D4.1.b - Process description (first version)

WP4.1 - Task Identification and Process Proposal

Abstract: This report is a deliverable prepared by Insitut-Mines Telecom Atlantique for the project FORMOSE funded by ANR France. It explains the initial proposed process of the development of formal requirement models for critical and complex systems. This process takes into account the proposed methods and tools developed under the banner of this research project. This deliverable caters for the initial process and the next deliverable (D4.1.c) shall update the process with working tools and a complete documentation.

Status: Public / ~~Confidential~~
Version: ~~Draft~~ / Review / ~~Final~~

Version history

Version	Date	Contributors	Contribution
1.0	15-05-2017	Fahad R. Golra	Drafting the initial process document.
1.1	22-05-2017	Fabien Dagnat	Corrections and suggestions.
1.2	09-07-2018	Fahad R. Golra	Process document updates.
1.3	04-10-2018	Antoine Beugnard	Internal review.
1.4	28-11-2018	Sylvain Guérin	Update after new tool delivery.
1.4	29-11-2018	Fabien Dagnat	Internal review.

Contents

List of Figures

1 Scope

1.1 Identification

Document Name:	Process description (first version)
Deliverable Identification Number:	D4.1.b
Project Name:	ANR Formose
File Name:	Formose-D4.1.b

1.2 System overview

Formose is an industrial research project aiming to provide a formally-grounded, model-based requirements engineering method for critical complex systems, supported by an open-source environment. A process for requirements engineering resides at the heart of the proposed method and tools. This process is the driving force for the FORMOD tool, developed in this research project. Further details on the FORMOSE project and the FORMOD tool are available on the Wiki of the project, accessible through the following URL:

https://formose-pass.enstb.org/

1.3 Document overview

This document describes the first version of the process description for the methodology and tools proposed by the FORMOSE project. The next deliverable (D4.1.c) will update this document to present the updated process with working tools and a complete documentation.

2 FORMOSE Architecture

The tool developed under the FORMOSE project for the modeling of formal requirements is named as *FORMOD - Formose Requirements Modeler*. This tool is developed around the proposed methodology for modeling the requirements for critical and complex systems, where formal methods can be used for requirements verification and validation.

2.1 Process-centric architecture

The architecture of the proposed system is developed in a process-centric way, where methodologies are the key concepts for the usage of multiple views. The complete architecture is developed around five main components *i.e.* FORMOSE Core (containing Formose VM and Methodology VM), Document Library VM, Domain modeling VM, SysML-KAOS VM and B-method VM. Figure 1 presents the high level architecture of the proposed system with four (4) methodology specialisations: SysML KAOS Methodology, Document Annotation Methodology, Domain Model Methodology and SysML-KAOS-B-Ontology Methodology.

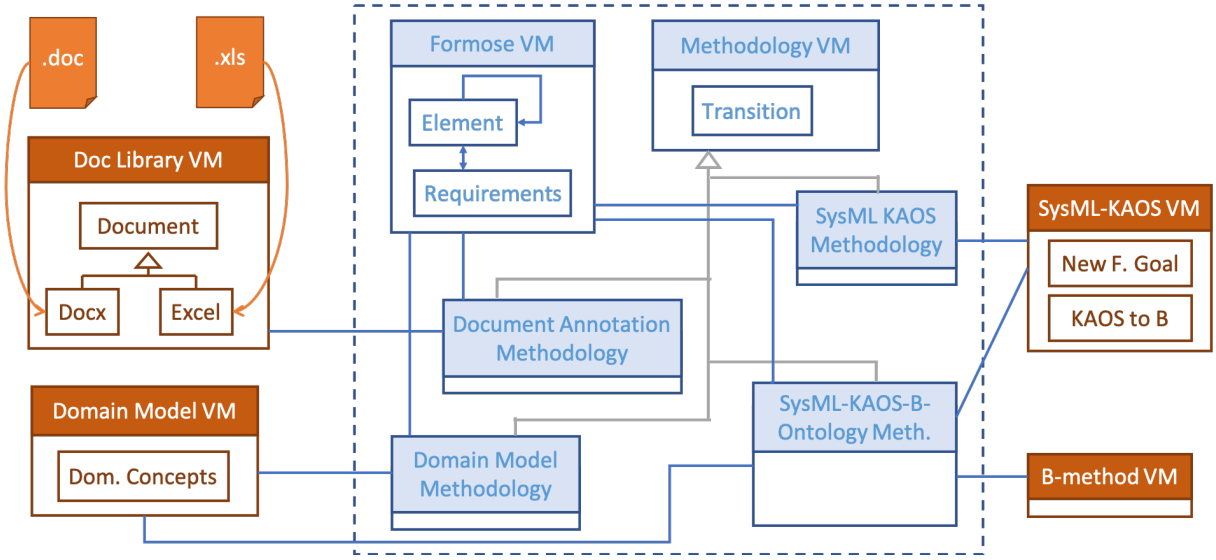

Figure 1: FORMOD Architecture

Each of these five components used for the development of the proposed system are explained in the following. Brown components in Figure 1 are domain specific, while blue ones are dedicated to the federation of these domains.

FORMOSE Core

The proposed system is developed as a process-centric architecture, where FORMOSE Core serves as the centralized model that manages the rest of the virtual models (VM) using specific methodology models. Formose Core is developed using two virtual models *i.e.* Formose VM and the Methodology VM. Formose VM serves as the main abstract requirements metamodel, where each **requirements**[1] are associated to other **elements** of the system. The Methodology

[1] this format denotes concepts of virtual models. See Fig 1.

virtual model presents the concept of transitions that serve as a building block for the overall process. A methodology for each of the other components is specialized from this Methodology virtual model. Each specific methodology for other virtual models, allows for linking the Formose virtual model to the specific virtual model.

Document Library Model

The document library model provides a means for the FORMOD tool to access, read and write to the different document files. In the initial prototype of the tool, the docx documents are implemented through the docx connector of Openflexo. This document files are used to import and export requirements documents / domains documents to the tool.

Domain Model

This VM allows to model the concepts of the domain model. The concepts of the domain are modeled using ontologies. owl connector of Openflexo is used for the implementation of this model. It allows linking the concepts used in the requirements to the ones described in the domain models.

SysML-KAOS Model

SysML-KAOS is a goal modeling language that allows the decomposition of the system through SysML while using KAOS goals as the motivation for each system decomposition. This virtual model allows linking the requirements to the SysML-KAOS modeling editor. Overall, this can be used to develop goal models out of the existing requirements and to elicit requirements through the refinement of goals.

B Model

This virtual model serves for connecting the FORMOD tool with the B method editor. The B method editor was not implemented in the first version of the tool. However, the final version of the tool shall offer a mechanism for the formal specification of requirements, using this virtual model.

2.2 Process-centric Instantiation

An instance of the FORMOSE architecture instantiates a tree of elements, where multiple requirements are linked to each of these elements, as shown in Figure 2. An element in itself is a very abstract level concept that does not describe which actual entity would be linked to these requirements. The use of multiple methodologies in the FORMOD architecture link these elements of the element tree to various concepts, hence giving a concrete meaning to the element itself. For example elements can be either concepts or individuals of a model. In this case, the domain modeling methodology needs to be instantiated. Once this process is chosen, a user can use the **activities** of this **process** like linking the element to a concept or to an individual of a domain model. For using any activities related to particular methodology, first the specific methodology needs to be instantiated.

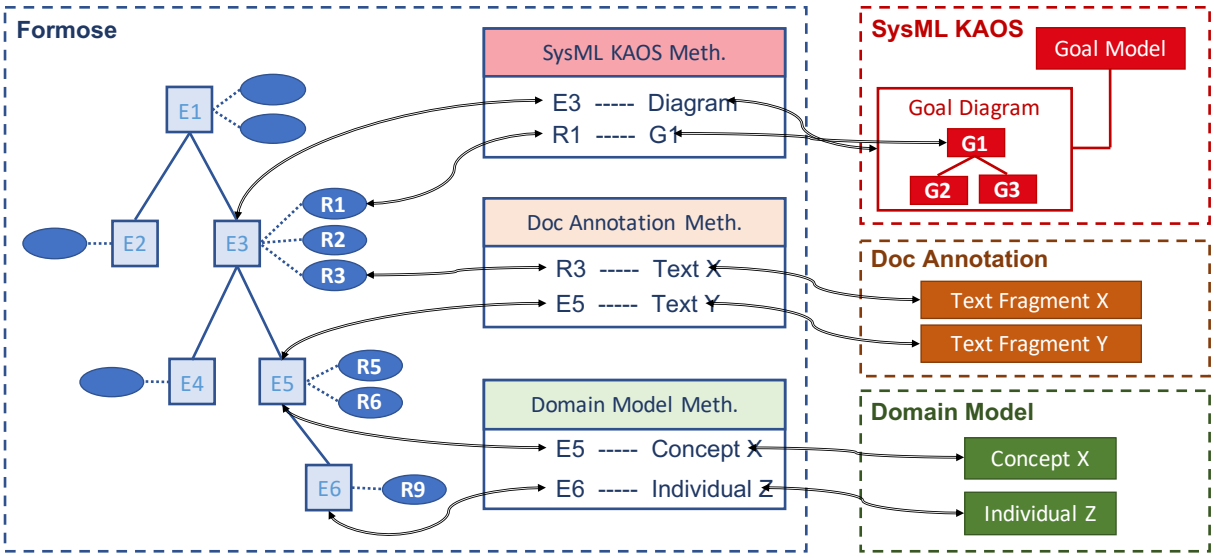

Figure 2: Process centric instantiations and federation using methodologies

3 FORMOSE Process

3.1 Formose high level process

In this project we used model federation approach in the context of requirements engineering for modeling formal requirements for critical and complex systems. The application of this approach is a stepwise approach, and till now we have integrated the requirements elicitation and analysis view, with SysML-KAOS and domain modeling views. However, process validation view is under development and would be integrated in the next development cycle. For these reasons, the high level process described in this deliverable does not take into account the process validation activities.

The low level process for each development project varies according to the specific settings and context of the project. For these reasons, we propose a highly flexible approach for the low-level process. However, this high level process gives an overall methodological perspective on the requirements development approach proposed by the FORMOSE project. Figure 3 explains the activities involved in the proposed methodology. This methodology is explained through the following activities:

1. *Gather information resources:* This **activity** involves the identification of possible information resources that can be used to elicit **requirements**. They can vary from early information resources like feasibility reports, standards and minutes of the meetings to late informations sources like deployment models, test plans, *etc.*

2. *Federate information resources:* Once the information resources are collected they need to be linked with the **requirements**. This is handled through the model federation approach where requirement models are developed as virtual models in the conceptual space and other models (information resources) are placed in their respective technological spaces.

3. *Identify/Elicit requirements:* Once the information resources are linked with the **requirements** model, we can identify the requirements from those information resources. For the

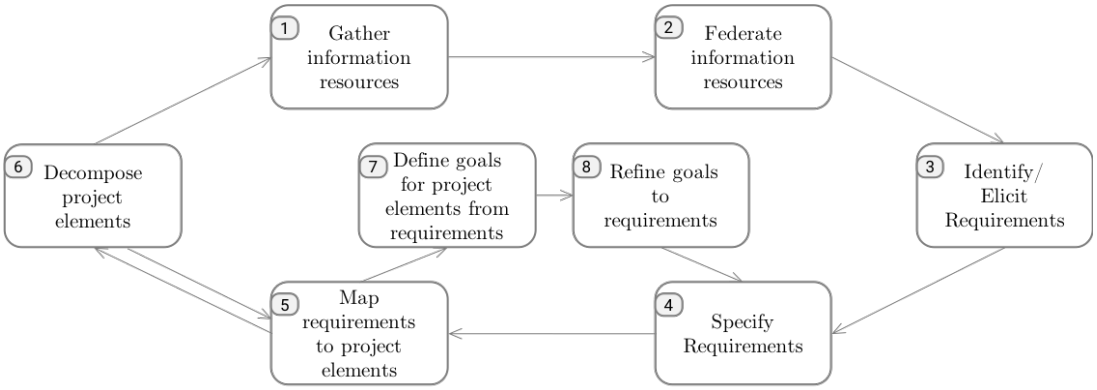

Figure 3: High level process

moment, we have used the technological space for MS Word documents to identify multiple fragments for possible requirements. Different documents are used for this purpose like project proposals, interview transcripts, feasibility reports, etc.

4. *Specify requirements:* Once the **requirements** are identified from multiple information resources, they are specified in the requirements model. In case the requirements are already specified, they are linked with the information resources for synchronization.

5. *Map requirements to project elements:* In this activity, the specified **requirements** are mapped to the project **elements**. Multiple requirements can be mapped to a single project element.

6. *Decompose project elements:* Multiple **requirements** mapped to a project **element** describe the expected functionality from that element. This allows the decomposition of the element into sub elements, such that each sub element takes care of a subset of the requirements mapped to its parent element.

7. *Define goals for project elements from requirements:* All the **requirements** associated with a project **element** are considered as **goals** in this **activity**. A goal model is developed for each of these goals.

8. *Refine goals to requirements:* All the **goal** models are refined to get **requirements** at the leaves of the goal models. These requirements can be specified in activity 4.

3.2 Formose low level process

The high level process for the proposed system explains the overall process from an abstract perspective. The actual working of the tool for the development of requirement models for complex systems requires multiple low level activities. These low level activities concern different views and modules of the proposed system. For example, Figure 4 shows some activities from FORMOSE Core, SysML-KAOS and B method. A process is hierarchical in nature, thus some of these activities might even contain more sub-activities to define the complete process to a task level.

The low level process proposed by the FORMOSE project makes sure not to constrain the user with a rigid pre-defined process. The activities provided at the disposal of the user can be

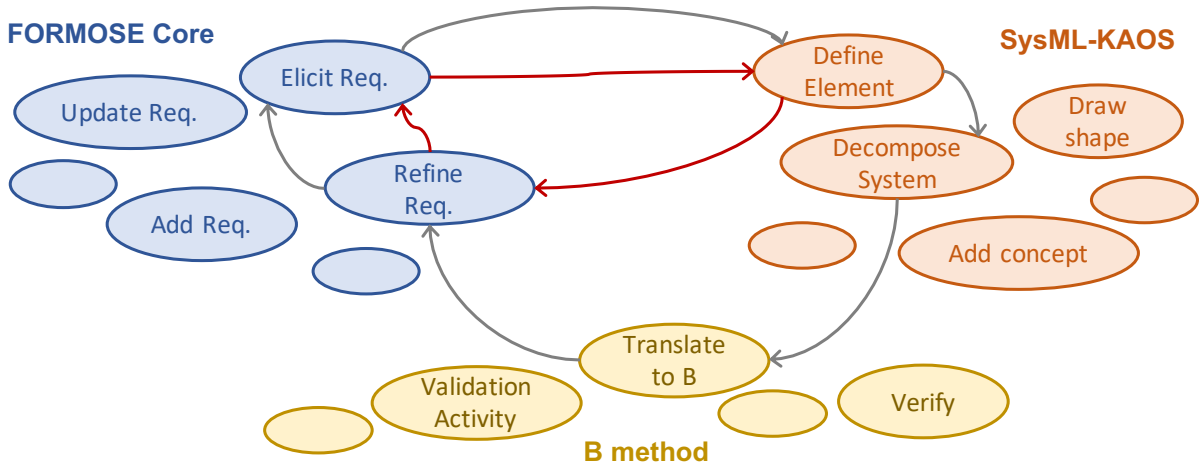

Figure 4: Low level process for the FORMOD tool

used for a customized process for each project. A natural dependency between the activities ensures that the overall process remains meaningful and objective. For example, a user can not refine an activity, if that activity is not elicited already. In multiple activities shown in Figure 4, a user can choose a low level process by using *elicit requirement*, *define element* and *refine requirement* activities. Using a cycle of these three activities (depicted by red arrows), a user can develop a process for the development of requirements that has corresponding elements defined for the SYSML-KAOS models. Another implementation could be to use the *decompose system* activity after the *define element* activity. This will allow decomposing the elements and then using the newly identified sub-elements for requirements refinement. The user is at liberty to engage further components of the methodology to the process. For example, it the user wants to translate a certain sub-element of the system into B-method, a corresponding activity can be added in the process to accomplish this.

3.3 Example process

This section presents an example process that can serve as a tutorial for the FORMOD tool. The idea behind presenting this example process is not to provide a comprehensive process, it is rather targeted towards demonstrating different activities that can be performed with this tool. We use the landing gear case study[2] as the domain document, from which we will demonstrate the process of requirements elicitation and refinement. We used the *.docx format of this document for this demonstration, where the first half of the document is saved as LG1.docx and the second half as LG2.docx.

Project creation

1. Launch the FORMOD tool.
2. In the dialog box (as shown in Figure 5), select the formose tool icon and click the "new project" button. Name this new project, "Landing Gear".

[2]Boniol, Frédéric, and Virginie Wiels. "The landing gear system case study." International Conference on Abstract State Machines, Alloy, B, TLA, VDM, and Z. Springer, Cham, 2014.

3. Gives the Formose nature to your newly created project.

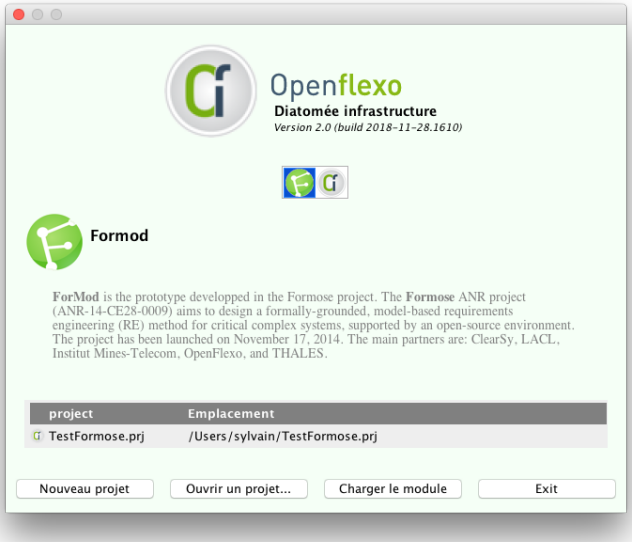

Figure 5: Dialog box

Federating information resources

3. Once the tool loads the new project, screen shown in Figure 6 opens.
4. We can see multiple perspectives on the top right corner of the tool. Amongst these perspectives (shown in Figure 7), the first one is management perspective, then the document annotation perspective, then the domain model perspective, the SysML-KAOS perspective and finally the B perspective. Choose the document annotation perspective.
5. In Document annotation perspective load the landing gear files (LG1.docx and LG2.docx). To do this, click on the green '+' button (or in the button in the bottom of panel labelled "Import docx document") and then browse to the files one by one. You shoud obtain this screen (Figure 8).

Requirements elicitation

5. Identify three elements from the section 2 of LG1 *i.e.* Mechanical part, Digital part, Pilot interface. The software resides in the *digital part*, so we select that element.
6. Now identify three requirements from the section 4 of LG2. To do this select the concerned portion of the text in the document and then use the "identify requirements" button in the document annotation perspective for each of the following requirements.
   - Controlling hydraulic devices
   - Monitoring the system
   - Informing the pilot

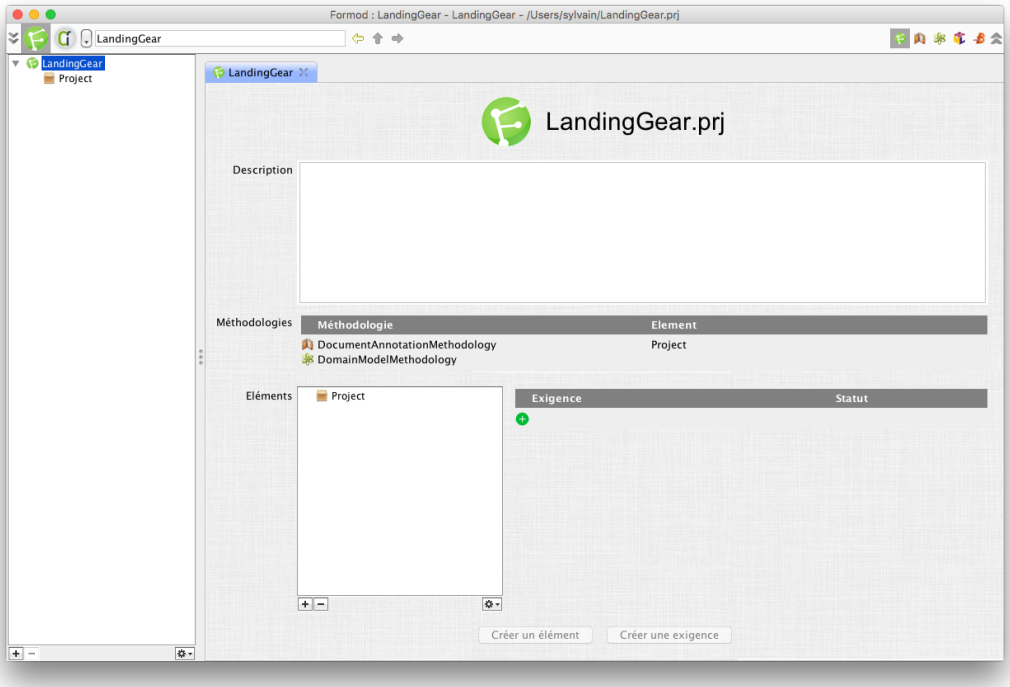

Figure 6: Formod main panel

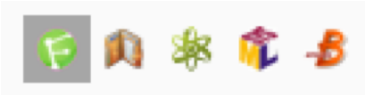

Figure 7: Perspectives of FORMOD tool

7. Select the “Controlling hydraulic devices” requirement amongst the newly elicited requirements in the requirements browser in the requirements browser pane. Select the fourth paragraph of section 1 in LG1, and identify it as a text fragment. This shall link this text to the selected requirement.

Requirements refinement

8. After reading the identified text of the ”control hydraulic devices”, we plan to refine it to the two controls: landing and retraction. Switch to the SysML KAOS perspective. You shall notice that the project tree in the left-side explorer pane is greyed out. This is a normal behavior as no methodology is activated for any of the elements yet.
9. Right click on the “digital part” element and select “Instantiate sysml kaos methodology”.
10. A blank goal diagram should open. Note that existing requirements (elicited in annotation perspective) are shown.
11. Drag existing “Controlling hydrolic devices” in blank goal diagram.

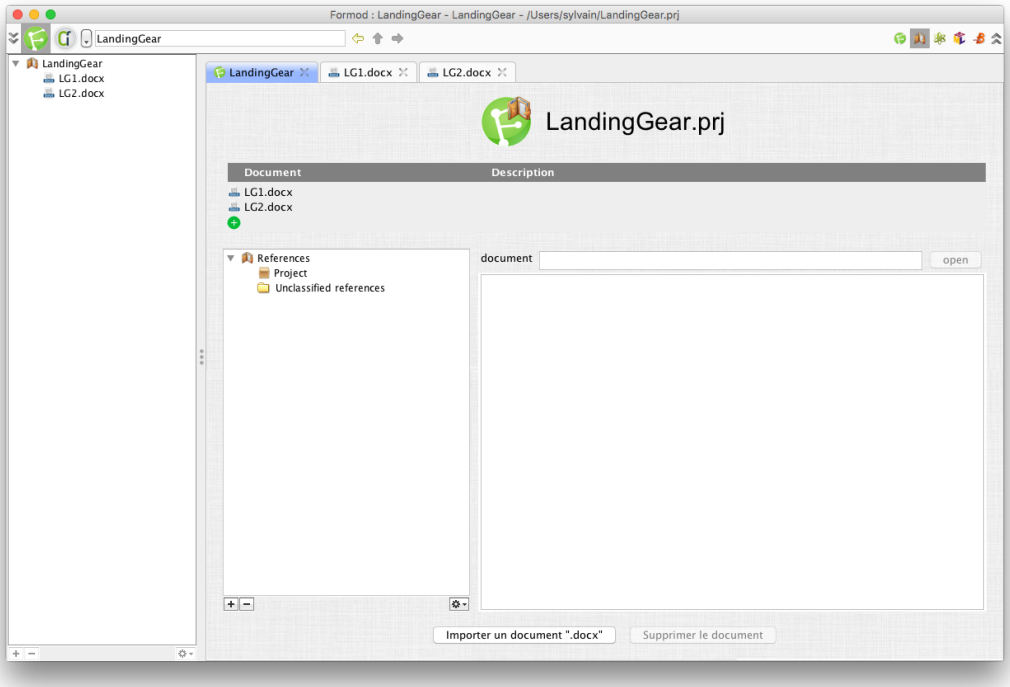

Figure 8: Document annotation perspective

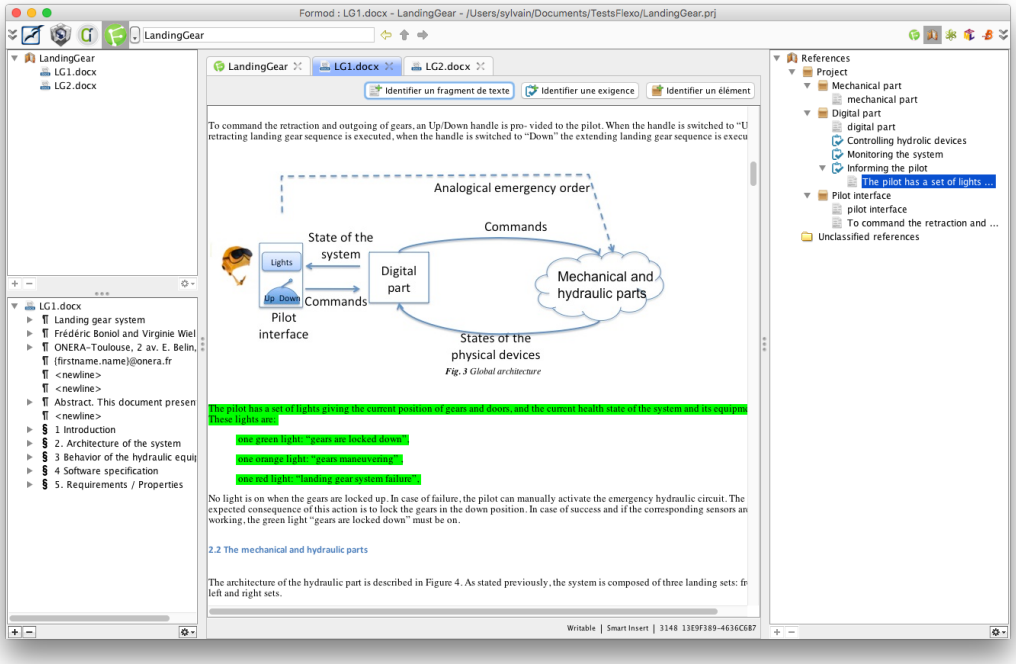

Figure 9: Requirements elicitation in text documents

12. Refine this goal by creating the following goals: "Control landing sequence" and "Control retraction sequence".
13. Link the first child goal to the parent goal, it shall create a “AND” link.
14. Link the second child goal to the ”AND” refinement Link.

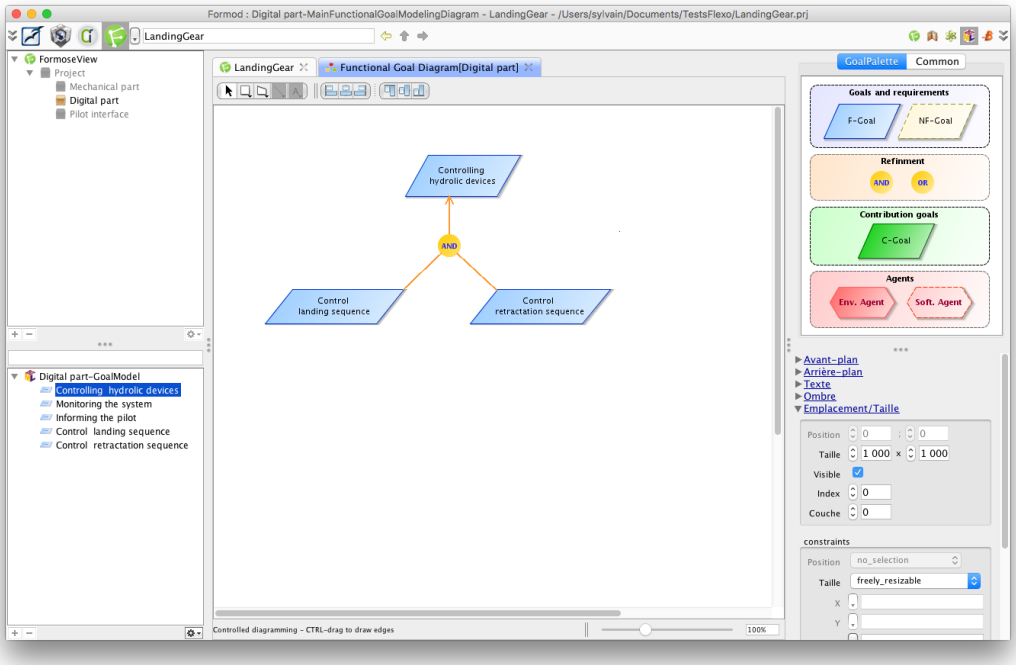

Figure 10: Requirements refinement in KAOS goal diagrams

Requirements justification

15. Switch back to document annotation view by selecting the LG1 tab. You should notice that the two new requirements have appear in that view: "Control landing sequence" and "Control retraction sequence"
16. Select the ”Control Landing Sequence” requirement in the right pane
17. Select the first sentence of the 4th paragraph of section 1 and select “Identify text fragment”. This gives justification for the requirement. Do the same for retraction sequence requirement in the same paragraph.

Goal modeling

18. Go to SysML KAOS perspective and refine the model creating the following 4 subgoals (Open the doors, Extend landing gears, Retract landing gears, Close the doors).

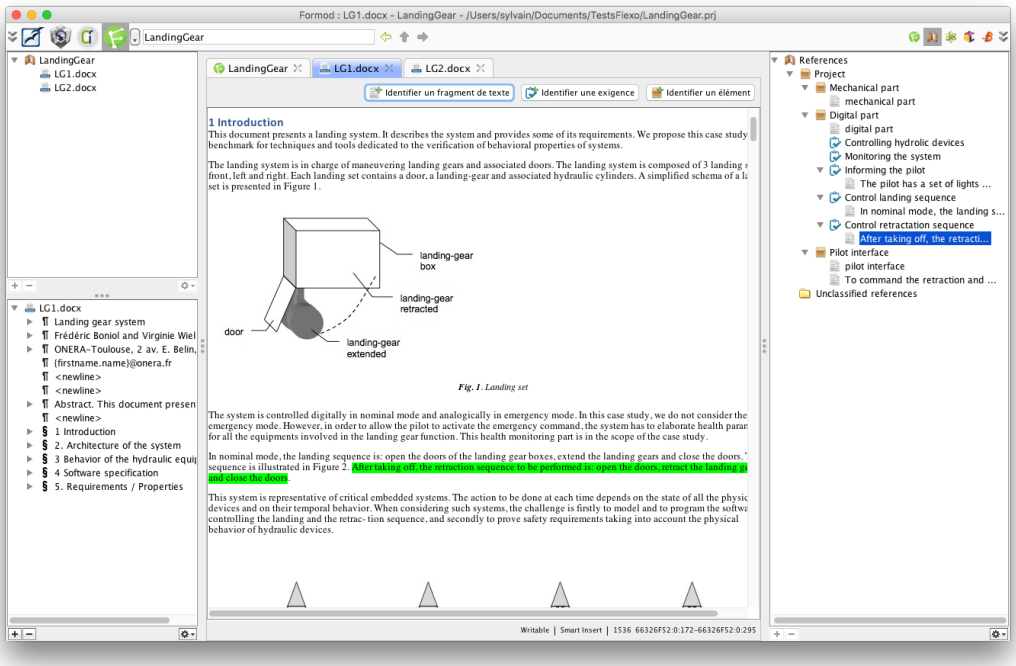

Figure 11: Requirements justification in text documents

19. The justifications for these requirements are already associated with the parent goals. Optionally, you can further associate the justifications from the lists available in LG2, section 4.1.

20. You can associate the agents (General EV[3], Gear Outgoing EV, Gear retraction EV, Door closure EV, Door opening EV) to the goals from SysML KAOS perspective, as shown in Figure 12 [4].

21. In the SysML KAOS perspective, right click on the “Digital part” element from the left side project explorer and select “Refine using agent allocation”. This shall create new goal diagrams for that element. And allow further refining of the requirements.

[3] EV in this case study stands for electric values.

[4] The requirements R11 and R12 in this figure are shown with their ids (even though they are not shown in the tool in this manner), because this id is coming from the requirement document. For other requirements, the user is free to assign an appropriate id

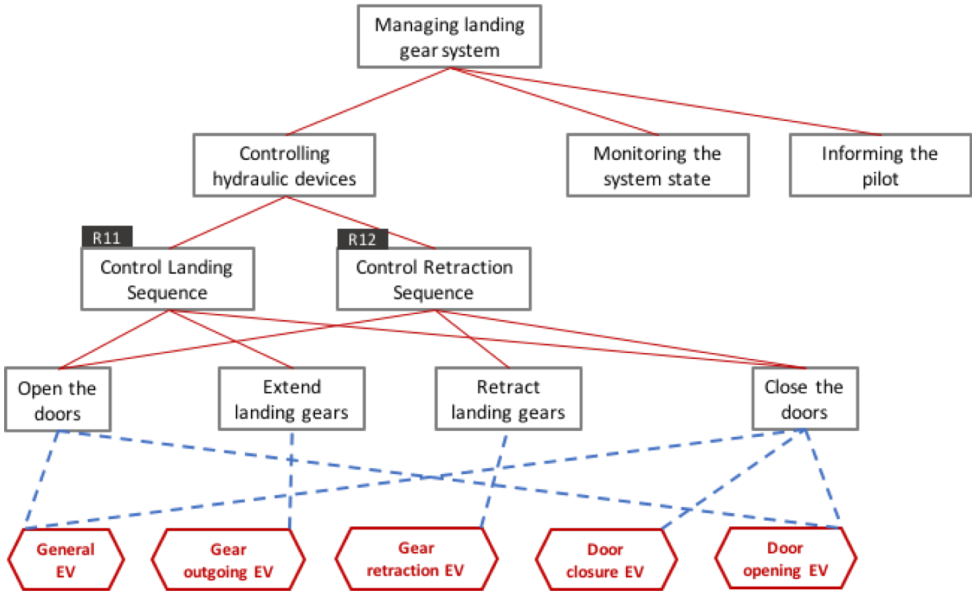

Figure 12: Landing gear goal model

4 Summary

This deliverable explains the first version of the proposed process of the development of formal requirement models for critical and complex systems. It explains the underlying process for the proposed tools and methods and the way models are federated.

The tool itself is explained and presented in another deliverable. So the focus of this deliverable is not to explain the working of the tool, rather to explain the activities that could be used for the process. The process is explained at two different abstractions: a high level process that governs that overall methodologies used by the proposed systems and a low level process that is a flexible set of activities used for the requirements engineering process. The deliverable presents an example process to demonstrate the application of the tool, using the process.