D1.2.1 deliverable) End paragraph

Start paragraph (Model Writer ) End paragraph

Start paragraph (Text & Model-Synchronized Document Engineering Platform) End paragraph

Start paragraph (••••••••••••••••••••••••••••••••••••••••••••••••••••••••••) End paragraph

Start paragraph (Industrial Use Cases for French Consortium) End paragraph

Start paragraph (Project number: ITEA 2 13028) End paragraph

Start paragraph (Work package: WP1) End paragraph

Start paragraph (Task: 1.2) End paragraph

Start paragraph (Edited by: Anne Monceaux (AIRBUS Group)) End paragraph

Marwa Rostren (Obeo)) End paragraph

Start paragraph (Date: 31/01/2015) End paragraph

Start paragraph (Document version no: 1.0) End paragraph

Disclaimer: *Contractors participating to this report shall incur no liability whatsoever for any damage or loss which may result from the use or exploitation of Information and/or Rights contained in this report.*

*report.*

*the use or exploitation of Information and/or Rights contained in this*

*report.*

Start paragraph (History) End paragraph

|  |  |  |  |
| --- | --- | --- | --- |
| **Document version #** | **Author(s)** | **Date** | **Remarks** |
| **New Cell** | **New Cell** | **New Cell** | **New Cell** |
| Version 1.0 | Anne Monceaux  Marwa Rostren | 31/01/2015 | Initial release |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Start paragraph (Table of contents) End paragraph

1 Introduction 4) End paragraph

Start paragraph (1.1 Role of the deliverable 4) End paragraph

Start paragraph (1.2 Structure of the document 4) End paragraph

Start paragraph (1.3 Terms, abbreviations and definitions 4) End paragraph

Start paragraph (2 Use cases description 5) End paragraph

Start paragraph (2.1 UC-FR1- Synchronization between Models and Documentation (OBEO - Sirius Product) 6) End paragraph

Start paragraph (2.1.1 SIRIUS: The team constraints 7) End paragraph

Start paragraph (2.1.2 SIRIUS: The software and its source code 7) End paragraph

Start paragraph (2.1.3 SIRIUS: The used models and the documents 8) End paragraph

Start paragraph (2.1.4 SIRIUS: The documentation synchronization examples 8) End paragraph

Start paragraph (1. Add a New Feature 8) End paragraph

Start paragraph (2. Improve an Existing Feature 9) End paragraph

Start paragraph (2.1.5 SIRIUS: The UC synthesis 10) End paragraph

Start paragraph (2.2 UC-FR2- Enterprise Architecture (OBEO - SmartEA Product) 12) End paragraph

Start paragraph (1- SMARTEA: The team constraints 13) End paragraph

Start paragraph (2- SMARTEA: The software and its source code 13) End paragraph

Start paragraph (3- SMARTEA: The used models and the documents 13) End paragraph

Start paragraph (4- SMARTEA: The use case Examples 15) End paragraph

Start paragraph (5- SMARTEA: The UC synthesis 17) End paragraph

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

Start paragraph (2.4 UC-FR4- Synchronization of regulation documentation with a design rule repository (Airbus Group) 19) End paragraph

Start paragraph (2.4.1 The context and teams constraints 19) End paragraph

Start paragraph (2.4.2 The SIDP documents 20) End paragraph

Start paragraph (2.4.3 Current SIDP-to-DB scenario 21) End paragraph

Start paragraph (2.4.4 Refinement of the scope of the use case 22) End paragraph

Start paragraph (2.4.5 The UC synthesis 23) End paragraph

Start paragraph (2.5 UC-FR5- Production of a context specific design document (Airbus Group) 24) End paragraph

Start paragraph (2.5.1 The context and teams constraints 24) End paragraph

Start paragraph (2.5.2 Representing the design task context 25) End paragraph

Start paragraph (2.5.3 Refined scope of the use case 25) End paragraph

Start paragraph (3 Conclusions & way forward 27) End paragraph

Start paragraph (4 Annex 1 27) End paragraph

# Start paragraph (Introduction) End paragraph

## Start paragraph (Role of the deliverable) End paragraph

Start paragraph (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. ) End paragraph

## Start paragraph (Structure of the document) End paragraph

Start paragraph (This document is organized as follows:) End paragraph

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

## Start paragraph (Terms, abbreviations and definitions) End paragraph

Start paragraph (Please add additional terms, abbreviations and definitions for your deliverable.) End paragraph

|  |  |
| --- | --- |
| A/C | Aircraft |
| New Cell | New Cell |
| ATA | Air Transport Association |
| DB | Data base |
| DP | Design Principles |
| RDF | Resource Description Framework |
| RDFS | Resource Description Framework Schema |
| SIDP | System Installation Design Principle |
| UC | Use case |
| WP | Work Package |
|  |  |
|  |  |
|  |  |
|  |  |

Start paragraph (Table 1‑1: Terms, Abbreviations and Definitions) End paragraph

# Start paragraph (Use cases description) End paragraph

Start paragraph (The use cases are provided by Obeo and Airbus Group companies.) End paragraph

Start paragraph (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. ) End paragraph

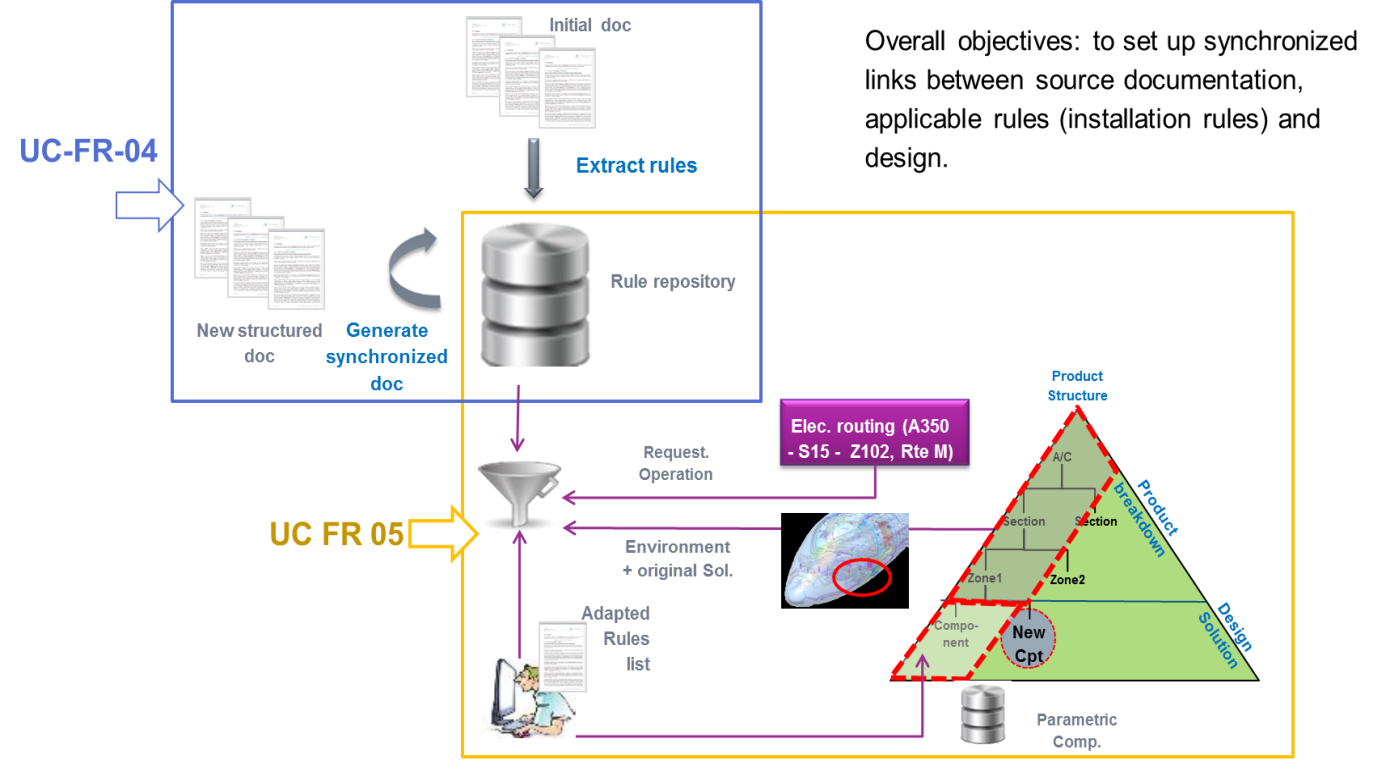
Start paragraph (In the first and second sections, OBEO describes and defines Sirius (UC-FR1) and SMARTEA (UC-FR2) products, its existing documentation life cycles and its requirements for each.) End paragraph

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

Start paragraph (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. ) End paragraph

Start paragraph (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.) End paragraph

Start paragraph (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.) End paragraph



Start paragraph (Figure 2‑1 UC-FR-04 + UC-FR-05 common context and interface) End paragraph

## Start paragraph (UC-FR1- Synchronization between Models and Documentation (OBEO - Sirius Product)) End paragraph

Start paragraph (OBEO’s activity is based on its own software’s like Sirius (http://eclipse.org/sirius/). Sirius documentation is the first important Use Case for OBEO to prove the validity of the “Text & Model – Synchronized Document Engineering” approach of ModelWriter.) End paragraph

Start paragraph (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:) End paragraph

* Start paragraph (The specification,) End paragraph
* Start paragraph (The Web page documentation content,) End paragraph
* Start paragraph (The Developer Manual) End paragraph
* Start paragraph (The Specifier Manual) End paragraph
* Start paragraph (The User Manual, tutorials and Cheat-sheets) End paragraph
* Start paragraph (The release Notes) End paragraph
* Start paragraph (The Javadoc) End paragraph

Start paragraph (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.) End paragraph

Start paragraph (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. ) End paragraph

Start paragraph (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. ) End paragraph

Start paragraph (Consequently, the current documentation is:) End paragraph

* Start paragraph (Too long to be considered as start-up documentation for users.) End paragraph
* Start paragraph (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.) End paragraph

Start paragraph (In addition, today the existing Javadoc is not trusty; it was not updated since a while.) End paragraph

Start paragraph (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. ) End paragraph

### Start paragraph (SIRIUS: The team constraints) End paragraph

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

### Start paragraph (SIRIUS: The software and its source code) End paragraph

Start paragraph (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/) End paragraph](http://www.obeodesigner.com/)

Start paragraph (The source code is available by cloning the repository:) End paragraph

Start paragraph ( [http://git.eclipse.org/gitroot/sirius/org.eclipse.sirius.git) End paragraph](http://git.eclipse.org/gitroot/sirius/org.eclipse.sirius.git)

### Start paragraph (SIRIUS: The used models and the documents) End paragraph

Start paragraph (The Sirius models are dispatched in several \*.ecore models listed below:) End paragraph

* Start paragraph (diagram.ecore – used to manage diagrams representations concepts. This model can be found in the “model” folder in the “org.eclipse.sirius.diagram” plugin.) End paragraph
* Start paragraph (layoutdata.ecore – used to manage layoutdata concepts. This model can be found in the “model” folder in the “org.eclipse.sirius.diagram.layoutdata” plugin.) End paragraph
* Start paragraph (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.) End paragraph
* Start paragraph (table.ecore – used to manage tables representations concepts. This model can be found in the “model” folder in the “org.eclipse.sirius.table” plugin.) End paragraph
* Start paragraph (tree.ecore – used to manage trees representations concepts. This model can be found in the “model” folder in the “org.eclipse.sirius.tree” plugin.) End paragraph
* Start paragraph (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.) End paragraph

Start paragraph (Any change in one or more of these models provides asynchronous parts in the documentation. ) End paragraph

**Start paragraph (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.) End paragraph

Start paragraph (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: ) End paragraph

[Start paragraph (http://git.eclipse.org/gitroot/sirius/org.eclipse.sirius.git](http://git.eclipse.org/gitroot/sirius/org.eclipse.sirius.git). ) End paragraph

Start paragraph (This documentation does not cover all the models artefacts but it covers the main concepts, scenarios and features.) End paragraph

Start paragraph (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.) End paragraph

### Start paragraph (SIRIUS: The documentation synchronization examples) End paragraph

#### Start paragraph (Add a New Feature) End paragraph

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

|  |
| --- |
| 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>* |
| New Cell |

Start paragraph (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: ) End paragraph

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

#### Start paragraph ( Improve an Existing Feature) End paragraph

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

|  |
| --- |
| 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>* |
| New Cell |

Start paragraph (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.) End paragraph

### Start paragraph (SIRIUS: The UC synthesis) End paragraph

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-FR1 | | |
| **New Cell** | New Cell | | |
| **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 |
| **New Cell** | New Cell |
| **Secondary actors:** | Sirius Contributors |
| **Description:** | 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.  Limitations:  - 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.  - 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:** | 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:** | 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:** | 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:** | Sirius Team Requirements:  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:** | Thanks to ModelWriter:  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 |

## Start paragraph (UC-FR2- Enterprise Architecture (OBEO - SmartEA Product)) End paragraph

Start paragraph (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.) End paragraph

Start paragraph (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.) End paragraph

Start paragraph (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.) End paragraph

Start paragraph (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.) End paragraph

### Start paragraph (SMARTEA: The team constraints) End paragraph

* Start paragraph (We need a new editor to facilitate documentation typing and synchronization with the diagrams, trees, and other models.) End paragraph
* Start paragraph (We won’t any code references in the body of the documentation.) End paragraph
* Start paragraph (We need a collaborative documentation editing.) End paragraph
* Start paragraph (We need a notification system about all documentations changes after editing.) End paragraph
* Start paragraph (The ModelWriter must be easy to integrate to be a part of the SmartEA project.) End paragraph
* Start paragraph (The ModelWriter must be able to be used on remote.) End paragraph
* Start paragraph (We won’t any models modification basing on documents modification. ) End paragraph
* Start paragraph (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.) End paragraph

### Start paragraph (SMARTEA: The software and its source code ) End paragraph

Start paragraph (The SmartEA software is not an open source project. Thus the source code will not be shared with the ModelWriter project partners.) End paragraph

Start paragraph (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. ) End paragraph

Start paragraph (To learn how to use the SmartEA product you can refer to the following link:) End paragraph

[Start paragraph (http://www.obeosmartea.com/product/online-demo](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).) End paragraph

### Start paragraph (SMARTEA: The used models and the documents) End paragraph

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

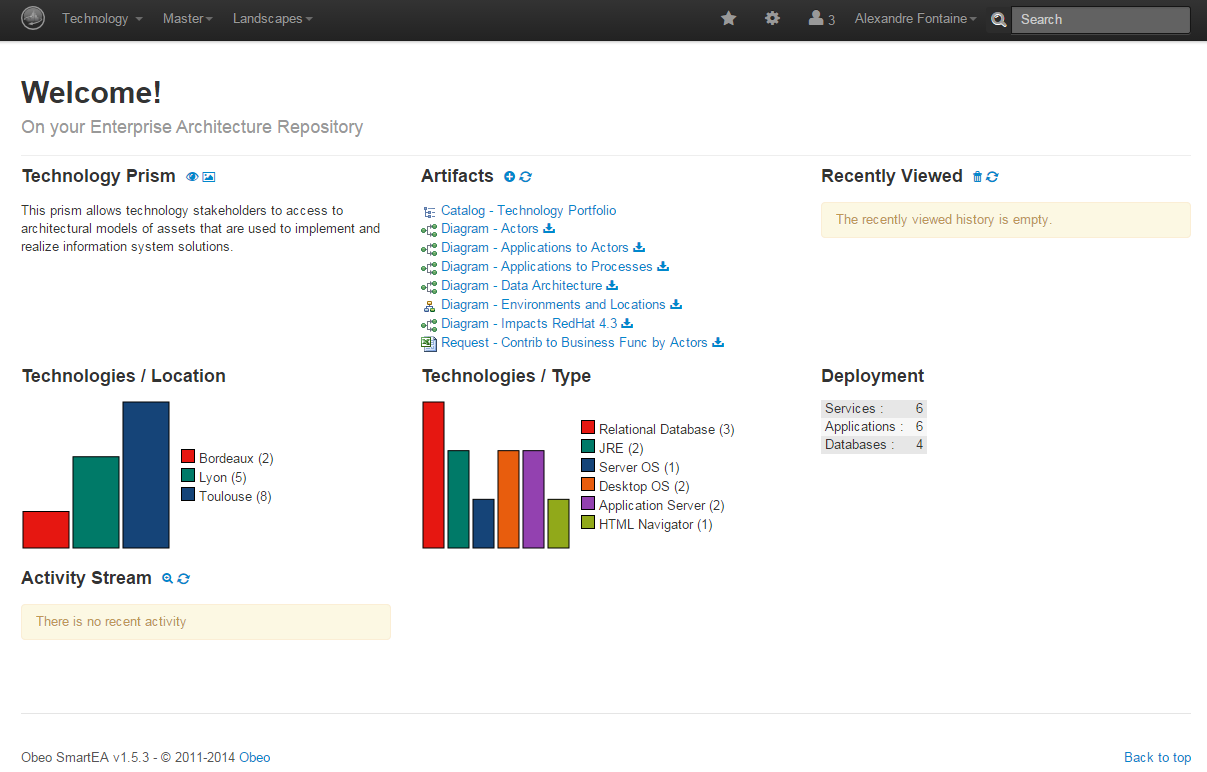
Start paragraph (Today, the migration plans are not documented but models are available to indicate all modifications between source and target architectures to deduce the migration plans. The impact analyzes are not documented neither but related artifacts, representations and references can be deduced automatically basing on given models.) End paragraph

Start paragraph (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/> .) End paragraph

Start paragraph (We assume that:) End paragraph

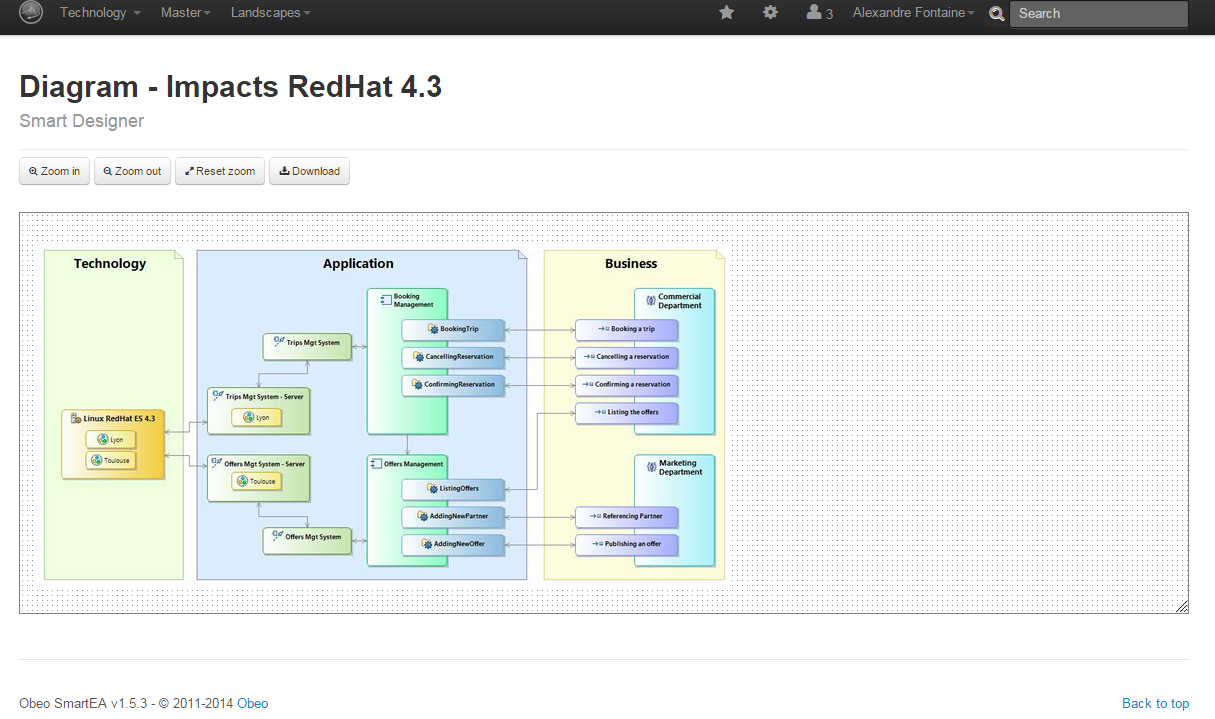
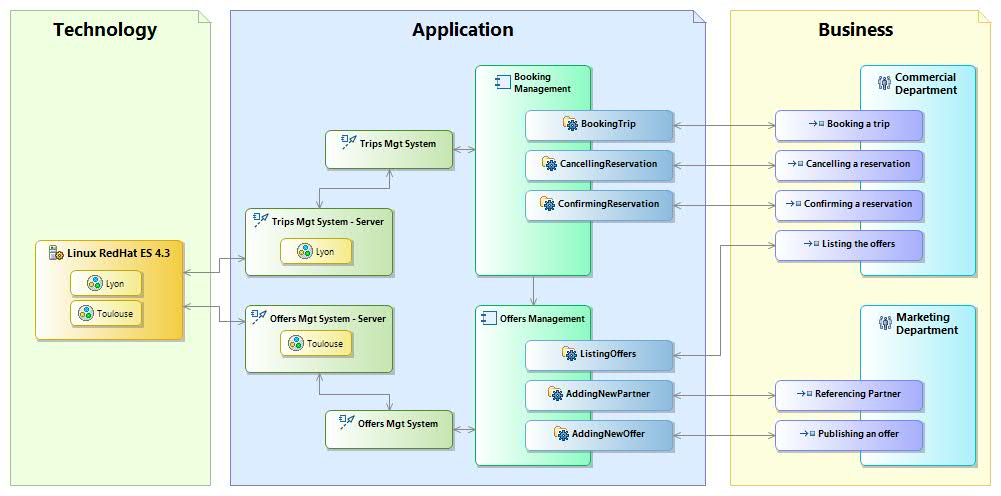
* Start paragraph (you have already installed the trial version of SmartEA product, ) End paragraph
* Start paragraph (you have started the server,) End paragraph
* Start paragraph (you are connected to: [**http://localhost:9000/**](http://localhost:9000/)as “**afontaine**” user, using the password “**123**”) End paragraph

Start paragraph (The welcome page will be displayed as follows:) End paragraph



Start paragraph (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.) End paragraph

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



### Start paragraph (SMARTEA: The use case Examples) End paragraph

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

Start paragraph (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. ) End paragraph

Start paragraph (We need to introduce a useful editor to facilitate the documentation typing and the documentation synchronization with the project models and diagrams. ) End paragraph

Start paragraph (The editor must also be able to manage collaborative features and must ensure users notification.) End paragraph

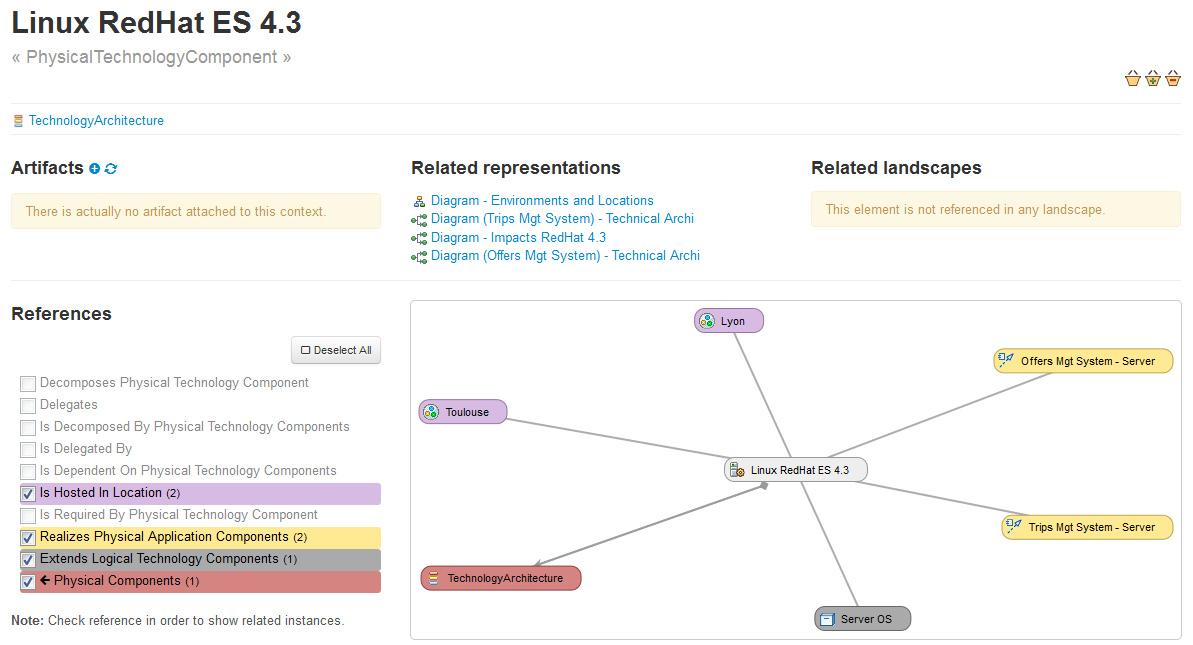
Start paragraph (The **Annex 1 (D1.2.1-OBEO-UC-Annex1-SMARTEA.doc)** presents a document example related to the **VOY** Company example. The document contains the general architecture of the Company and a section focusing on the used technology replacement to establish the impact analysis and migration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

Start paragraph (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.) End paragraph

Start paragraph (Deleting the “**Linux RedHat ES 4.3**” component impacts several representations:) End paragraph

* Start paragraph (The Environments and Locations diagram) End paragraph
* Start paragraph (The Trips Mgt System – Technical Architecture diagram) End paragraph
* Start paragraph (The Offers Mgt System – Technical Architecture diagram) End paragraph
* Start paragraph (The Impacts RedHat 4.3 diagram) End paragraph

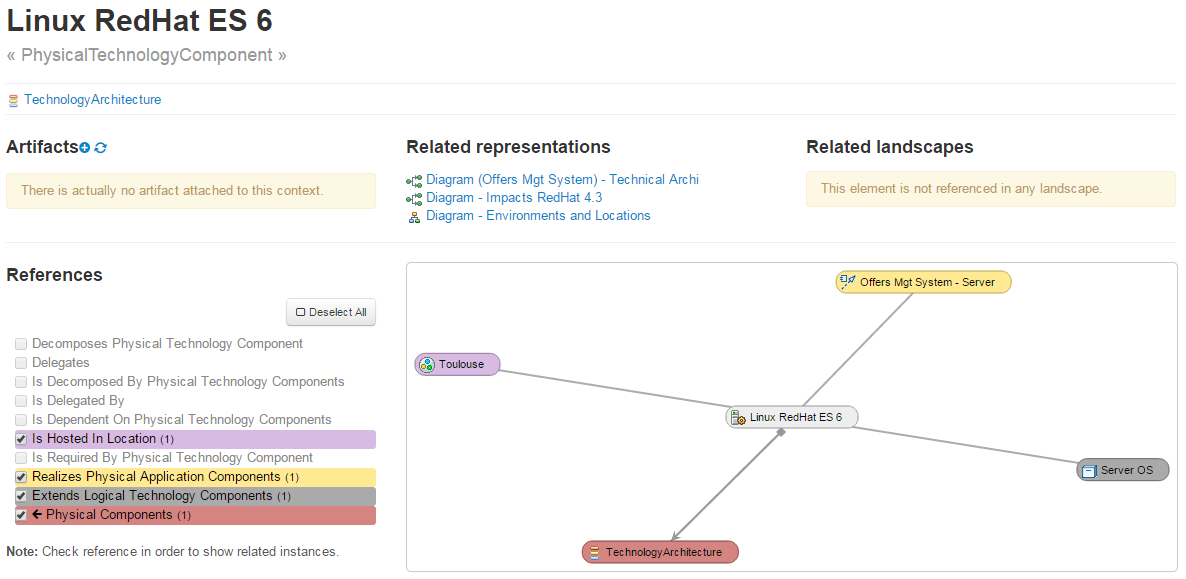
Start paragraph (This information can be reached by displaying the related representations by double clicking on the “**Linux RedHat ES 4.3**” component of the tree above.) End paragraph



Start paragraph (Adding the “**Linux RedHat ES 6**” component impacts several representations:) End paragraph

* Start paragraph (The Environments and Locations diagram) End paragraph
* Start paragraph (The Offers Mgt System – Technical Architecture diagram) End paragraph
* Start paragraph (The Impacts RedHat 4.3 diagram) End paragraph

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



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

Start paragraph (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.) End paragraph

### Start paragraph (SMARTEA: The UC synthesis) End paragraph

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-FR2 | | |
| **New Cell** | New Cell | | |
| **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 |
| **New Cell** | New Cell |
| **Secondary actors:** | SmartEA Team |
| **Description:** | 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 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.  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.  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. |
| **Preconditions:** | 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:** | As for user documentation:  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.  As for migration plans:  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:** | 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:** | SmartEA Team Requirements  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:** | 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. |
| **Notes and Issues:** | Thanks to ModelWriter:  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 |

## Start paragraph (UC-FR3- TITLE (Subject of a Request change)) End paragraph

Start paragraph (Decision about this use case is pending – To be decided for the next version of this document. ) End paragraph

## Start paragraph (UC-FR4- Synchronization of regulation documentation with a design rule repository (OBEO + Airbus Group + LORIA)) End paragraph

### Start paragraph (The context and teams constraints) End paragraph

Start paragraph (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.) End paragraph

Start paragraph (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. ) End paragraph

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

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

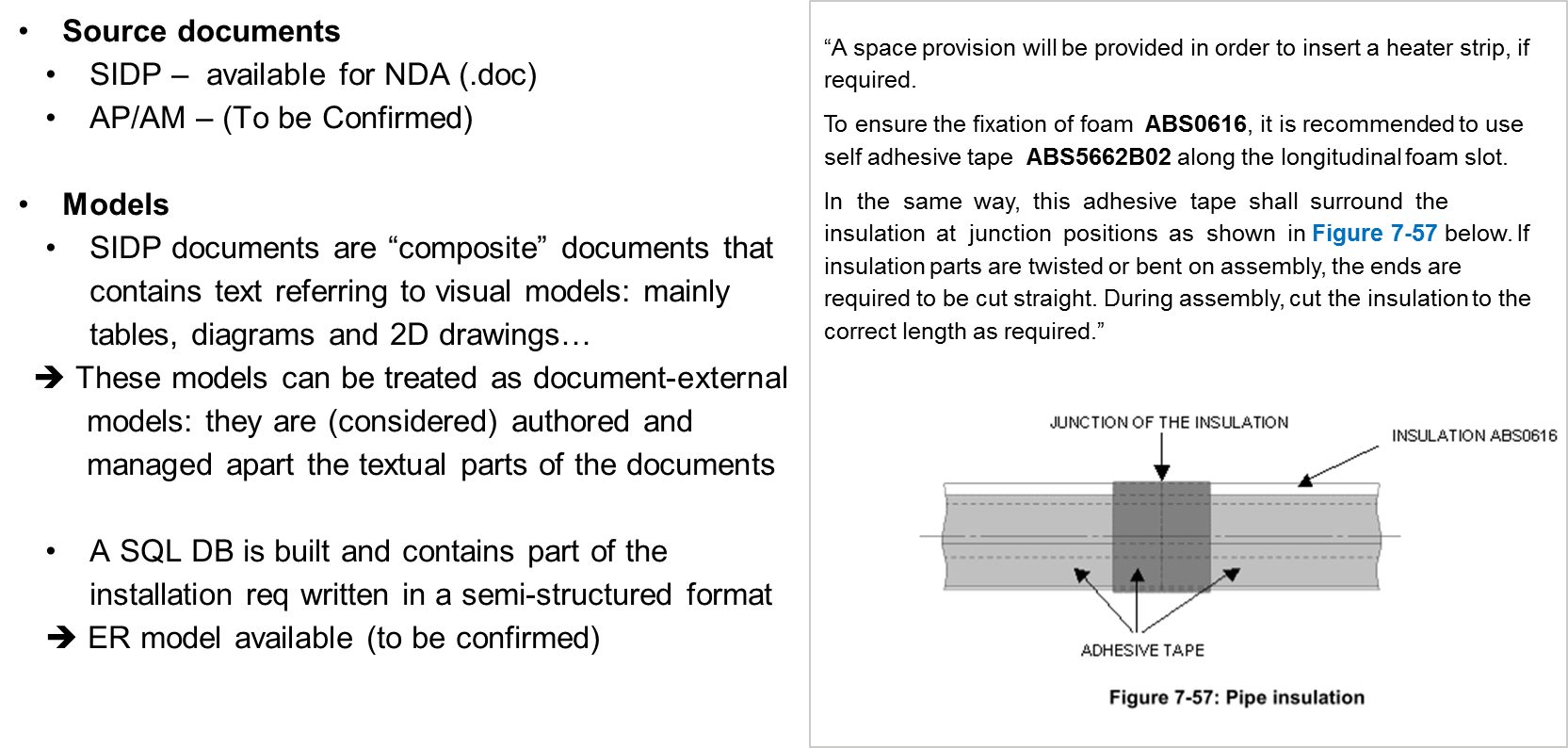
Start paragraph (This is why System Installation team started the building of a rule database to be able to manage rules individually, and to speed up and to ease the authoring of SIDP documents.) End paragraph

Start paragraph (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.) End paragraph

### Start paragraph (The SIDP documents) End paragraph

Start paragraph (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.) End paragraph

Start paragraph (Within chapter 7 - Design Principle, the rules and requirements for system installation are documented using text and images as illustrated below Figure 2‑2. ) End paragraph



Start paragraph (Figure 2‑2 – Example rule) End paragraph

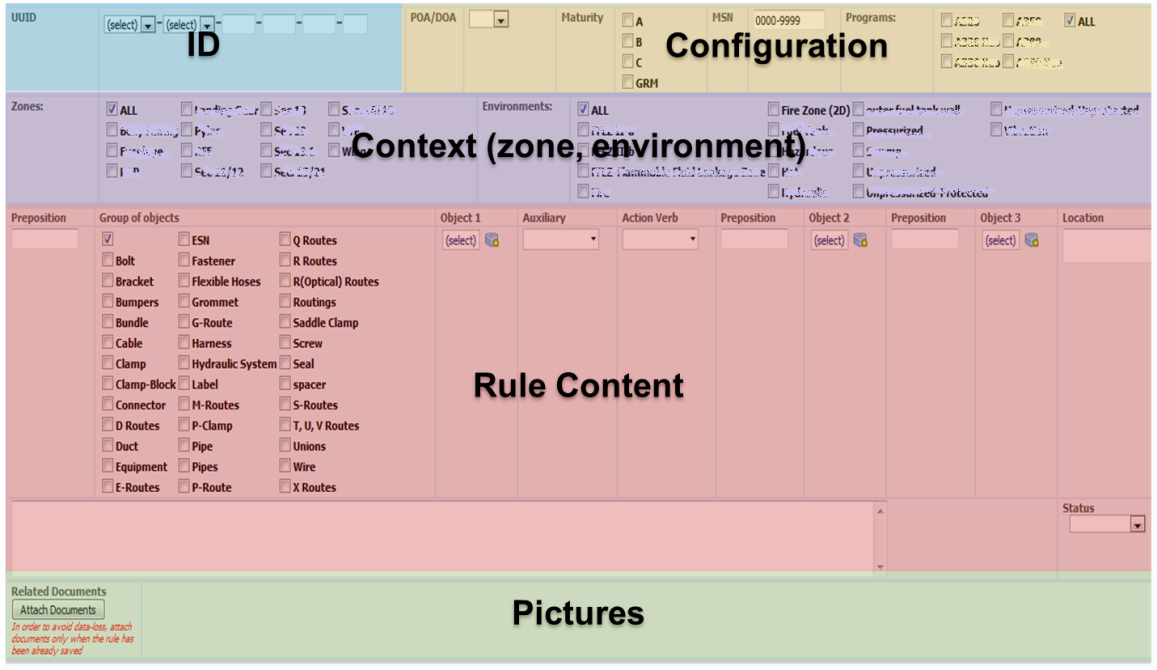
Start paragraph (The images are actually today screenshots of models that are otherwise available in numeric and processable format. We call “visual” models the source models for these images; they are mainly tables, diagrams and 2D drawings. Their originating editors may be Excel, UML or SysML or E-R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

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

### Start paragraph (Current SIDP-to-DB scenario) End paragraph

Start paragraph (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.) End paragraph

Start paragraph (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): ) End paragraph



Start paragraph (Figure 2‑3 – preliminary writing interface for rules) End paragraph

Start paragraph (In this semi-structured format: ) End paragraph

* Start paragraph (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.) End paragraph
* Start paragraph (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.) End paragraph

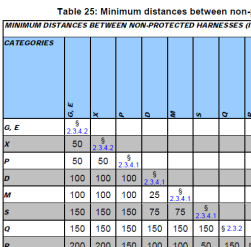
Start paragraph (Considering this starting situation, other needs were expressed for ModelWriter. ) End paragraph

Start paragraph (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. ) End paragraph

* + Start paragraph (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) End paragraph
  + Start paragraph (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:) End paragraph

Start paragraph (Text*: Minimum distances between routes shall be in accordance with Table 25) End paragraph*

Start paragraph (“Visual” model: Table 25) End paragraph



Start paragraph (Figure 2‑4 – rule content is partially in text and partially in a visual model) End paragraph

* + *Start paragraph (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.) End paragraph*
* Start paragraph (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.) End paragraph

### Start paragraph (Refinement of the scope of the use case) End paragraph

Start paragraph (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.) End paragraph

Start paragraph (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.) End paragraph

### Start paragraph (The UC synthesis) End paragraph

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-FR4 | | |
| **New Cell** | New Cell | | |
| **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:** | 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:** | 1. 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) 2. text elements and model elements that match can be automatically synchronized (semantic annotation) | | |
| **Normal Course of Events:** | 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:** | 1. System installation author edits some rules    1. in the document (text)    2. 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 5. MW manages the synchronization between text and model and warns in case of changes 6. Visualize synchronization between the models and the existing linked documentation | | |
| **Exceptions:** |  | | |
| **Special Requirements:** | 1. 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:** |  | | |

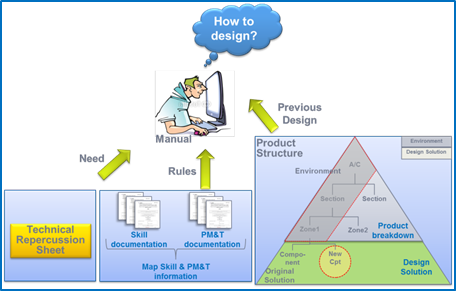
## Start paragraph (UC-FR5- Production of a context specific design document (OBEO + Airbus Group + LORIA)) End paragraph

### Start paragraph (The context and teams constraints) End paragraph

Start paragraph (Once authored, the SIDP documents are used by the designers. ) End paragraph

Start paragraph (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.) End paragraph

Start paragraph (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 2‑4). 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.) End paragraph



Start paragraph (Figure 2‑5 – Design context) End paragraph

Start paragraph (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.) End paragraph

### Start paragraph (Representing the design task context) End paragraph

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

Start paragraph (– “Find all rules” related to: ) End paragraph

* Start paragraph (A specific route (G, M, X), ) End paragraph
* Start paragraph (A specific A/C zone,) End paragraph
* Start paragraph (A specific system (ATA chapter), ) End paragraph
* Start paragraph (A type of problem (segregation, electrical protection, minimal distance), ) End paragraph

Start paragraph (– Find all rules that contains a specific keyword.) End paragraph

Start paragraph (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.) End paragraph

Start paragraph (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). ) End paragraph

### Start paragraph (Refined scope of the use case) End paragraph

Start paragraph (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.) End paragraph

Start paragraph (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.) End paragraph

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-FR5 | | |
| **New Cell** | New Cell | | |
| **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:** | 1. 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) 2. the contextual information is available as a knowledge representation format which can be queried and/or reasoned with (e.g., RDF or OWL) | | |
| **Postconditions:** | 1. composite rules that are relevant wrt a design context are retrieved and presented to the end-user | | |
| **Normal Course of Events:** | 1. designer express his installation need (context) 2. designer request the rules that fit his context 3. Rules are retrieved using both text and model representations 4. rules are presented to the designer | | |
| **Alternative Paths:** |  | | |
| **Exceptions:** |  | | |
| **Special Requirements:** | 1. MW should provide a user friendly way to configure the “design need/context” information 2. 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:** |  | | |

# Start paragraph (Conclusions & way forward) End paragraph

Start paragraph (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/> ). ) End paragraph

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

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

Start paragraph (This Use Case document will be updated in a second iteration. ) End paragraph

Start paragraph (Annex 1 ) End paragraph

|  |  |  |
| --- | --- | --- |
| **UC Identifier** | **D.1.2.2 Public corpora** | **D.1.2.3 Private corpora** |
| **New Cell** | **New Cell** | **New Cell** |
| *UC-FR1* | Sirius source code:  <http://git.eclipse.org/gitroot/sirius/org.eclipse.sirius.git>  Sirius documentation:  <http://git.eclipse.org/gitroot/sirius/org.eclipse.sirius.git> |  |
| *UC-FR2* | *D1.2.2-OBEO-UC-Annex1-SMARTEA.doc* |  |
| *UC-FR4* | *Sample examples to be constructed* | 1st iteration:   1. (available) initial corpora    1. ATA38 SIDP document - SIDP38A001V (pdf format, MSWord format should be made available)    2. ATA38 semi-structured rules manually edited in the SQL DB (tabular format)    3. Source visual models: Excel tables from SIDP38A001V 2. Current SQL DB rule data-model and associated vocabulary   2d iteration:   1. Other type of “visual” models 2. more comprehensive corpora with several SIDP documents |
| *UC-FR5* | *Sample examples to be constructed* | 1. ATA38 semi-structured rules (tabular format - excel file) 2. Current SQL DB rule data-model and associated vocabulary 3. System Installation vocabulary (to be built) |