:	UC-FR5		
Use Case Name:	Production of a context specific design document		
Created By:	Airbus Group	Last Updated By:	Airbus Group
Date Created:	30/10/2015	Date Last Updated:	31/01/2015

Primary actor:	Designer
Secondary actors:	Knowledge manager
Description:	This use case will explore the use of ModelWriter concept to identify the rules “just relevant” to a specified design context. As a result a document, at least a sub-set of rules, must be presented to the end user.
Preconditions:	7) rules elements have been extracted from text and “visual” models parts are converted to a knowledge representation format which can be queried and possibly reasoned with (e.g., RDF or OWL) 8) the contextual information is available as a knowledge representation format which can be queried and/or reasoned with (e.g., RDF or OWL)
Postconditions:	9) composite rules that are relevant wrt a design context are retrieved and presented to the end-user
Normal Course of Events:	10) designer express his installation need (context) 11) designer request the rules that fit his context 12) Rules are retrieved using both text and model representations 13) rules are presented to the designer
Alternative Paths:	
Exceptions:	
Special Requirements:	14) MW should provide a user friendly way to configure the “design need/context” information 15) MW should provide a user friendly mean to manage the knowledge resources that may be needed by its functions (e.g. vocabulary, terminology, ontology)
Assumptions:	
Notes and Issues:	

3 Conclusions & way forward

Based on this document, a set of user requirements has been defined and they have been edited in the project GitHub environment (<https://github.com/ModelWriter/Requirements/>).

These User requirements now need to be reviewed and analysed by ModelWriter technical partners. Following these review and analysis activities:

- the version 1 of User Requirement Document will be delivered (URD) for the complete set of use cases in the project,
- the version 1 of the Software Requirements will be derived, and the first iteration for technical development will be grounded of them.

This Use Case document will be updated in a second iteration.

Annex 1

UC Identifier	D.1.2.2 Public corpora	D.1.2.3 Private corpora
<i>UC-FR1</i>	<p>Sirius source code: http://git.eclipse.org/gitroot/sirius/org.eclipse.sirius.git</p> <p>Sirius documentation: http://git.eclipse.org/gitroot/sirius/org.eclipse.sirius.git</p>	
<i>UC-FR2</i>	<i>Annex 2</i>	
<i>UC-FR4</i>	<i>Sample examples to be constructed</i>	<p><u>1st iteration:</u></p> <ol style="list-style-type: none"> 1) (available) initial corpora <ol style="list-style-type: none"> a. ATA38 SIDP document - SIDP38A001V (pdf format, MSWord format should be made available) b. ATA38 semi-structured rules manually edited in the SQL DB (tabular format) c. Source visual models: Excel tables from SIDP38A001V 2) Current SQL DB rule data-model and associated vocabulary <p><u>2d iteration:</u></p> <ol style="list-style-type: none"> 3) Other type of “visual” models 4) more comprehensive corpora with several SIDP documents

<i>UC-FR5</i>	<i>Sample examples to be constructed</i>	5) ATA38 semi-structured rules (tabular format - excel file) 6) Current SQL DB rule data-model and associated vocabulary 7) System Installation vocabulary (to be built)
---------------	--	--

Annex 2 OBEO-UC-Annex1-SMARTEA

VOY Company

This document describes the **VOY** Company to ensure a better understanding of the company architecture since **VOY** Company is facing a problem at the selling service level. The service is quite unavailable especially during the rush hour. To resolve this problem **VOY** decides to describe some key points of its architecture in this document to help identifying the source of the service unavailability.

In the next sections of the current document, we will describe the company's principles; we will then introduce the existing information system before illustrating the migration plans for the next year.

The company description

VOY Company is a service travel discount company that provides for its customers a list of most unsold trips by other travel agencies. It offers various travel/vacation packages at sale price. The discount of prices may be up to 50% of the original price basically offered by the source travel agencies.

The sale prices of **VOY** offers can be explained by the fact that the departure going and coming dates are fixed and imposed by the company. The customer must be ready to travel in the fifteen days after the reservation.

The existing Information System modeling (AS-IS)

The mission of **VOY** is to enable travelers to seek for and purchase a board array of products from more than 600 airlines, 25 car rental companies, many thousands of lodging properties worldwide, etc.

Therefore, **VOY** is facing the need to provide a very good system quality to book cheapest reservations for multi-modals travels by train, plane, moto, boat and car with an innovative way to reference suppliers. It is going to consolidate the C2C existing systems and allow visitor adding by himself new travel offers.

Today the company achieved a fast hit thanks to its booking abilities. Its booking site experienced a pretty growth period that depends in part on bringing round new customers during the information-unit phase of their purchase cycles.

The company now wants to achieve higher recipes per sale and higher levels of customer retention, while developing new recipes from sponsorships, syndication and third-party retail channels.

To realize these goals, **VOY** teams plan for content-related initiatives in the next fiscal year that would expose customers to more and better-targeted content, both onsite and pushed to customers via a variety of vehicles. At the same time, the company targets to lower costs with its management systems.

The basic System Architecture of **VOY** includes the travel agency's complete customer service process. It contains functions for reservation, invoicing, cashier, account ledgers, accounts receivable, accounting reports, basic customer relationship management, agents, destinations/products, producers, event reports, as well as E-mail and reference payments from external interfaces, and sales statistics, etc.

The travel agency operating process is built into the basic system. All general specialty functions are also included:

- Reservation changes
- Reservation cancellation,
- Tax management,
- Preliminary and final invoices,
- Automatic credit notes,
- Target-specific terms of payment,
- Commissions,
- Entry into accounts based on the first day of travel,
- Reporting to the Consumer Agency, etc.

Expansion modules can be chosen according to needs:

- A tour operator;
- A hotel booking office, etc.

Choose a customer, products sold, and pricing if necessary. Ask **VOY** site to perform invoicing, the Company's site then takes care of the rest: Taxes, commissions, capacity, pricing, and entry of information into accounting.

Sales

As **VOY** targets to get a large market, the **VOY** reputation needs to be very faultless. Some regular surveys will be proposed to customers and we target to reach a satisfying rate of 80% of customers. Here, the central booking office refers to all companies that provide accommodation and related services in the area they are located. Usually, **VOY** has connection with ski resort or other kind of holiday facility. **VOY** can also offer some of their own capacity for sale. An important part of the company's intermediately services is communication with the owners and caretakers of the accommodation, and with providers of other services. Another important duty is the automatic settlement of owner /service provider portions for reservations made. But the central booking office is already using this technics offered by the suitable group of additional services to reach the current satisfaction which is about 60% today.

Consequently we focus on the reservation process, **VOY** targets to get a rate of cancellations of less than 3%. The Commercial Department needs to be proactive to detect all problems in the following services: booking a trip, paying a trip. The department must manage correctly the reservation situations according to the system capacities. They need to constantly supervise capacity and warn of and prevent over-bookings. They also need to calculate utilization rates. In addition, they need to facilitate the automatic accounting process. Information is logged

automatically into the system when reservations are made. The accounting report shows how much is accounted and to whom. Material due for banks are prepared using the accounting process so users don't have to ask for it. All work related to accounting can be done quickly and efficiently with a few keystrokes.

Offers

VOY Company offers travels which correspond to formulas, destinations and hosting services.

A formula is a choice between the following:

- 1- Plane or boat
- 2- Hotel
- 3- Plane or boat + hotel
- 4- Plane or boat + car
- 5- Plane or boat + hotel+ car

A destination is identified by the continent:

- 1- North Africa,
- 2- Africa (except the North),
- 3- Europe,
- 4- Asia,
- 5- America, etc.

The journey is always done in a single country.

The hosting service can be:

- 1- Accommodation only,
- 2- Breakfast
- 3- Half board
- 4- Full board

There is only one provision of accommodation for the whole stay.

The **VOY** innovation this year is focused on their customers: Allow a customer adding new travel offers. Therefore, the subscription of partnership needs to be very easy to use and very reliable. The central booking office can be expanded by taking into account this new function. Accommodation reservations are made on a day-level, and no capacity is reserved for the day of departure. Everyone in the company knows what time check-out must occur on the day of departure, and new customers coming in on the same day to the same room are known. Handling based on times of the day is therefore not completely necessary. Client needs for each product a specific detailed calendar to manage himself his travels, arriving, departures, etc.

With time reservation functions, two factors can be managed:

- Unusual arrival or departure times
- Reservation of additional services on a per hour-basis

If it has been agreed with the customer that instead of the normal check-out time or 12, check-out will take place at 3 PM, then this information will be included with the reservation, and the corresponding calendar is updated consequently so other customers are related to the same information. When making a new reservation for the same room, the system gives a warning about the unusual check-out time, and suitable arrangements can be made with the newly arriving customer.

Additional services such as conferences rooms and sauna times can be reserved on a per-hour basis with the time reservation function. If the conference room is reserved for 91M-3PM, another

reservation can be made for 5PM-8PM. A number of sauna times can be reserved for the same evening, for example 4PM-6PM and 7PM-9PM, and the system warns if overlapping bookings are being made.

Travel Broking

A secondary business is offered by **VOY**: selling trip listing for B2B partners. The IP system will need to provide an SOA access based on Websphere AS 4 to access remotely to several services like ListingOffers.

Here **VOY** employed their travel agencies which act primarily as an intermediary for indirect sales. Sales items include flight and ship tickets, package trips from different tour operators, accommodations, event tickets, etc. In addition to indirect sales, activities also include the agency's own production on a made to order basis. This means that a package is created when an order comes in. An order is often preceded by offer stage, where the agency creates package for the offer. Actual tour operator activities don't exist, which means trips are not available for general sale. The **VOY** System is sufficient for this type of agency, but expansions may be useful. The Fin voice system is often required by customers. Other expansions will make internal activities more efficient. Available expansions to the agency's system are generally intended to make indirect sales more efficient.

Internal cost optimization

To improve the service quality and the TCO, OpenSource software's are preferred as more as possible.

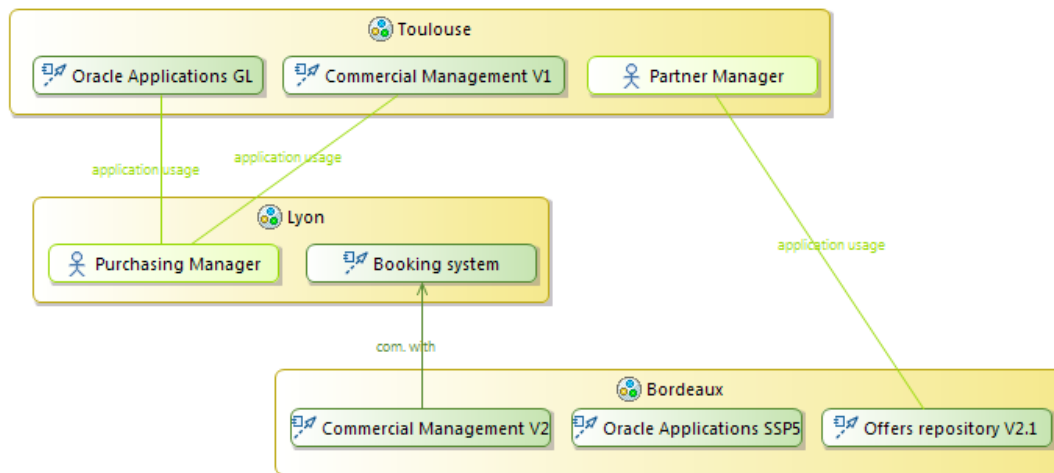
VOY is working on consolidating their booking site. The objective is primarily focusing on decreasing the rate of face to face reservations; as well as the rate of reservations by phone.

Intermediary agent's system can be expanded with the Internet system, through which consumers can reserve flights and hotels by themselves as well as create travel packages (dynamic packages). The site manages connections to external systems used for flights and hotel searches according to customer requests. The site calculates and organizes the results and presents them to customers. The customer is then free to choose a trip, selecting either flight only, accommodation only, or creating a package with both flights and accommodation.

The Company site manages the whole reservation process from the moment a customer first makes contact until the customer pays for the reservation. Once the site has received confirmation of payment, it saves the whole reservation, including payment details into the Company server, accommodation reservations into hotels, Plane companies etc.

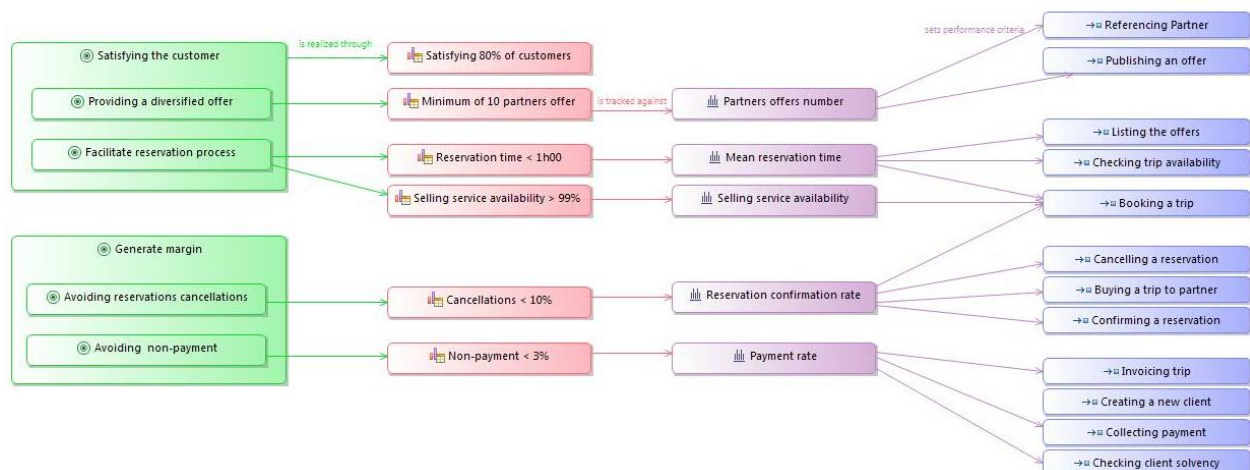
Geographic organization

Today, **VOY** is composed of 3 agencies: Toulouse, Lyon and Bordeaux. The last agency is newly bought; since October 2014. The existing company based in Bordeaux had a duplicated commercial service with the Toulouse agency. As there was a duplication of commercial management business service, **VOY** decided to focus the Toulouse agency service on Partnership management and the Bordeaux agency service on offers management. Then Lyon agency is responsible of the booking system centralization and hosts every purchasing manager's.



The Migration plans for the next year

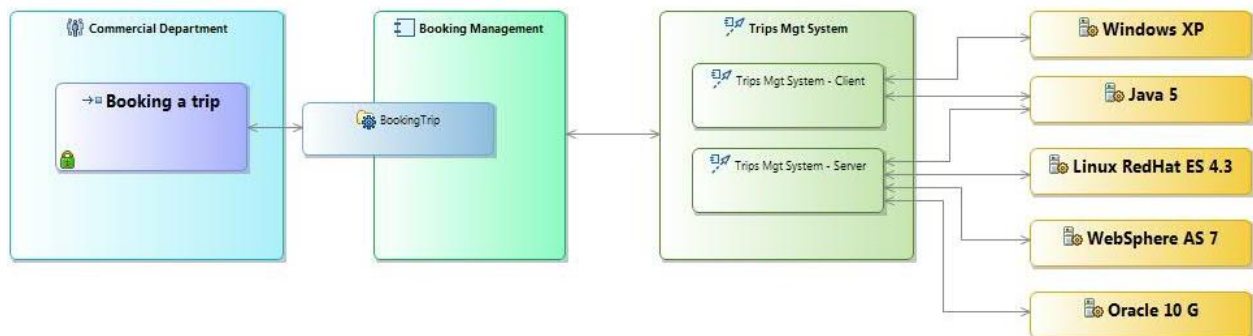
Let's zoom in the general objective of customer's satisfaction, **VOY** need primarily to facilitate the reservation process. The goal of this step is to reach more than 99% of selling service availability. By looking into the Company's architecture, we see that the "Booking a trip" business service is central for the described objective.



This "Booking a trip" service is provided by the booking management application which has identified several failures during the last year.

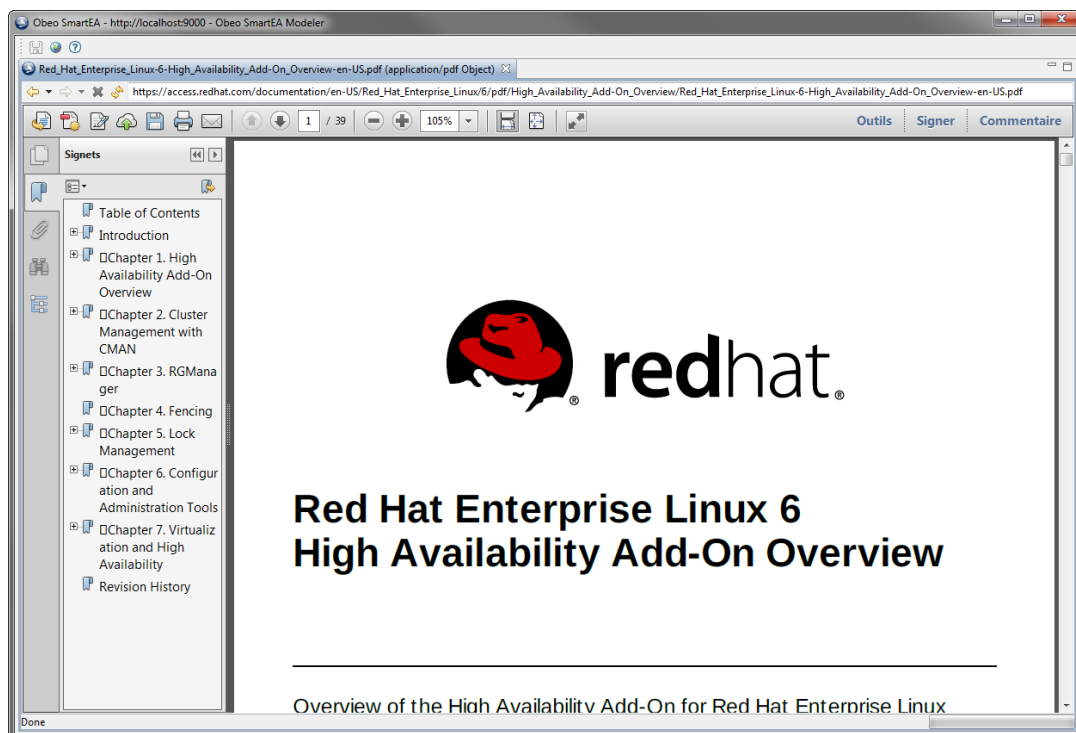
This application relies on several technologies:

- Client :
 - o OS : Windows XP
 - o Language : Java 5
- Server :
 - o OS : Linux Redhat ES 4.3
 - o Language : Java 5
 - o Applications Server: WebShpere AS 7
 - o Database : Oracle 10G



An audit has been realized and proves that the concerned failure is basically related to the technology used to run the whole application. It's the Linux Redhat ES.

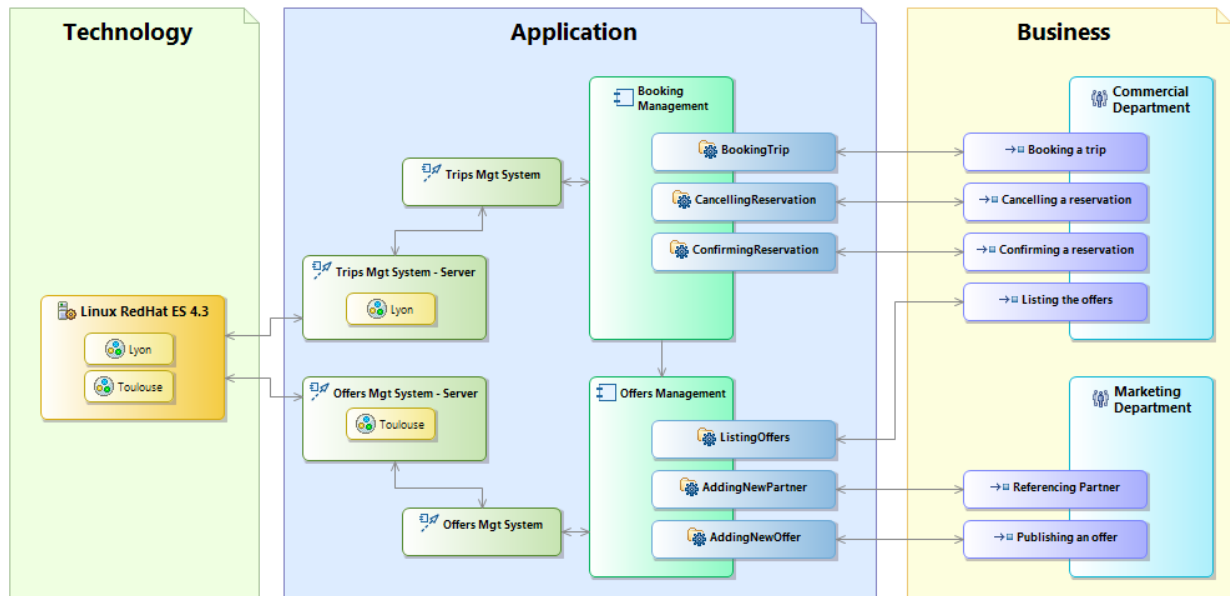
VOY IT Managers recognized that the used server version is quite old (4.3). They also found that the "Linux Redhat AS 6.0"; the new version of the same technology does fix all the detected problems in the earlier one.



That's why **VOY** managers are now looking to replace the buggy technology by the newest one. They need first to justify their choice; then they need to establish a detailed impact analysis of the Linux RedHat ES 4.3 upgrade before producing a migration plan to take action.

They noticed that the use of the existing technology is impacting many levels: the technology, the application and the business levels.

All the related impacts described earlier in the document are illustrated in the following picture:



We look to replace the “**Linux RedHat ES 4.3**” component by the “**Linux RedHat ES 6**” component. The migration plan of improvements between the current architecture and the future one are represented by replacing at the Technology level the buggy server by the new one. This impacts directly the “Trips Mgt System Server” at Lyon agency and the “Offers Mgt System Server” at Toulouse agency.