PROPRIETARY RIGHTS STATEMENT

THIS DOCUMENT CONTAINS INFORMATION, WHICH IS PROPRIETARY TO THE CRYSTAL CONSORTIUM. NEITHER THIS DOCUMENT NOR THE INFORMATION CONTAINED HEREIN SHALL BE USED, DUPLICATED OR COMMUNICATED BY ANY MEANS TO ANY THIRD PARTY, IN WHOLE OR IN PARTS, EXCEPT WITH THE PRIOR WRITTEN CONSENT OF THE CESAR CONSORTIUM THIS RESTRICTION LEGEND SHALL NOT BE ALTERED OR OBLITERATED ON OR FROM THIS DOCUMENT. THE RESEARCH LEADING TO THESE RESULTS HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S SEVENTH FRAMEWORK PROGRAM (FP7/2007-2013) FOR CRYSTAL – CRITICAL SYSTEM ENGINEERING ACCELERATION JOINT UNDERTAKING UNDER GRANT AGREEMENT N° 332830 AND FROM SPECIFIC NATIONAL PROGRAMS AND / OR FUNDING AUTHORITIES.

CRYSTAL

CRitical S**YST**em Enginieering **A**cce**L**eration

CRYSTAL aerospace use case description Report – V1

D208.010

DOCUMENT INFORMATION

Project	CRYSTAL
Grant Agreement No.	ARTEMIS-2012-1-332830
Deliverable Title	CRYSTAL aerospace use case description
Deliverable No.	D208.010
Dissemination Level	Public
Nature	R
Document Version	V1.00
Date	2014-01-30
Contact	Claudio Pessa
Organization	ALA
Phone	+39 0117562678
E-Mail	claudio.pessa@alenia.it

AUTHORS TABLE

Name	Company	E-Mail
Andreas Mitschke	EADS IW G	Andreas.mitschke@eads.net
Luciana Lo Verde	ALA	Luciana.loverde@alenia.it
Wladimir Schamai	EADS IW G	Wladimir.schamai@eads.net
Stefan Richter	EADS IW G	stefan.richter@eads.net
Claudio Pessa	ALA	claudio.pessa@alenia.it
Ivo Viglietti	ALA	ivo.viglietti@alenia.it
Francesco Brunetti	POLITO	francesco.brunetti@polito.it

REVIEW TABLE

Version	Date	Reviewer
0.09	10.01.2014	Andreas Mitschke - Airbus IW
0.10	24.01.2014	Thomas Kuhn – IESE; Gray Bachelor - IBM
1.00	30.01.2014	final version

CHANGE HISTORY

Version	Date	Reason for Change	Pages Affected
0.01	15.10.2013	Initial version	All
0.05	23.10.2013	Adding of TBD	All
0.09	10.01.2014	Partner contribution	All
0.10	24.01.2014	Introduction of int. reviewer input	All
1.00	30.01.2014	Introduction of ext. reviewer input	minor changes

CONTENT

Contents

Content of Figures

1 Introduction

1.1 Role of deliverable

This document provides a description of the CRYSTAL Public Aerospace Use Case. The CRYSTAL Public Aerospace Use Case has the following major objectives:

- Describing typical aerospace engineering challenges with respect to (tool) interoperability, in order to:
  - Help SP2 Use Cases refinement
  - Help IOS Working Group and SP6 Bricks provider to get a first understanding of typical aerospace (and other domain) needs
  - Help creating synergies between tools providers, academics and industry
- Performing a prototyping of IOS Concept, in order to:
  - Refine and validate the feasibility and value of the CRYSTAL interoperability approach
  - Show the main “idea” behind the CRYSTAL approach
  - Demonstrate CRYSTAL IOS Concept
- Facilitate the presentation of CRYSTAL results in publications without facing IPR concerns, in order to support dissemination activities.

1.2 Relationship to other CRYSTAL Documents

The CRYSTAL Public Aerospace Use Case description is intented to be used as input for technical Bricks provider managed by SP6.

1.3 Open Points

N/A

1.4 Structure of this document

The CRYSTAL Public Aerospace Use Cases focusses on the design, analysis and specification of a regional aircraft de-icing system. Chapter 2 provides an overview of this system itself, and describes the process steps that are in scope of the use case. Chapter 3 provides a description of the engineering methods that shall be applied to the use case. Chapter 4 provides an overview of the current technical implementation of the use case as of January 2014. Chapter 5 provides some information about the data that has been defined for the use case. The Annex includes a Meeting Minutes for priorisation of engineering methods, and further detailed descriptions of engineering methods described in Chapter 3.

2 Use Case Description

The use case focuses on the design, analysis, and specification of a de-icing system of a regional turboprop aircraft.

2.1 Overview

The purpose of the de-icing system is to prevent the creation of ice on safety critical components of the aircraft, such as sensors, wings, engines, or ogives. There are many different alternative concepts to provide this de-icing capability. For example, pneumatic boots can be used that break ice through permanent inflation and deflation. Alternatively, bleed air can be used to melt ice through aerothermal effects. Other alternative concepts include the use of electrical heating or of de-icing fluids or electro-impulsive technologies. In Figure 2-1 different solution are shown for de-ice wing and tail .

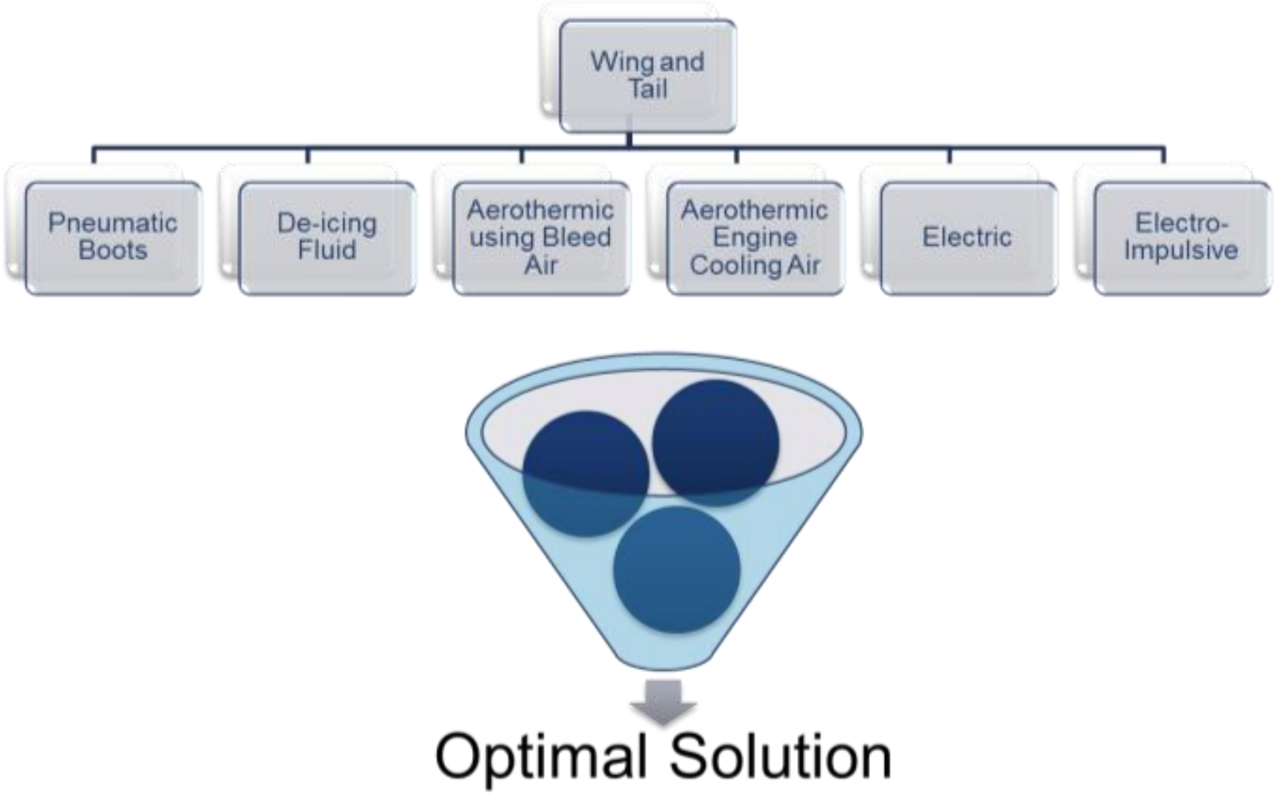


Figure 2-1: Possible alternative de-icing concepts on wing and tail

The main challenge for engineers is to find the “best” design solution for the de-icing system. It means that engineers have to select a de-icing concept and – based on this - define a system architecture that fulfils the functional need of providing the de-icing capability, that satisfies all safety related constraints, and that has the least operational and production cost and the best performance against other relevant criteria.

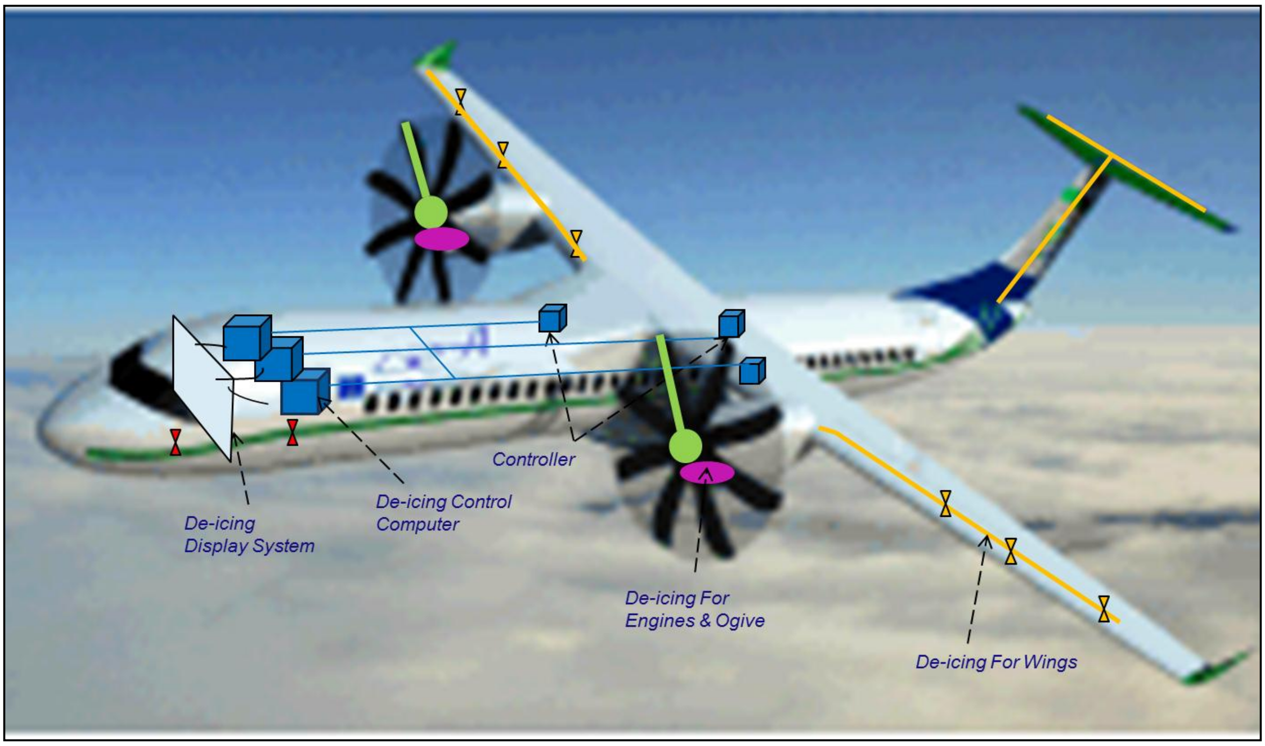


Figure 2-2: Overview De-icing System

For the Public Aerospace Use Case, we will show how enhanced engineering methods can be used to support such kind of “trade-off-analysis” challenges related to the design, specification and analysis of safety critical systems. By concentrating on interoperability related challenges of engineering methods, we provide input for the definition of the CRYSTAL interoperability specification.

2.2 Use Case Process

As shown in Figure 2-3, the process steps covered by WP208 will be limited to design, specification and analysis activities and exclude any real physical development or testing of de-icing system components. Concretely, we address the following main activities on OEM (Original Equipment Manufacturer) and supplier sides:

- Definition and analysis of Requirements (OEM)
- Pre-liminary concept definition and pre-liminary trade-off analysis (OEM)
- Detailed concept definition and trade-off analysis (OEM)
- Preparation of a system specification for supplier (OEM)
- Definition and trade-off analysis of sub-system components based on specification (supplier)

Figure 2-3: Process steps covered by WP208

3 Identification of Engineering Methods

3.1 Overview

One of the main objectives of WP208 is to describe typical aerospace engineering challenges with respect to interoperability. Taking into account this objective, within WP208 we have defined a set of engineering methods that are described in the following chapters.

Some of the engineering methods mentioned herafter apply at a single process step only, while other methods can be used at multiple or at all process steps (see Figure 3-1).

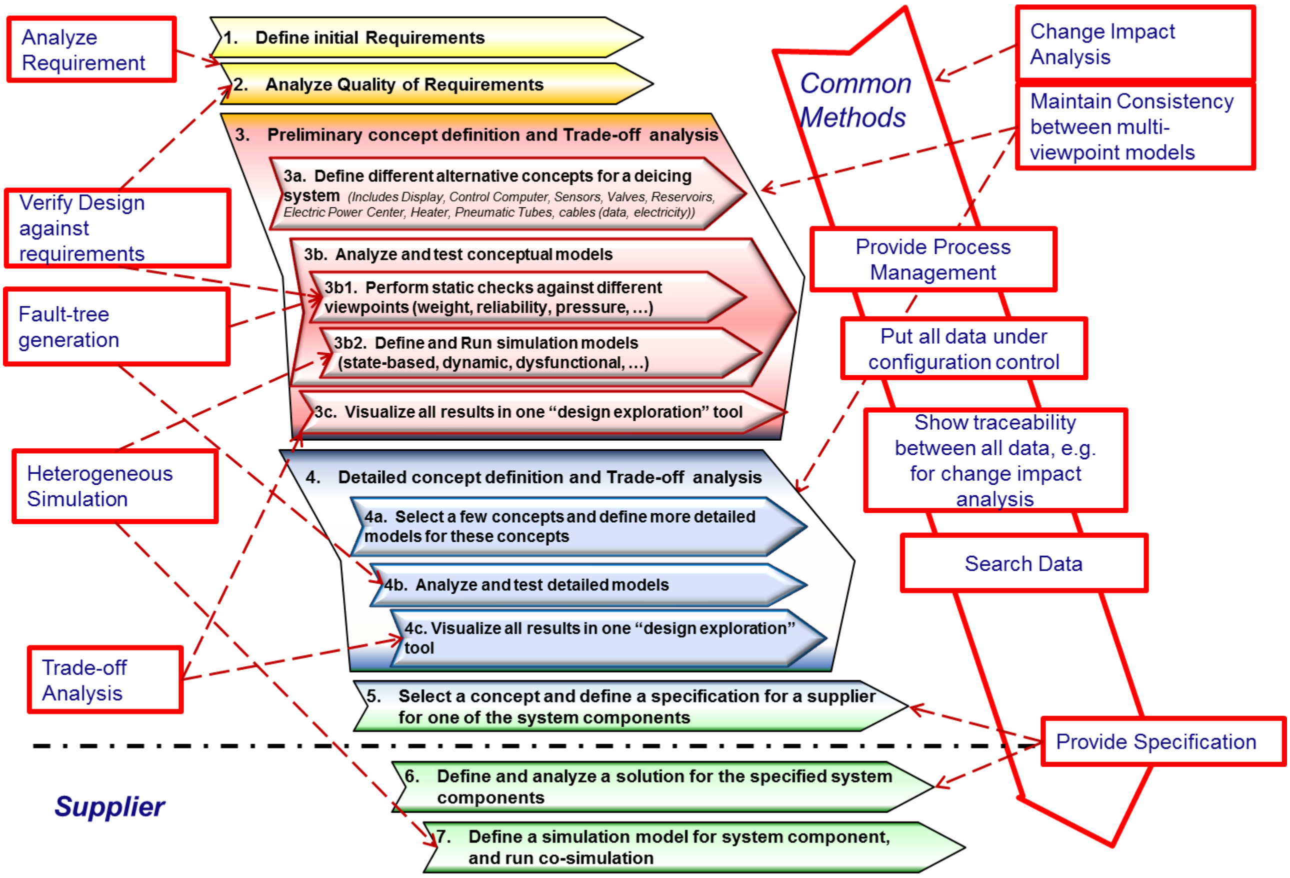


Figure 3-1: Relation Process Steps to Engineering Methods

Figure 3-2 includes the current list of engineering methods defined for WP208, and illustrates the expected relationship between WP208 and SP6. As shown, the engineering methods described hereafter shall be used as input for the definition the Interoperability Specification and for SP6 Bricks development. Brick Implementations defined within SP6 shall then be integrated into the WP208 System Engineering Environment (SEE).

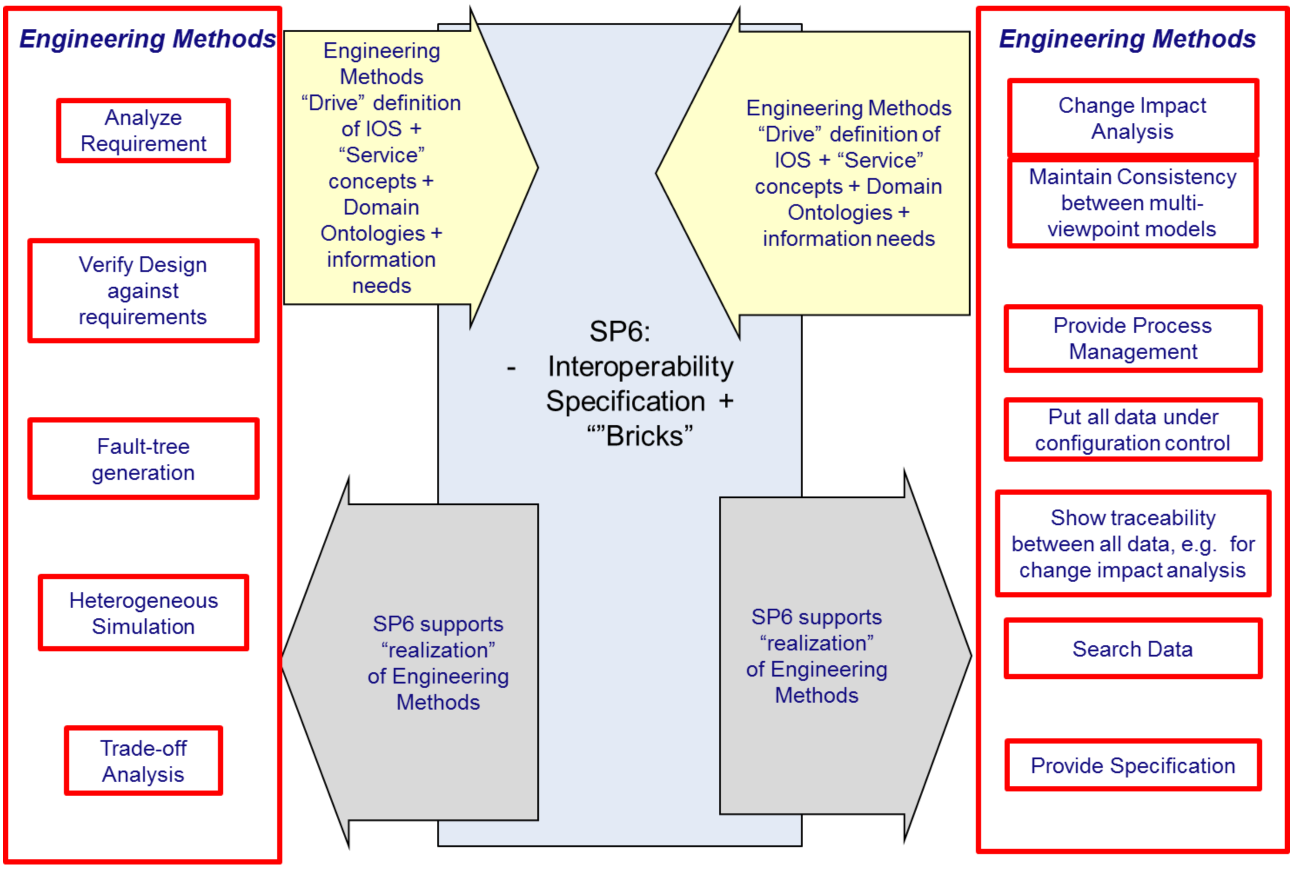


Figure 3-2: Relation WP208 Engineering Methods with CRYSTAL SP6

The Engineering Methods described hereafter have been proposed initially by the core partners of WP208. As WP208 is supposed to cover topics that are representative for the Aerospace Domain, the engineering methods have been reviewed and prioritised by SP2 partners at a dedicated meeting on November 12th, 2013. The minutes of this meeting can be found in the annex of this deliverable. This version of the deliverable also includes the feedback received by participants of that meeting.

3.2 Method "Analyse Requirements"

The general purpose of this engineering method is to improve the quality of requirements defined in a textual format. It is assumed that at least 2 tools are involved, one tool for managing the textual requirements under configuration, and another tool for analysing the quality of the textual requirement. The interoperability need in this scenario therefore concerns the interaction between the requirements management and the requirements quality analysis tool.

Engineering Method: UC208_AnalyzeRequirement_001					
Purpose: The Requirement Engineer wants to check quality of a requirement using RQS					
Comments:					
Pre-Condition		**Engineering Activities (made of steps)**		**Post-Condition**	
Requirements are stored in Doors D/B Requirement characteristics are stored in Doors D/B Requirement quality information is stored in other D/B		1. In RQS, launch service “Get List of requirements”(*The service should allow to filter requirements according to characteristics (e.g. show all requirements that refer to “weight”)* 2. Request is forwarded to Doors D/B 3. List of all requirements is assembled and send back to RQS tool 4. In RQS, receive requirements 5. In RQS, select the requirement to be analyzed in detail, and launch service “Get Requirement” 6. Request is forwarded to Doors D/B 7. Identify and send Requirement to RQS 8. In RQS, analyze requirement. Afterwards, send back (service “send requirement”) 9. In RQS, select the new created requirement to be sent and stored in DOORS, and launch service “Send Requirement” 10. Requirements is sent to DOORS D/B 11. The original requirement is updated (new version) in the DOORS module accordingly.		Original Requirement has been approved or modified.	
Notes: Artifacts provided as input of the activity Requirement with appropriate detailed description				Notes: Updated Requirement with appropriate detailed description	
Artefacts Required as inputs of the Activities		**Artefacts used internally within the Activities (optional)**		**Artefacts Provided as outputs of the Activities**	
Name	Requirement	Name	DOORS internal Requirement	Name	Requirement
Generic Type: (Tool or language independend type)	Natural Language Requirement	Type:	Requirement as stored in DOORS D/B	Generic Type: (Tool or language independend type)	Natural Language Requirement

Required Properties: (Information required in interactions between steps)	- Requirement ID - Requirement Statement in natural language - Requirement Version	Properties:	TBD	Provided Properties: (Information provided in interactions between steps)	- Requirement ID - Requirement Statement in natural language - Requirement Version - Reviewing state (e.g., "checked", "unchecked" state to be set by the RQA Requirement Engineer) - Requirement Type (e.g., simply encoded in a string, "weight", "safety", "maintenability", "functional", etc).
Description & Interoperability Additional constraints: For the first steps of the activity, basic properties are required by the Requirement Engineer in order to pick up the ones he/she is interesting in analysing.		Description:		Description & Interoperability Additional constraints:	
		Name	RQS internal Requirement		
		Type:	Requirement as stored in RQS		
		Properties:	TBD		

Additional information about this scenario can be found in the Annex of this deliverable.

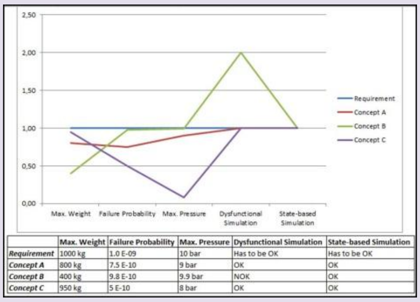
3.3 Method "Trade-off Analysis 001"

The general purpose of this engineering method is to compare different given system concepts with each other. This engineering method can occur at different phases of the system development process, such as preliminary concept evaluation or detailed concept definition.

It is assumed that several modelling and simulation tools are involved, each tool providing models for a different viewpoint for each of the alternative system concepts.

To compare the system concepts with each other, the relevant metrics for comparison have to be identified first. Then, the models describing the system concepts must be identified, and for each type of metric the corresponding values must be extracted from the right model (e.g. through simulation or calculation).

Engineering Method: UC208_Trade-Off Analysis_001		
Purpose: The System Architect of the De-icing system wants to evaluate different alternative de-icing system concepts		
Comments: The concepts are described by many different models, each representing one or several viewpoints		
Pre-Condition	**Engineering Activities (made of steps)**	**Post-Condition**

The alternative concepts for the de-icing system are described by many different models, each representing one or several viewpoints	1. The System Architect defines the metrics that are important for assessing a de-icing system concept (e.g.: weight, failure probability, max. pressure, max. response time, etc.) 2. System Architect launches request “Get Constraints” (*The service should allow to filter constraints according to characteristics (as in EM “Analyse Requirements”*) 3. Request is transferred to DOORS 4. Constraints for the de-icing system are sent back (or shown) 5. System Architect launches request “Assemble analysis results” 6. Request is transferred to tools that are storing data that is relevant to assess the de-icing model “Concept A”, “Concept B”, and “Concept C” 7. Tools are launching simulations and calculations 8. Tools are sending simulation and calculation results to Trade-off Tool	All simulation and calculation results are presented to the System Architect. Example: 
Notes:	Notes:	Notes:

	Max. Weight	Failure Probability	Max. Pressure	Dysfunctional Simulation	State-based Simulation
Requirement	1000 kg	1.0 E-09	10 bar	Has to be OK	Has to be OK
Concept A	800 kg	7.5 E-10	9 bar	OK	OK
Concept B	400 kg	9.8 E-10	9.9 bar	NOK	OK
Concept C	950 kg	5 E-10	8 bar	OK	OK

Artefacts Required as inputs of the Activities		**Artefacts used internally within the Activities (optional)**		**Artefacts Provided as outputs of the Activities**	
Name	Requirements	Name		Name	Trade-off Analysis Results
Generic Type: (Tool or language independend type)	Requirements in natural language format	Type:		Generic Type: (Tool or language independend type)	Comprehensive Representation of simulation and calculation results for each alternative de-icing system concept against pre-defined metrics
Required Properties: (Information required in interactions between steps)	Values for metrics that are constraining the de-icing system concepts, such as maximum weight, cost, failure propabilities, pressure values, required time for de-icing	Properties:	Values for metrics that are constraining the de-icing system concepts, such as maximum weight, cost, failure propabilities, pressure values, required time for de-icing	Provided Properties: (Information provided in interactions between steps)	Required Values (from Requirements) per Metric Provided Values (from different models) for each alternative system concept
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Concept Model	Name		Name	
Generic Type: (Tool or language independend type)	Logical Architecture Model with State-based behavior	Type:		Generic Type: (Tool or language independend type)	

Required Properties: (Information required in interactions between steps)	State-based Simulation results + some simple static parameters such as purchase cost	Properties:	State-based Simulation results + some simple static parameters such as purchase cost	Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Safety Model	Name		Name	
Generic Type: (Tool or language independend type)	Safety Model for Failure Rate Calculation and for dysfunctional Simulation	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Dysfunctional Simulation result + overall system failure rates (for loss and erroneous)	Properties:	Dysfunctional Simulation result + overall system failure rates (for loss and erroneous)	Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Weight Model	Name		Name	
Generic Type: (Tool or language independend type)	Weight Model	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Overall System Weight	Properties:	Overall System Weight	Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Physical Behavior Model	Name		Name	
Generic Type: (Tool or language independend type)	Physical Behavior Model (e.g. in Modelica or Matlab/Simulink)	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Resulting values for physical behavior simulation - this could be e.g. resulting ice accretion in mm, time for ice elimination in seconds, amount of consumed "goods"	Properties:	Resulting values for physical behavior simulation - this could be e.g. resulting ice accretion in mm, time for ice elimination in seconds, amount of consumed "goods" for ice elimination in kg, liter, or kWh, and	Provided Properties: (Information provided in interactions between steps)	

	for ice elimination in kg, liter, or kWh, and many more.		many more.		
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	

Additional information about this scenario can be found in the Annex of this deliverable.

3.4 Method "Trade-off Analysis 002"

The general purpose of this engineering method is to compare different given system solutions once the concept is defined. This engineering method is usually called after having carried out the preliminary analysis of different concepts according the evaluation criterias which are taken into account by the engineering method Trade-off Analysis 001.

The main aspects to be investigated are the safety and the availability of the system, therefore the analysis are focused on the redundancy characterics of the architecture and the reliability performance of the components (MTBF, MTBCF parameters…).

OEM V&V Engineer wants to run simulation of the de-icing system functional model using Rhapsody Models that are stored in the Rhapsody DB. In order to have a complete functional architecture some analyses have to be performed in PTC, then the RBD, FMEA/FMES and Reliability Prediction data are stored in PTC D/B and the result are reported in the Rhapsody functional model.

Engineering Method: UC208_Trade-Off Analysis_002		
Purpose: The System Architect / RM&T Expert of the De-icing system wants to evaluate different alternative solutions of de-icing system once the candidate concept solution is selected (i.e. pneumatic system). The Safety Expert want to make consistent the functional behavior built in Rhapsody (Rainy day analysis) to the Disfunctional analysis carried out in "safety discipline" tools.		
Comments: The candidate concept is already described by a preliminary functional model. This needs to be refined in terms of Rainy Day scenarios / Failure events that are consistent with the safety analysis.		
Pre-Condition	**Engineering Activities (made of steps)**	**Post-Condition**
The candidate concepts for the de-icing system is described by the functional model.	1. System Architect / Safety Expert launch service "Request list of available simulation model" in Rhapsody tool. System Architect / Safety Expert launch service "Simulate the selected model" in Rhapsody 2. Safety expert call part of the Model built in Rhapsody and this one is forwarded to PTC and stored In the DB. 3. PTC send back list of analyses results to Rhapsody 4. Results are reported in the Rhapsody functional model	All simulation and calculation results are presented to the System Architect / RM&T.
Notes:	Notes:	Notes:
Artefacts Required as inputs of the Activities	**Artefacts used internally within the Activities**	**Artefacts Provided as outputs of the Activities**

		(optional)			
Name	De-icing system Model	Name		Name	
Generic Type: (Tool or language independend type)	Logical Architecture Model with state-based behaviour	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	State-based Simulation results + some component attribute values	Properties:		Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:				Description & Interoperability Additional Constraints:	
Name	De-icing System Safety Model + related data object	Name		Name	
Generic Type: (Tool or language independend type)	Safety Model for Failure Rate Calculation and dysfunctional analysis	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Dysfunctional analysis results + overall system failure rates (for loss and erroneous)	Properties:		Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	

3.5 Method "Verify Design against Requirements"

The general purpose of this engineering method is to highlight if a given system concept does violate any of the system requirements.. This engineering method can occur at different phases of the system development process, such as preliminary concept evaluation or detailed concept definition.

It is assumed that several modelling and simulation tools are involved, each tool providing models for a different viewpoint.

To ensure that requirements are not violated, the models describing the system concepts must be identified, and for each type of requirement the corresponding values must be extracted from the right model (e.g. through simulation or calculation).

Engineering Method: UC208_VerifyDesignAgainstRequirements_001		
Purpose: The Requirements Engineer wants to check if a Design alternative meets a set of given requirements		
Comments:		
Pre-Condition	**Engineering Activities (made of steps)**	**Post-Condition**

Requirements constraining the de-icing system have been defined. Models describing the design alternative with information about components weight and pressure have been defined.	1. In a Design Exploration tool, the required weight and pressure values are requested 2. Request is forwarded to Doors 3. System Weight and Pressure Requirements are send back 4. In the Design Exploration Tool, service “request System Weight” is launched by Requirements Engineer 5. Request is forwarded to Rhapsody 6. System Weight is calculated 7. System Weight result is sent back 8. In Design Exploration tool, service “request maximum pressure in System” is launched by Requirements Engineer 9. Request is sent to Rhapsody 10. Rhapsody sends model to Dymola for simulation 11. In Dymola, pressure simulation model is executed to determine max. pressure 12. In Dymola, the maximum pressure value as resulting from the simulation is sent back 13. Results are sent to Design Exploration Tool 14. In Design Exploration tool, the system weight and pressure requirements are compared with calculation and simulation results	As a result of the verification of Design against weight and pressure requirements, a verification case is created and linked to the requirements
Notes:	Notes:	Notes:

Artefacts Required as inputs of the Activities		**Artefacts used internally within the Activities (optional)**		**Artefacts Provided as outputs of the Activities**	
Name	Requirements	Name		Name	Verification with status "Passed" or "Failed" linked to each requirement.
Generic Type: (Tool or language independend type)	Requirements in natural language format	Type:		Generic Type: (Tool or language independend type)	Link
Required Properties: (Information required in interactions between steps)	Values for metrics that are constraining the de-icing system concepts, such as maximum weight, or max. allowed pressure values for de-icing fluid reservoir.	Properties:	Values for metrics that are constraining the de-icing system concepts, such as maximum weight, or max. allowed pressure values for de-icing fluid reservoir.	Provided Properties: (Information provided in interactions between steps)	Status "Passed" or "Failed" for each requirement.
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Weight Model	Name		Name	
Generic Type: (Tool or language independend type)	Weight Model	Type:		Generic Type: (Tool or language independend type)	

Required Properties: (Information required in interactions between steps)	Overall System Weight	Properties:	Overall System Weight	Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Physical Behavior Model	Name		Name	
Generic Type: (Tool or language independend type)	Physical Behavior Model (e.g. in Modelica or Matlab/Simulink)	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Resulting values for physical behavior simulation - this could be e.g. Resulting pressure value for a de-icing fluid reservoir.	Properties:	Resulting values for physical behavior simulation - this could be e.g. Resulting pressure value for a de-icing fluid reservoir.	Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	

Additional information about this scenario can be found in the Annex of this deliverable.

3.6 Method "Heterogenous Simulation"

The general purpose of this engineering method is to assess system concepts through simulation. This engineering method can occur at different phases of the system development process, such as preliminary concept evaluation or detailed concept definition. Please note that the tools mentioned hereafter are listed for better illustration of the scenario. The list of tools and the order in which they are used is not fixed.

It is assumed that several modelling and simulation tools are involved, each tool providing models in a specific formalism that contribute to the envisaged simulation. To run the simulation, the different simulation models must be identified and then integrated with each other.

Engineering Method: UC208_HeterogeneousSimulation_001		
Purpose: OEM V&V Engineer wants to run simulation of the de-icing system		
Comments:		
Pre-Condition	**Engineering Activities (made of steps)**	**Post-Condition**

He uses the tool DYMOLA for simulation Models required for simulation are stored in DYMOLA, but also in other tools such as Rhapsody or Simulink. Some Models are managed by the supplier The missing model is stored in Simulink. For simulation, he needs a “black-box” version of this model (FMI/ FMU)	1. In Dymola, launch service “Request list of available simulation model” 2. Request is forwarded to other tools 3. Rhapsody and Simulink send back list of available simulation models 4. In Dymola, V&V Engineers receives list of available simulation models. He selects the model from Simulink. 5. In Dymola, after selecting the model from Simulink, launch service “Get Simulation Model” 6. Request is forwarded to other tools 7. In Simulink, send back the requested simulation model 8. In Dymola, run combined simulation with simulation model from Simulink	Simulated de-icing system
Notes: List of tools not fixed	Notes: Alternatively, the simulation models are not send to Dymola, but run directly in the other simulation models at runtime.	Notes:

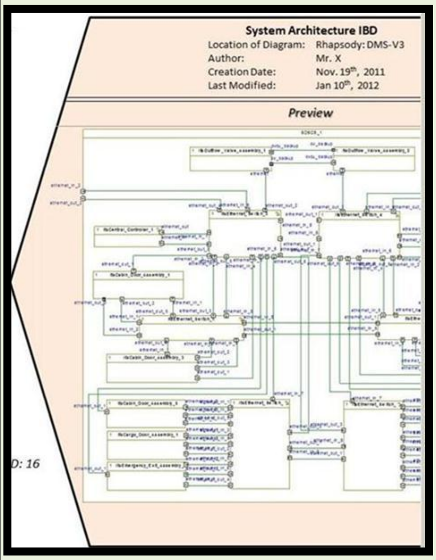
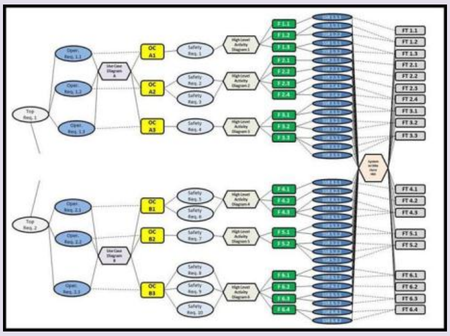
Artefacts Required as inputs of the Activities		**Artefacts used internally within the Activities (optional)**		**Artefacts Provided as outputs of the Activities**	
Name	Input Simulation Model	Name	FMI Model	Name	Output Simulation Model
Generic Type: (Tool or language independend type)	Simulation Model	Type:	FMI Standard Model	Generic Type: (Tool or language independend type)	Simulation Model
Required Properties: (Information required in interactions between steps)	- Simulation Model ID - Simulation Model version - Simulation Model description (e.g., "simulation of the flow of deicing fluid from reservoir into aileron") - List of properties representing the inputs required by the simulation (e.g., event "launch deicing fluid")	Properties:	TBD	Provided Properties: (Information provided in interactions between steps)	- List of properties representing the results of the simulation (e.g., "Time for fluid flow until complete deicing"). - Additional list of properties defined by the FMI Standard
Description & Interoperability Additional Constraints: A "Simulation Model" Type acts as a wrapper of any kinds of simulation models handled internally by simulation tools and expose a set of meta-properties to be shared between these tools		Description:		Description & Interoperability Additional Constraints:	

Additional information about this scenario can be found in the Annex of this deliverable.

3.7 Method “Change Impact Analysis”

The general purpose of this engineering method is to assess the impact of a change in a top level requirement to a given system definition baseline. It is assumed that several tools are involved, each tool providing a different type of data that is relevant to describe the system definition baseline (e.g. derived requirements, models with different levels of granularity and for different viewpoints, simulation results, test results, implemented code, bill of materials, and other types of product documentation).

To assess the impact of a change in a top level requirement, all data elements have to be identified and presented to the engineer that are related to the top level requirement.

<table>
<tr><th colspan="3">Engineering Method: UC208_ChangeImpactAnalysis_001</th></tr>
<tr><td colspan="3">Purpose: Requirements Engineer wants to assess the impact of a requirement change to the current technical solution of the de-icing system.</td></tr>
<tr><td colspan="3">Comments:</td></tr>
<tr><th>Pre-Condition</th><th>Engineering Activities
(made of steps)</th><th>Post-Condition</th></tr>
<tr><td>Engineers have defined a first technical solution for the de-icing system based on a given set of requirements. The technical solution is described in many different models managed by various tools and data-bases. A change request is launched by Top Management that would lead to a modification of a key requirement (e.g. reducing the max. allowed system weight).</td><td>1. Upon receival of a Change Request, Requirements Engineer selects the weight requirement that shall be modified, and launches a request to get list of related data objects
2. Request is forwarded to other tools
3. Tools are sending back list of related data
4. A “traceability” table or matrix is created to illustrate the related data
5. Requirements engineer requests a preview of a system architecture model that is related to the weight requirement
6. Request is forwarded to the respective modeling tool
7. Modeling tool is generating a preview of the architecture and sends it back.
Example:

</td><td>A Traceability matrix that illustrates the data impacted by a change of the weight requirement is created. Example:
</td></tr>
<tr><td>Notes:</td><td>Notes:</td><td>Notes:</td></tr>
<tr><th>Artefacts Required as inputs of the Activities</th><th>Artefacts used internally within the Activities</th><th>Artefacts Provided as outputs of the Activities</th></tr>
</table>

		(optional)			
Name	Requirements	Name		Name	Change Impact Results
Generic Type: (Tool or language independend type)	Requirements in natural language format	Type:		Generic Type: (Tool or language independend type)	Table, Document or Model
Required Properties: (Information required in interactions between steps)	Version, Baseline, Date of Creation, Approval Status, Author + Values for metrics that are constraining the de-icing system concepts and that are now being changed, such as maximum weight.	Properties:	Version, Baseline, Date of Creation, Approval Status, Author + Values for metrics that are constraining the de-icing system concepts and that are now being changed, such as maximum weight.	Provided Properties: (Information provided in interactions between steps)	Version, Baseline, Date of Creation, Approval Status, Author of all data and models that are impacted by the requirements change.
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Model	Name		Name	
Generic Type: (Tool or language independend type)	Logical Architecture Model with state-based behavior	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Version, Baseline, Date of Creation, Approval Status, Author, State-based Simulation results + some simple static parameters such as purchase cost	Properties:	Version, Baseline, Date of Creation, Approval Status, Author, State-based Simulation results + some simple static parameters such as purchase cost	Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Safety Model	Name		Name	
Generic Type: (Tool or language independend type)	Safety Model for Failure Rate Calculation and for dysfunctional Simulation	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Version, Baseline, Date of Creation, Approval Status, Author, Dysfunctional Simulation result + overall system failure rates (for loss and erroneous)	Properties:	Version, Baseline, Date of Creation, Approval Status, Author, Dysfunctional Simulation result + overall system failure rates (for loss and erroneous)	Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing Physical Behavior Model A based on Simulink	Name		Name	
Generic Type: (Tool or language independend type)	Physical Behavior Model (in Matlab/Simulink)	Type:	Model elements, especially diagrams	Generic Type: (Tool or language independend type)	

Required Properties: (Information required in interactions between steps)	Version, Baseline, Date of Creation, Approval Status, Author, Resulting values for physical behavior simulation - e.g. resulting ice accretion in mm.	Properties:	Version, Baseline, Date of Creation, Approval Status, Author, Resulting values for physical behavior simulation - e.g. resulting ice accretion in mm.	Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing Physical Behavior Model B based on Dymola	Name		Name	
Generic Type: (Tool or language independend type)	Physical Behavior Model (e.g. in Modelica)	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Version, Baseline, Date of Creation, Approval Status, Author, Resulting values for physical behavior simulation - e.g. resulting ice accretion in mm.	Properties:	Version, Baseline, Date of Creation, Approval Status, Author, Resulting values for physical behavior simulation - e.g. resulting ice accretion in mm.	Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Weight Model	Name		Name	
Generic Type: (Tool or language independend type)	Weight Model	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Version, Baseline, Date of Creation, Approval Status, Author, Overall System Weight	Properties:	Version, Baseline, Date of Creation, Approval Status, Author, Overall System Weight	Provided Properties: (Information provided in interactions between steps)	
Name	De-icing System Product Data	Name		Name	
Generic Type: (Tool or language independend type)	Files, Code, Documents, Models under configuration, etc.	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Version, Baseline, Date of Creation, Approval Status, Author.	Properties:	Version, Baseline, Date of Creation, Approval Status, Author.	Provided Properties: (Information provided in interactions between steps)	

Additional information about this scenario can be found in the Annex of this deliverable.

3.8 Method “Fault-tree Generation”

The general purpose of this engineering method is to generate fault-tree's out of a given set of engineering data. This fault-tree will be used to assess the failure probability for a given system concept.

This engineering method has not been reviewed in detail at the review meeting on November 12th. Modifications to this engineering method are likely and will be detailed in the next version of this deliverable.

<table>
<tr><th colspan="6">Engineering Method: UC208_GenerateFaultTrees_001</th></tr>
<tr><td colspan="6">Purpose: The safety designer would like to generate fault trees corresponding to a list of failure conditions.</td></tr>
<tr><td colspan="6">Comments:</td></tr>
<tr><th colspan="2">Pre-Condition</th><th colspan="2">Engineering Activities
(made of steps)</th><th colspan="2">Post-Condition</th></tr>
<tr><td colspan="2">The safety data is stored in a safety in-house tool

Dysfunctional models are available</td><td colspan="2">1. In FT+, search list of Failure Conditions by applying service “Get Failure Condition List”
2. Request is forwarded to In-House Safety Tool
3. In-House Safety tool sends failure condition list
4. In FT+, search list of failure components by applying service “Get Failure Components List”
5. Request is forwarded to a safety modeling and analysis tool based on AltaRica Language
6. For each Failure Condition, the list of components which is linked to the Failure condition is sent to FT+
7. In FT+, the fault-trees are generated</td><td colspan="2">Fault-trees are generated</td></tr>
<tr><td colspan="2">Notes:</td><td colspan="2">Notes:</td><td colspan="2">Notes:</td></tr>
<tr><th colspan="2">Artefacts Required as inputs of the Activities</th><th colspan="2">Artefacts used internally within the Activities
(optional)</th><th colspan="2">Artefacts Provided as outputs of the Activities</th></tr>
<tr><td>Name</td><td>Dysfunctional Models (with appropriate detailed descriptions)</td><td></td><td></td><td>Name</td><td>Fault-Trees Model (with appropriate detailed descriptions)</td></tr>
<tr><td>Generic Type: (Tool or language independend type)</td><td>Dysfunctional Models</td><td></td><td></td><td>Generic Type: (Tool or language independend type)</td><td>Fault-Tree Model</td></tr>
<tr><td>Required Properties: (Information required in interactions between steps)</td><td>TBD</td><td></td><td></td><td>Provided Properties: (Information provided in interactions between steps)</td><td>TBD</td></tr>
<tr><td colspan="2">Description & Interoperability Additional Constraints:</td><td colspan="2"></td><td colspan="2">Description & Interoperability Additional Constraints:</td></tr>
</table>

Name	Safety Data (with appropriate detailed descriptions)			Name	TBD
Generic Type: (Tool or language independend type)	Safety Data			Generic Type: (Tool or language independend type)	TBD
Required Properties: (Information required in interactions between steps)	TBD			Provided Properties: (Information provided in interactions between steps)	TBD
Description & Interoperability Additional Constraints:				Description & Interoperability Additional Constraints:	

Additional information about this scenario can be found in the Annex of this deliverable.

3.9 Method "Maintain Consistency between multi-viewpoint models"

The general purpose of this engineering method is to ensure that a models describing a given system concept are consistent with each other.

This engineering method has not been reviewed in detail at the review meeting on November 12th. Modifications to this engineering method are likely and will be detailed in the next version of this deliverable.

Engineering Method: UC208_MaintainConsistencyBetweenMultiViewpointModels_001		
Purpose: Engineers want to ensure that their models are consistent (for those data that is used in many different tools) after a change occurs.		
Comments:		
Pre-Condition	**Engineering Activities (made of steps)**	**Post-Condition**

Engineers have defined many models to describe a technical solution for the de-icing system. Each model represents a different viewpoints of the de-icing system: - For example, a SysML tool could be used to describe the baseline architecture for the deicing system (logical or technical view) - For example, the AltaRica tool could used to define a model that describes the safety view of the system - For example, Matlab/Simulink could be used to define a model that describes the pressure view - For example, Papyrus could be used to define a weight model Some of the Models that describe the de-icing system contain data that is used by other models as well (e.g. a valve that regulates a de-icing fluid is used in the Safety Model and in the Pressure Model)	1. In SysML tool, the engineer managing the baseline model of de-icing system is changing Valve A (e.g. using a different Valve from another supplier). He launches the service “send data update” 2. The new data for the modified Valve A is forwarded to all other tools that are using Valve A in their models 3. Engineers working on other tools get the notification that there models are not consistent any more with the baseline, since Valve A has been changed 4. Engineers are accepting the update of the data in their models Alternative 1: Data would be automatically updated, and engineers would just get a respective notification Alternative 2: A Data Object “Valve A” does not physically exist in the models of the other engineers, they just have links to the original “Valve A” object. In that case, their models are also automatically updated as soon as the original data in baseline model changes.	All models are consistent with each other
Notes:	Notes:	Notes:

Artefacts Required as inputs of the Activities		**Artefacts used internally within the Activities (optional)**		**Artefacts Provided as outputs of the Activities**	
Name	De-icing System Baseline Architecture Model + related Data Objects	Name		Name	
Generic Type: (Tool or language independend type)	Logical or Technical Architecture Model and related data objects (e.g. components, interfaces)	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Data object type, Data object ID, Version, Baseline, Date of Creation, Approval Status, Author	Properties:	Data object type, Data object ID, Version, Baseline, Date of Creation, Approval Status, Author	Provided Properties: (Information provided in interactions between steps)	Data object type, Data object ID, Version, Baseline, Date of Creation, Approval Status, Author
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Safety Model + related data objects	Name		Name	
Generic Type: (Tool or language independend type)	Safety Model and related data objects with safety properties	Type:		Generic Type: (Tool or language independend type)	

Required Properties: (Information required in interactions between steps)	Data object type, Data object ID, Failure Rate of Data object, Version, Baseline, Date of Creation, Approval Status, Author	Properties:	Data object type, Data object ID, Failure Rate of Data object, Version, Baseline, Date of Creation, Approval Status, Author	Provided Properties: (Information provided in interactions between steps)	Data object type, Data object ID, Failure Rate of Data object, Version, Baseline, Date of Creation, Approval Status, Author
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing Physical Behavior Model based on Simulink + related data objects	Name		Name	
Generic Type: (Tool or language independend type)	Physical Behavior Model and related data objects	Type:	Model elements, especially diagrams	Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Data object type, Data object ID, physical behavior property of Data object (e.g. max. allowed pressure that can pass through a valve), Version, Baseline, Date of Creation, Approval Status, Author	Properties:	Data object type, Data object ID, physical behavior property of Data object (e.g. max. allowed pressure that can pass through a valve), Version, Baseline, Date of Creation, Approval Status, Author	Provided Properties: (Information provided in interactions between steps)	Data object type, Data object ID, physical behavior property of Data object (e.g. max. allowed pressure that can pass through a valve), Version, Baseline, Date of Creation, Approval Status, Author
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Weight Model + related data objects	Name		Name	
Generic Type: (Tool or language independend type)	Weight Model and related and data objects	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Data object type, Data object ID, weight of Data object, Version, Baseline, Date of Creation, Approval Status, Author	Properties:	Data object type, Data object ID, weight of Data object, Version, Baseline, Date of Creation, Approval Status, Author	Provided Properties: (Information provided in interactions between steps)	Data object type, Data object ID, weight of Data object, Version, Baseline, Date of Creation, Approval Status, Author

3.10 Method “Search Data”

The general purpose of this engineering method is to provide information about the data that describes a given system definition (e.g. models, requirements, product documentation), such as version, author, date of creation.

This engineering method has not been reviewed in detail at the review meeting on November 12th. Modifications to this engineering method are likely and will be detailed in the next version of this deliverable.

Engineering Method: UC208_SearchData_001
Purpose: The System Architect of the De-icing system wants to visualize the history of a data (different versions of a data). He does not know in which tool the data is defined.
Comments:

Pre-Condition	Engineering Activities (made of steps)	Post-Condition
Data describing the de-icing system has been defined (e.g. Requirements, different types of models, simulation results, test results, safety calculation results). The data is stored in many different data-bases. It is assumed that each set of data has an owner and a version.	1. In a dedicated search engine tool, launch service “get owner data version” 2. Request is forwarded to the tools that are managing data 3. Information about the data managed by the tools is sent back to the search engine. Only the tool which owns the searched data sends the data.	In the search engine tools, information about the data is displayed (e.g. version, owner, type of data)
Notes:	Notes:	Notes:

Artefacts Required as inputs of the Activities		Artefacts used internally within the Activities (optional)		Artefacts Provided as outputs of the Activities	
Name	De-icing System Baseline Architecture Model + related Data Objects	Name		Name	
Generic Type: (Tool or language independend type)	Logical or Technical Architecture Model and related data objects (e.g. components, interfaces)	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Data object type, Data object ID, Version, Baseline, Date of Creation, Approval Status, Author	Properties:	Data object type, Data object ID, Version, Baseline, Date of Creation, Approval Status, Author	Provided Properties: (Information provided in interactions between steps)	Data object type, Data object ID, Version, Baseline, Date of Creation, Approval Status, Author
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Safety Model + related data objects	Name		Name	
Generic Type: (Tool or language independend type)	Safety Model and related data objects with safety properties	Type:		Generic Type: (Tool or language independend type)	

Required Properties: (Information required in interactions between steps)	Data object type, Data object ID, Failure Rate of Data object, Version, Baseline, Date of Creation, Approval Status, Author	Properties:	Data object type, Data object ID, Failure Rate of Data object, Version, Baseline, Date of Creation, Approval Status, Author	Provided Properties: (Information provided in interactions between steps)	Data object type, Data object ID, Version, Baseline, Date of Creation, Approval Status, Author
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing Physical Behavior Model based on Simulink + related data objects	Name		Name	
Generic Type: (Tool or language independend type)	Physical Behavior Model and related data objects	Type:	Model elements, especially diagrams	Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Data object type, Data object ID, physical behavior property of Data object (e.g. max. allowed pressure that can pass through a valve), Version, Baseline, Date of Creation, Approval Status, Author	Properties:	Data object type, Data object ID, physical behavior property of Data object (e.g. max. allowed pressure that can pass through a valve), Version, Baseline, Date of Creation, Approval Status, Author	Provided Properties: (Information provided in interactions between steps)	Data object type, Data object ID, Version, Baseline, Date of Creation, Approval Status, Author
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Weight Model + related data objects	Name		Name	
Generic Type: (Tool or language independend type)	Weight Model and related and data objects	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Data object type, Data object ID, weight of Data object, Version, Baseline, Date of Creation, Approval Status, Author	Properties:	Data object type, Data object ID, weight of Data object, Version, Baseline, Date of Creation, Approval Status, Author	Provided Properties: (Information provided in interactions between steps)	Data object type, Data object ID, Version, Baseline, Date of Creation, Approval Status, Author

3.11 Method “Provide Specification”

The general purpose of this engineering method is to identify and assemble all data needed for a system specification. It is assumed that the relevant data is managed by many different tools.

Engineering Method: UC208_ProvideSpecificationDocument_001

Purpose: Responsible Engineer at OEM level wants to send specification for a sub-system to a supplier					
Comments:					
Pre-Condition		**Engineering Activities (made of steps)**		**Post-Condition**	
Data for the sub-system specification is spread among different tools		1. In all tools where specification relevant data is stored, engineers identify and tag the relevant data 2. Engineer launches service "get data for specification" 3. Request is forwarded to all relevant tools 4. From tools, specification relevant data is send back 5. In the main tool of the engineer, all data is assembled to one specification and send to supplier 6. Supplier receives specification ALTERNATIVE 5b. Supplier requests specification 6.b Specification is send upon request to supplier		Sub-System specification has been assembled and sent to supplier	
Notes:		Notes:		Notes:	
Artefacts Required as inputs of the Activities		**Artefacts used internally within the Activities (optional)**		**Artefacts Provided as outputs of the Activities**	
Name	Requirements	Name		Name	Subsystem Specification
Generic Type: (Tool or language independend type)	Requirements in natural language format	Type:	Set of Requirements	Generic Type: (Tool or language independend type)	Document or Model
Required Properties: (Information required in interactions between steps)	Version, Baseline, Date of Creation, Approval Status, Author	Properties:		Provided Properties: (Information provided in interactions between steps)	Version, Baseline, Date of Creation, Approval Status, Author
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-icing System Model	Name		Name	
Generic Type: (Tool or language independend type)	Logical Architecture Model	Type:	Model elements, especially diagrams	Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Version, Baseline, Date of Creation, Approval Status, Author	Properties:		Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional		Description:		Description & Interoperability Additional	

Constraints:				Constraints:	
Name	De-icing System Safety Model	Name		Name	
Generic Type: (Tool or language independend type)	Safety Model	Type:	Model elements, especially diagrams	Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Version, Baseline, Date of Creation, Approval Status, Author	Properties:		Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	

Additional information about this scenario can be found in the Annex of this deliverable.

3.12 Method "Provide Process Management"

This engineering method is part of the so called Life Cycle identified methods which supports the stakeholders during the entire design and development period. It is considered highly relevant for the Industrial partners and, even if its current description is quite a preliminary one, we plan to study this topic in more details during the project timeframe.

The general purposes of this engineering method are:

- To provide the involved stakeholders, such as the project manager and Quality assurance manager, with "context-aware" support about the activities and the process part they have to apply within their daily activity.

  Task-level work items that are considered by this method evolve from an initial state (New) to a terminal state (Closed). The following diagram shows a typical workflow for task-level work items. Workflow shall be customizable through proper process formalization language and authoring tool.

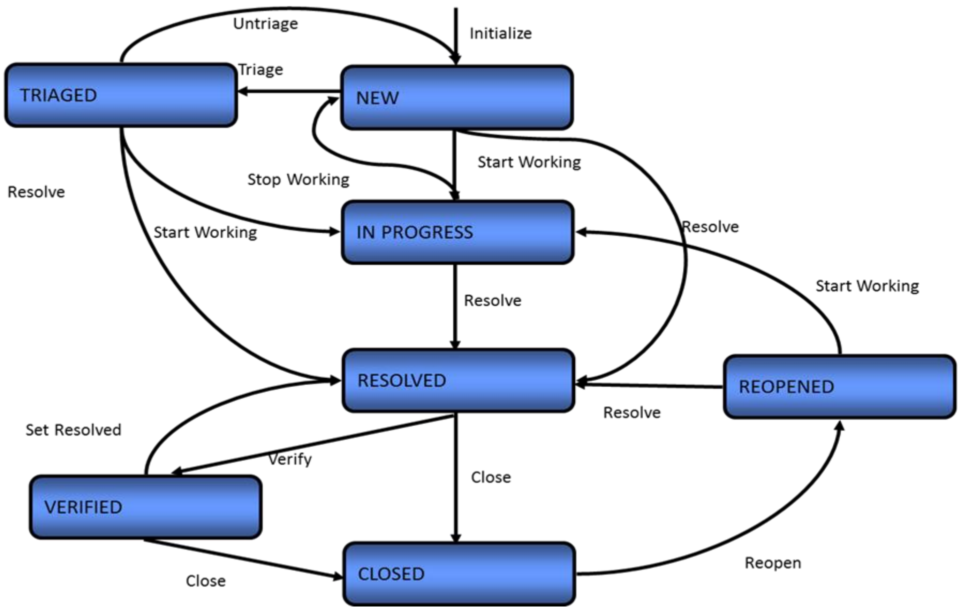


Figure 3-3: Typical Task status workflow

- To define a structured approach for establishing a “single point” and tool neutral access to the relevant information about the system under development, including the monitoring of development status and process implementation. This approach would cover both ALM and PLM domains. Tools from both domains should cooperate through an “hub” capable of managing both artifacts and links in a persistent way. This hub would rely on the IOS philosophy.

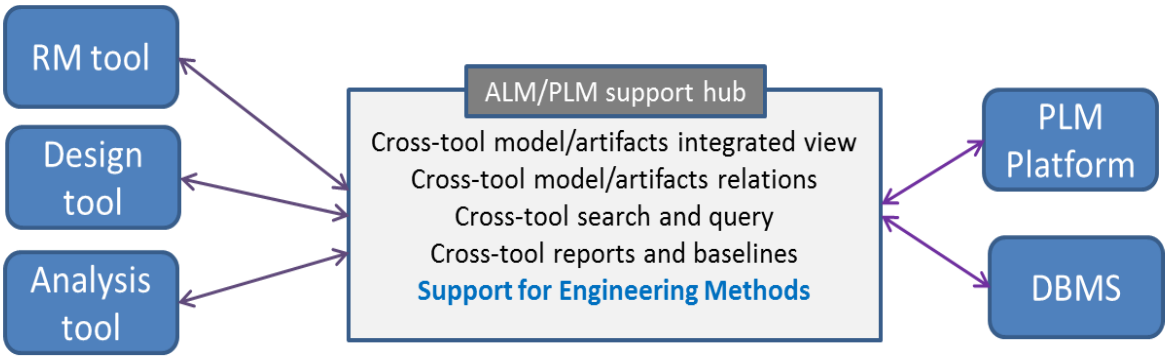


Figure 3-4: ALM and PLM support HUB approach

- With respect to the instance we are providing here, we can add that the stakeholders are supported in monitoring the system concept development status and the quality of identified solution through a requirement based criteria.
- The instance itself intend to show how a support can be provided to those stakeholders that are involved in the approval and base lining process for the so called “as conceived” solution. This architectural solution is the result of the trade-off analysis and preliminary design activities.
- The inspection and validation of the solution shall be based on a check for achievement of the planned goals as stated by the applicable system requirements belonging to the different categories: functional and non-functional.

Engineering Method: UC208_Provide Process Management_001		
Purpose: To provide the involved stakeholders, such as Project Manager and Quality Manager, with "context-aware" support about the actions and the process part they have to apply within their daily activity. To gain access to the information about the system under development, including the monitoring of development status and process implementation. The presented instance intend to show how support can be provided to stakeholders that are involved in the assessment of the identified system concept.		
Comments: Workflow Management services shall be configured through well recognized process specification standards. Currently envisaged standards are SPEM (specification) and possibly BPMN2 (enactment). This means that in some cases the need would occur for a model transformation from these formalisms to the ones supported by the services.		
Pre-Condition	**Engineering Activities (made of steps)**	**Post-Condition**
Applicable process specification and Customized workflow are available. This includes the involved stakeholders. Project Area created in the ALM platform repository. Artefacts relations (links) created <u>Instance related conditions:</u> Functional Specification Baseline available in PLM Requirements list is available in PLM Systems/Subsystems/Logical Equipment/ Functions are managed as Configuration Item in PLM Tool. System views are available in PLM: - establish the traces among requirements, design models, verification cases and analysis reports.	1. Configure the Workflow Manager platform through the desired "formal" process specification. (model transformation may be needed). 2. Invite stakeholders to join the project. 3. Inform the stakeholders about their current task to be performed. 4. On the work bench, provide relevant (context aware) task details to the stakeholders (on demand) 5. Display updated list of task related events 6. Display available process (progress) monitor information in the tracking task-board. Stakeholder asks for system information and links to development data through proper queries issued through the dedicated interface. • GetVerifiedRqList • GetScenario • GetAsConceivedConceptOverview 7. In PLM select the SYSTEM under analysis 8. Workflow management retrieves system information and traces from the support hub through the established links (OSLC linked data) 9. The involved stakeholders perform their inspection by navigating data displayed on Workflow management work bench 10. Workflow management evaluates progress information 11. Task relevant (updated) information are displayed to stakeholders through dedicated work item perspectives of the work bench.	12. All the needed stakeholders have been contacted and informed about their current duties 13. Each stakeholder got exhaustive information about the task to be performed. 14. Each stakeholder got context aware support about the task 15. Monitoring information about the on-going process are available to the stakeholders 16. Link to Integrated views about relevant system aspects is made available. 17. Task related events are notified according to stakeholder's need to know. 18. System Concept assessed
Notes:	Notes:	Notes:
Artefacts Required as inputs of the Activities	**Artefacts used internally within the Activities (optional)**	**Artefacts Provided as outputs of the Activities**
Name: Process	Name: Process monitoring data	Name: Updated progress data

	specification				
Generic Type: (Tool or language independend type)	SPEM (extended)	Type:	SPEM (extended)	Generic Type: (Tool or language independend type)	SPEM (extended)
Required Properties: (Information required in interactions between steps)	Library, Version, creation date, configuration ID, Ontology ID, name	Properties:	Project name, Project plan, metrics, milestones, Task ID	Provided Properties: (Information provided in interactions between steps)	Project name, task ID
Description & Interoperability Additional Constraints:		Description: these information may need adaptation before being consumed by the workflow manager		Description & Interoperability Additional Constraints: structured according to SPEM extension	
Name	Collaboration process (workflow)	Name	Task related Work Items	Name	System Information (updated)
Generic Type: (Tool or language independend type)	BPMN2 or equivalent	Type:	Configured through SPEM	Generic Type: (Tool or language independend type)	Requirements Function list Validation info
Required Properties: (Information required in interactions between steps)	Name, version, creation date	Properties:	Project name, System ID, Task ID	Provided Properties: (Information provided in interactions between steps)	Version, Baseline, Approval status, Validation Status, Verification Status
Description & Interoperability Additional Constraints:		Description: SPEM extension may be needed to represent these data		Description & Interoperability Additional Constraints:	
Name	Requirements list (instance specific)	Name	GetAsConceivedConcept Overview	Name	Messages to stakeholders
Generic Type: (Tool or language independend type)	Natural Language Requirements	Type:	OSLC extended service request	Generic Type: (Tool or language independend type)	Textual information
Required Properties: (Information required in interactions between steps)	Version, Baseline	Properties:	Project name, System ID	Provided Properties: (Information provided in interactions between steps)	Task ID, class, explanation
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	System views data	Name	Get Scenario	Name	Task log
Generic Type: (Tool or language independend type)	Rhapsody model Simulink model RMT analysis tables	Type:	OSLC extended service request	Generic Type: (Tool or language independend type)	textual
Required Properties: (Information	Item name, Version, Baseline, Planned Delivery	Properties:	Project name, System ID, Scenario ID	Provided Properties: (Information	Task ID, activity, explanation

required in interactions between steps)	Date, Issue Date			provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	Traces to models and V&V cases	Name	Integrated system view for display	Name	
Generic Type: (Tool or language independend type)	Links to resources	Type:	Aggregated information: Textual data + links	Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Version	Properties:	Requirements IDs Model elements Reports references Scenario description	Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	Traces to models and V&V cases	Name	GetVerifiedRqList	Name	
Generic Type: (Tool or language independend type)	Links to resources	Type:	OSLC extended service request	Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Version	Properties:	Project name, System ID	Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	Traces to models and V&V cases	Name	Task related events	Name	
Generic Type: (Tool or language independend type)	Links to resources	Type:	Displayed events (textual)	Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Version	Properties:	ID, Event explanation, severity, class	Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	

3.13 Method "Put all data under configuration control"

The general purpose of this engineering method is to put under Configuration Control the "functional view" (As Required, As Conceived) system solution once the concept is defined.
This engineering method aims to manage all the items related with the aeronautical product in the PLM tools, including the main output of the engineering conceptual and design phase, more specifically Requirements, Functions, System Elements (ASD S1000D compliance), Logical and Physical Architecture.
The System Architect / Configuration Manager will be able to import specific data and information (activities mapped to the functions, blocks mapped with the system) from the Rhapsody Model in order to be managed under configuration control and to support traceability all along the Product Life Cycle.
this activity shall be carried out in an integrated way with the following already available PLM view (as Designed, As Planned).

Engineering Method: UC208_Put all data under configuration control		
Purpose: CM wants to put under Configuration Control the "functional view" in order to manage and reuse these artifacts for similar product/capability classes.		
Comments:		
Pre-Condition	**Engineering Activities (made of steps)**	**Post-Condition**
Functions are managed as Activities of Activity Diagrams in MBSE Tool (SysML modeling). Systems/Sub-Systems/Logical Equipment are managed by Blocks in MBSE tool (SysML modeling). Systems/Subsystems/Logical Equipment/ Functions are managed as Configuration Item in PLM Tool. A Functional Specification defined in a SysML Model has been frozen as Baseline at the end of Functional Analysis (Black box activity diagram) applicable to a specific configuration A Functional Specification defined in a SysML Model has been frozen as Baseline at the end of Design Synthesis (White box activity diagrams) applicable to a specific configuration	AT SYSTEM LEVEL 1. In PLM, select the SYSTEM under analysis 2. In PLM, launch service "Get List of System Functionalities" 3. Request is forwarded to MBSE Tool (SysML modeling) 4. List of all functions is assembled and send back to PLM tool 5. In PLM, receive functions 6. In PLM, the developer associates information related to applicability to the imported Functions 7. In PLM, correlate System View CI to As-Designed View CI AT SUBSYSTEM LEVEL 1. In PLM, select the SYSTEM under analysis 2. In PLM, launch service "Get List of All Syb-system Functionalities" 3. Request is forwarded to MBSE (SysML modeling) 4. For each SUBSYSTEM the List of allocated functions is assembled and send back to PLM tool 5. In PLM, receive SUBSYSTEM functions 6. In PLM, the developer associates information related to applicability to the imported Functions	Systems View Management in PLM tool with Applicability management of Functionalities defined in MBSE tool (SysML modelling). Management of Commonalities and Comparison of different Functional Configuration in PLM. Management of traceability from System View CI to As-Designed View CI (e.g. Function to Part Number) in the PLM tool.
Notes:	Notes:	Notes:
Artefacts Required as inputs of the	**Artefacts used internally within the**	**Artefacts Provided as outputs of the**

Activities		Activities (optional)		Activities	
Name	De-Icing System Model	Name		Name	
Generic Type: (Tool or language independend type)	Functional behavior model and Logical Architecture (Rhapsody - SysML)	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Version, Baseline, Activity Diagram (System / Subsystem Primitive Operations and Event), Internal Block Diagram, Block Definition Diagram	Properties:	Version, Baseline, Activity Diagram (System / Subsystem Primitive Operations and Event), Internal Block Diagram, Block Defition Diagram	Provided Properties: (Information provided in interactions between steps)	System View (System Functionalities list) System View links to other views (i.e “as designed view”)
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	
Name	De-Icing System Product Data	Name		Name	
Generic Type: (Tool or language independend type)	Files, Codes, Document, Models under configuration, Functional Specification Baseline	Type:		Generic Type: (Tool or language independend type)	
Required Properties: (Information required in interactions between steps)	Version, Baseline, Date of Creation, Approval Status, Author	Properties:		Provided Properties: (Information provided in interactions between steps)	
Description & Interoperability Additional Constraints:		Description:		Description & Interoperability Additional Constraints:	

4 System Engineering Environment for Public Aerospace Use Case

4.1 Envisaged System Engineering Environment (SEE)

Figure 4-1 illustrates the System Engineering Environment for the Public Aerospace Use Case as it is currently envisaged. Please note that the list of tools and types of tools is not yet complete and may be updated in the future.

Figure 4-1 refers to a concept where the data used for the specification, design and analysis of aerospace systems such as the de-icing system is spread amongst many different databases and tools.

Taking into account the perimeter of the public aerospace use case, the envisaged System Engineering Environment will have to include tools and databases at least for Requirements Management, Functional Models, Physical Behaviour Models, Safety Models, Digital Mock-ups, Product Life Cycle Management, and Application Lifecycle Management. These tool and databases can be deployed at many different company sites or even across different companies.

In order to realise interoperability, each tool and database has to provide a connector that is based on open standards. The connector approach as well as the open standard for interoperability will be defined in WP6.1. The communication between the tools (e.g. sending of requests to other tools, receiving data from tools) can be realised by any kind of Network that is using web-protocols (e.g. LAN, Intranet, Internet).

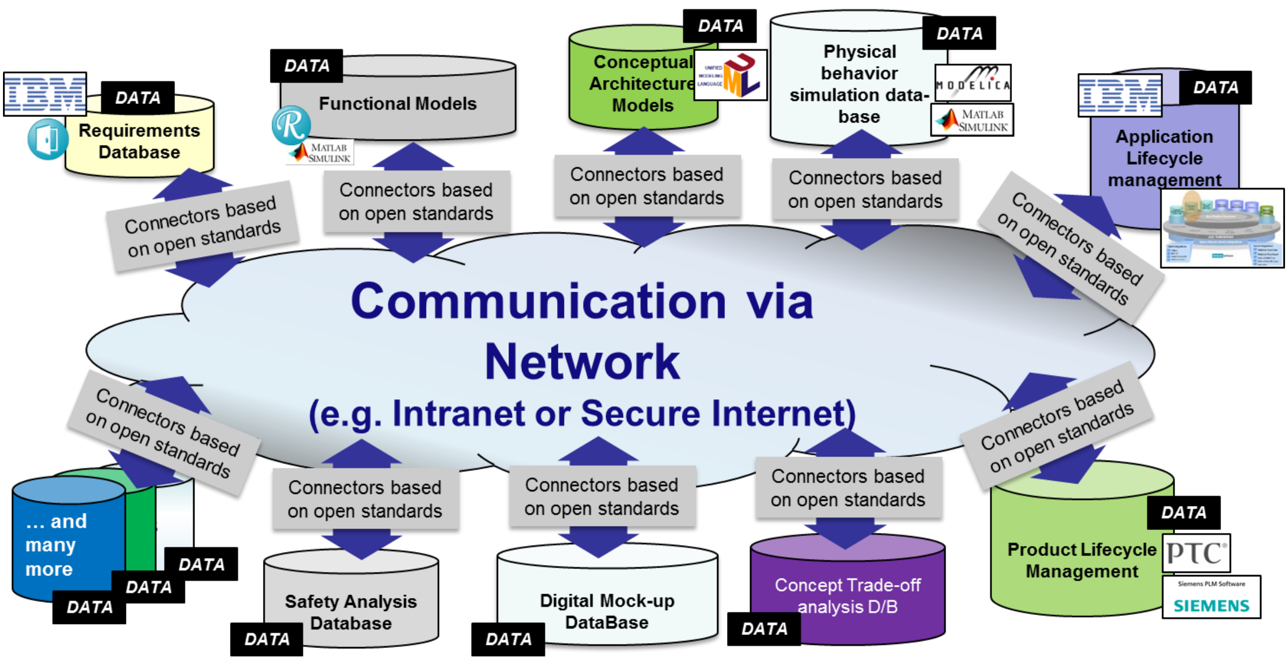


Figure 4-1: Initial version of Envisaged System Engineering Environment (SEE) for Public Aerospace Use Case

4.2 Demonstrator Set-up – December 2013

The Demonstrator Set-up as of December 2013 is illustrated in Figure 4-3. The current Demonstrator has been set-up mainly with support from IBM through WP6.11. The Demonstrator is currently based on IBM System and Software Engineering (SSE 4.03) and IBM Rational Engineering Lifecycle Management (RELM)

4.0.3 solutions. The decision to use IBM SSE and RELM as initial basis for the Public Aerospace Demonstrator has been taken since these software products are already using OSLC connectors.

The IBM SSE and RELM solutions have been installed on a Virtual Machine that is running on a Workstation at EADS Innovation Works in Hamburg. The characteristics of Workstation and Virtual Machine are illustrated in Figure 4-2. The overall performance of the demonstrator with these hardware characteristics is good for a single user access (e.g. time to access and edit data is normally less than 1 second).

	Workstation	Virtual Machine (for IBM SSE / RELM)
CPU	32 x Intel Xeon CPU E5-2687W @ 3.10 Ghz (2 sockets)	8 Cores
RAM	128 GB	32 GB
HD Space	2 x 2 TB (Raid 1)	400 GB HDD
OS	Proxmox (Linux based Hypervisor)	Windows 7 Professional 64 Bit

Figure 4-2: Characteristics of Workstation and VM for Demonstrator

The IBM SSE solution includes applications for Requirements Management, Design Management, Change and Configuration Management, Quality Management, and Configuration Management. Together with RELM, these applications are running as Web-front ends on a Tomcat Web Server. All applications are connected to dedicated Databases. The Jazz Team Server Database is responsible for Server and Application Management and also includes Requirements as well as references from Requirements to other data. The Design Manager Database is responsible for model store and contains currently functional models that have been defined using Rhapsody, as well as references from functional models to other data.

The CCM Database contains Change Requests, and the RELM Database contains Views on data and on data references.

As of December 2013 all databases have been deployed using Derby. It is foreseen to change the deployment of the databases to DB2 in order to improve the performance of the demonstrator especially with regard to multi-user access. It is also foreseen to connect additional tools to the demonstrator environment, such as Open Modelica (for traceability to physical behaviour simulation models). These additional tools could be deployed on dedicated virtual machines or other workstations. The development of a connector for Open Modelica has been initiated in cooperation with WP6.6. A deployment is foreseen for February 2014.

So far, access to the Demonstrator environment is limited to EADS IW G clients using remote desktop connection via the EADS Hamburg Intranet. The client thereby accesses remotely to a web-browser on the Virtual Machine, which then connects to the Tomcat Web-server. A connection via Internet using a secured VPN is foreseen but not yet working. Therefore, at the moment the only way to present the demonstrator outside of EADS IW Hamburg is through a WebEx connection via a local EADS Hamburg Client-PC sharing its screen.

In the future it is foreseen to enable a direct connection of external company networks to the Web-server on which the Demonstrator applications have been installed. This would allow us to simulate an Extended Enterprise System Engineering Environment, and also improve the sharing of data amongst WP208 participants.

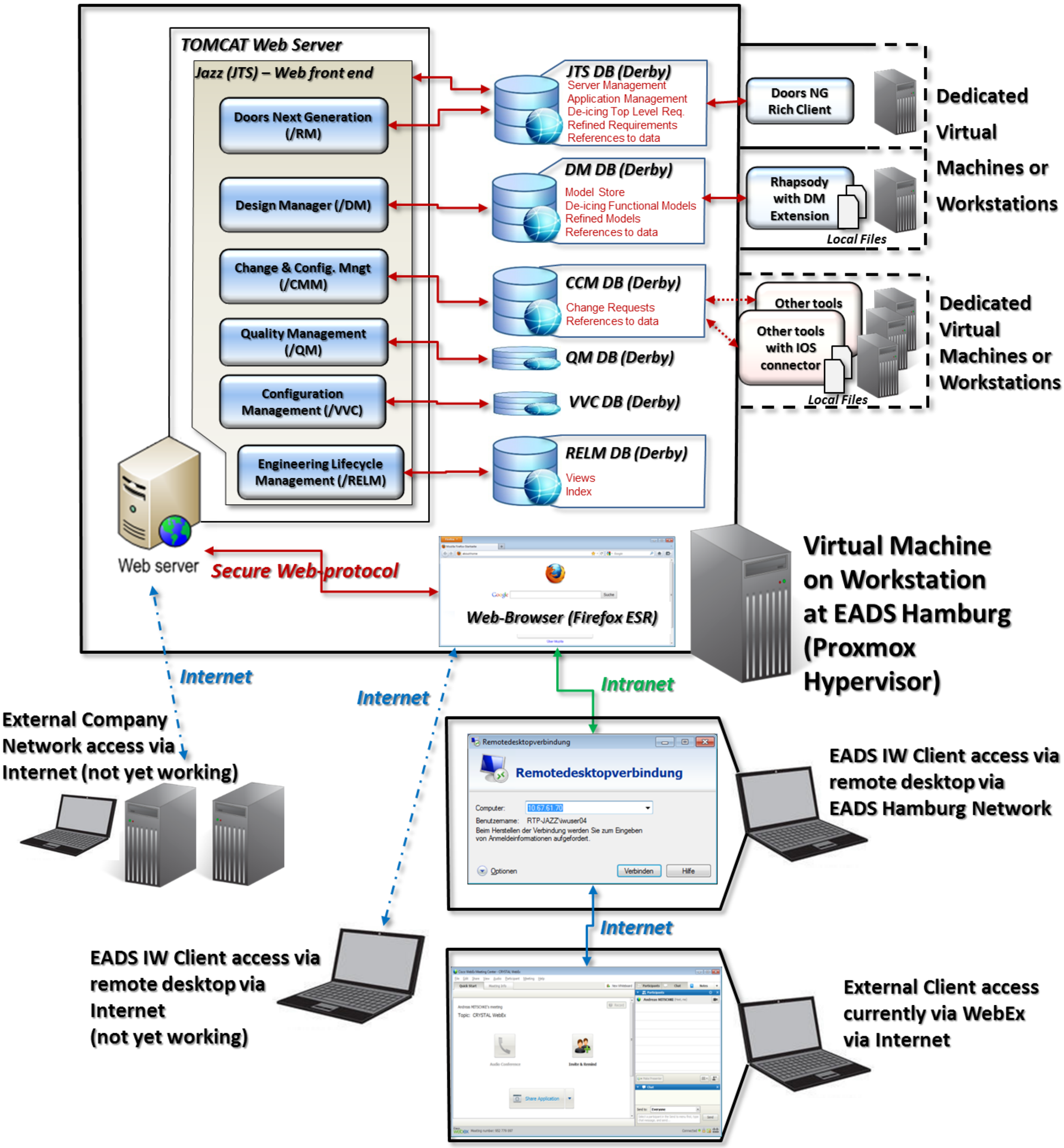


Figure 4-3: Demonstrator set-up as of December 2013

5 Overview De-icing System data and implemented Engineering Methods

This chapter provides an overview of the data that has been defined by WP208 Partners for specifying and describing the De-Icing System. The chapter also provides an overview of an Engineering Method that has been already implemented using the Demonstrator Set-up described in Chapter 4.

5.1 Data defined for the De-Icing System

In Chapter 2 we have explained the key process steps to be applied to the Public Aerospace Use Case (see Figure 2-3). According to these process steps, for the De-icing System we have defined the following set-of data:

- Initial top level requirements for the De-Icing System
- Top level functional analysis model for the De-Icing System
- De-Icing System Context View (System under Developement [SuD] vs. External Systems)
- Overview of alternative concepts for SuD
- Refined requirements and models for each alternative concept, including refined functional analysis views and product breakdown structures
- Dedicated analysis models to assess the alternative concepts

5.1.1 De-icing System Requirements

One of the first tasks for the project management of a new real Aircraft program relates to the definition of a Requirements Cascading Process and the implementation of this process into a Requirements Management tool. Figure 5-1 illustrates this step in a simplified way for the De-Icing System [ref Prel_Sizing_rev_2013].

As shown in Figure 5-1, we have defined a folder structure that distinguishes between Top Level Requirements, Sub-System Requirements, and Environmental Scenarios. The Top Level Requirements folder contains one Requirements Module that combines all Top Level Requirements for the De-Icing System. The Sub-System Requirements folder contains further Sub-folders for the Cockpit Display System and for alternative De-icing concepts. Those Sub-folders contain further modules that are combining a set of related requirements.

In a real Aircraft program several hundred requirements modules might be defined, organised in numerous hierarchical folders. In addition, a requirements traceability concept is defined, which e.g. states that individual requirements on an upper hierarchical level have to be allocated to related requirements modules on a lower hierachical level using a “satisfied by” relationship. In the aerospace industry this approach is typically called “Requirements Cascading”.

Today, the implementation of the traceability concept is often limited to the Requirements Management tool. For the public aerospace use case, we have defined and implemented a traceability concept that allows to set and to navigate on links beyond Requirements Management. This concept is described in Chapter 5.2.

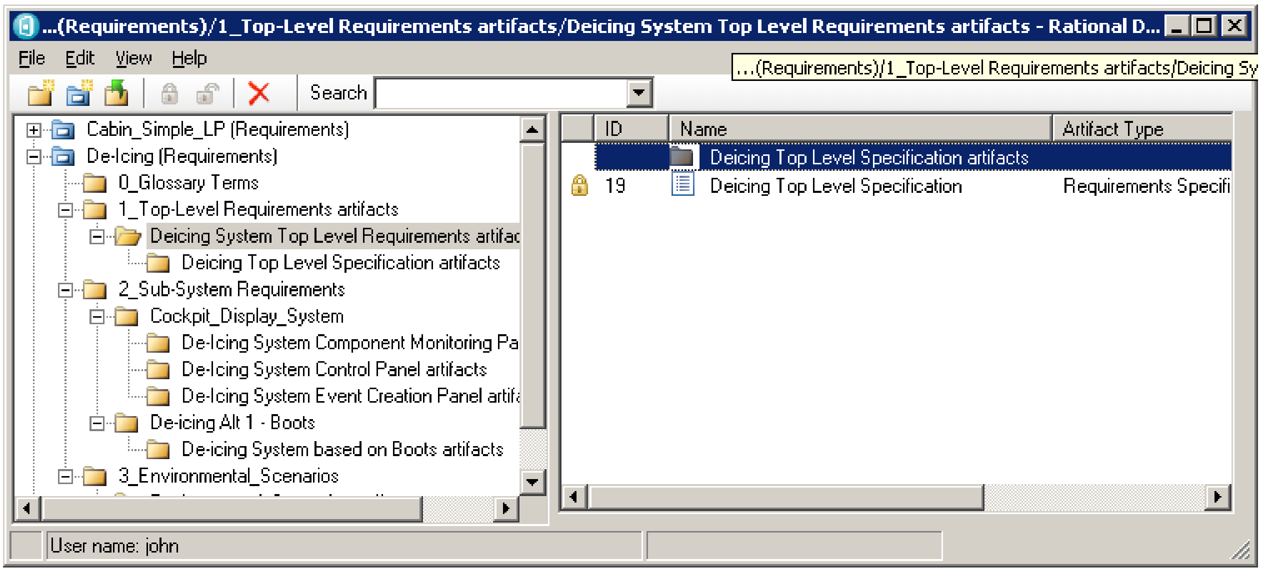


Figure 5-1: Organisation of De-Icing System Requirements in IBM Doors NG

The very initial list of top level requirements have been defined using DOORS version 9.1, later on we have transferred those Requirements into Doors Next Generation (NG), since Doors NG already provides an OSLC connector (see also Figure 4-3 for the demonstrator set-up).

A number of requirement types have been defined: Operational, Functional, Safety, Cost, Maintenance, Installation as well as Performance. Each of these requirements, in regardless with categories, are specified with an ID, type, allocation and attributes which will be important when it comes to functional analysis.

ID REQ	De-icing requirements at AC level specification	Requirement Type	Allocation	Requirement?
	rely on such service or intention.			
	The word MAY in the text expresses a permissible practice or action. It does not express a requirement of the Specification.	N/A	N/A	False
	1.3 Operational Requirements	N/A	N/A	False
SRS_30_00_01	The Ice Protection Sub-system shall permit aircraft operation without restriction in icing condition through "continuous maximum" and "intermittent maximum" as specified by FAR/JAR 25.	Operational	General	True
SRS_30_00_02	The Ice Protection Sub-system shall permit aircraft operation without restriction in icing condition within One Engine Inoperative condition.	Operational	General	True
SRS_30_00_03	The ice protection system shall be operated under all flight phase from take off to landing when ice protection is required and taking into account the following temperature envelope to which the AC is exposed:	Environmental Operational	General	True
SRS_30_01_09	Fast de-icing cycles shall be provided by the stand-by control system.	Operational	Control and indicators	True
	1.4 Safety Requirements	N/A	N/A	False
SRS_30_00_05	Probability of single catastrophic failure of the system shall be less than 1*E-09 per Flight Hour	Safety	General	True
SRS_30_00_06	Each ice protection components shall be redundant in order to garantee ice protection functionality after single failure event.	Safety	General	True
	The ice-protection failure which is categorized as catatrophic is the undetected malfunction of ice-detection.	N/A	N/A	False
	1.5 Physical Requirements	N/A	N/A	False
SRS_30_00_07	The weight of the ice-protection system shall not exceed 150 kg.	Non Functional	General	True
	1.6 Functional Requirements	N/A	N/A	False
SRS_30_01_10	The De-Icing Control Unit shall control the de-icing cycles and include control logics to automatically select the type of cycle (high- or low- speed cycle) depending on Static Air Temperature.	Functional	Control and indicators	True
SRS_30_01_11	The ice protection system shall provide the monitoring of its health status.	Functional	Control and indicators	True
SRS_30_01_12	The ice protection system shall provide the monitoring of its operating status.	Functional	Control and indicators	True
SRS_30_10_01	In non-icing conditions, the equipment shall enable the aerodynamic profile.	Functional	Airfoil	True

Figure 5-2: Initial Requirements managed in Doors 9.1

Each requirement has been defined with attributes which can be populated with other tools, for example the compliance status.

Name	Description	Type	Default value	Inherit value	Exists for	Multi valued
Absolute Number	System Attribute	Integer		No	Object	No
Compliance status	Report the compliance l...	Compliance type		No	Object	No
Component allocation	Report the component t...	Component alloc...		No	Object	No
Created By	System Attribute	String		No	Module & Object	No
Created On	System Attribute	Date		No	Module & Object	No
Created Thru	System Attribute	Created Thru	Manual Input	No	Object	No
Description	System Attribute	String		No	Module	No
ID REQ	Requirement number ba...	REQ NUMBER -...		No	Object	No
Last Modified By	System Attribute	String		No	Module & Object	No
Last Modified On	System Attribute	Date		No	Module & Object	No
MOC	Mean of Compliance	MOC type		No	Object	Yes
Name	System Attribute	String		No	Module	No
Notes	Report notes and comm...	Text		No	Object	No
Object Heading	System Attribute	String		No	Object	No
Object Short Text	System Attribute	String		No	Object	No
Object Text	System Attribute	Text		No	Object	No
Prefix	System Attribute	String		No	Module	No
Requirement Type	Define the type of the re...	Req_Type		No	Object	Yes
Requirement?	Attribute for filtering - req...	Boolean		No	Object	No
Risk Type	Cost / Technology / Sc...	Risk_Type		No	Object	Yes
Risk Value	High / Medium / Low	Risk_Value_Type		No	Object	No
Status	Draft / Under Review / ...	StatusType		No	Object	No
Validation method	Reference to the validat...	Text		No	Object	No

Figure 5-3: Type of requirements attributes list

Figure 5-4 illustrates the resulting Top Level Requirements Specification Document for the De-Icing System.

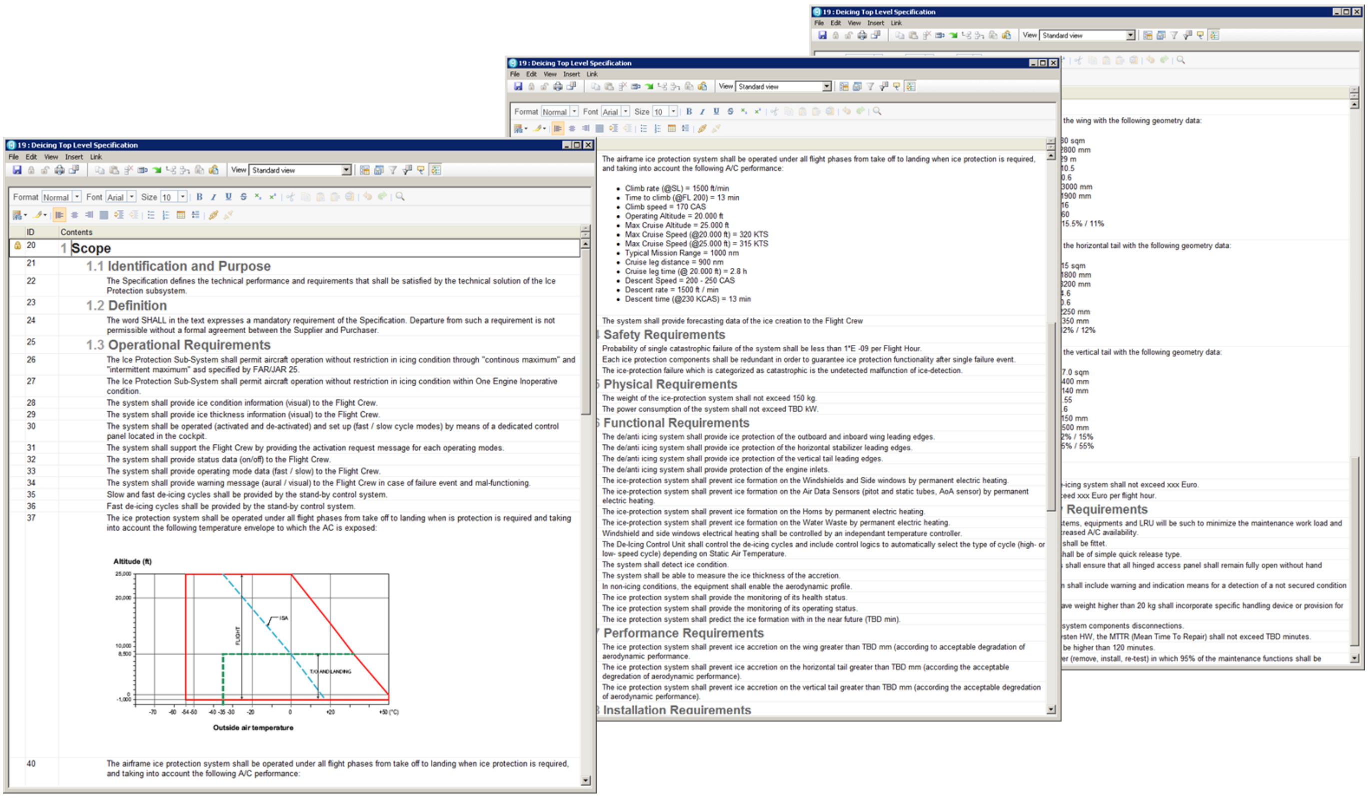


Figure 5-4: Top-Level De-icing System Requirements in IBM Doors NG

Hereafter are some examples for key Top Level Requirements:

- The system shall provide ice condition information (visual) to the Flight Crew (Functional)
- The de/anti icing system shall provide ice protection of the engine inlets (Functional)
- Probability of single catastrophic failure of the system shall be less than 1*E-09 Flight Hour (Safety)
- The cost for operating the system shall not exceed a fixed amount of Euro per flight hour (Cost)

- The Time To Repair shall not be higher than 120 minutes (Maintainability)

A complete list of all Top Level Requirements for the De-Icing System can be found in Annex (see Chapter 8.11)

In addition to the Top Level Requirements for the De-Icing System, a first set of requirements for the interaction of the Aircraft Crew with the De-Icing System have been defined. Figure 5-5 provides an illustration of Requirements for the De-Icing Control Panel and for the De-Icing System Component Monitoring Panel. Both Requirements provide mainly a first graphical representation of the expected Panels. The Requirements have satisfy relationships with some requirements in the Top Level Requirements Module.

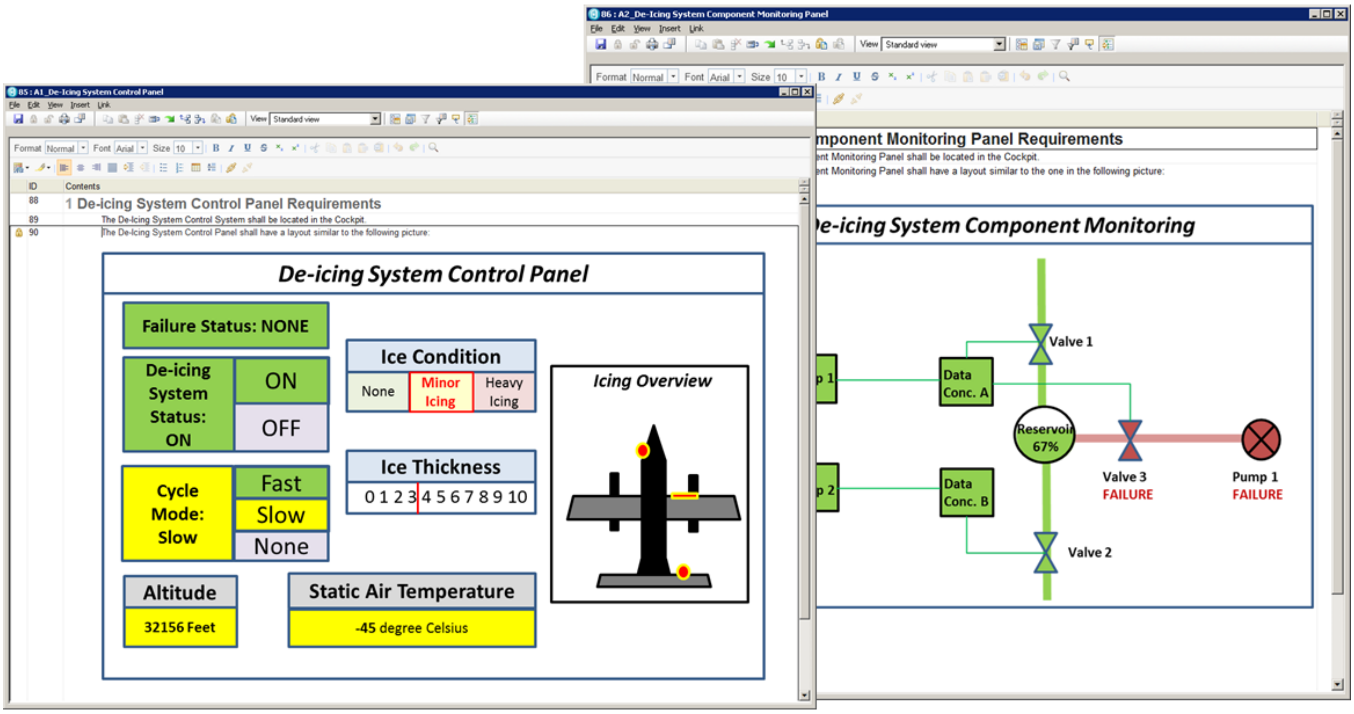


Figure 5-5: Requirements for De-Icing System Panels

5.1.2 Top Level functional analysis and system context views

The Rhapsody modelling process of the de-icing system follows the Harmony principles, although further refinement on the methodology will be introduced according to the Use Case needs. The process that has been used in the following modelling phase can be broken up in three different Model Based System Engineering steps: Requirements Analysis, System Functional Analysis and Design Synthesis.

The requirements analysis is a phase through which the stakeholder specifications are categorized and represented with a model (system use case diagram). The system Functional analysis is more focused on the translation of the functional requirements into a coherent description of system functions or operations, whereas the design synthesis is devoted in the architectural design of the system. More iteration are necessary, deepening the definition of the system. The entire process that has been followed can be reviewed in the following figure:

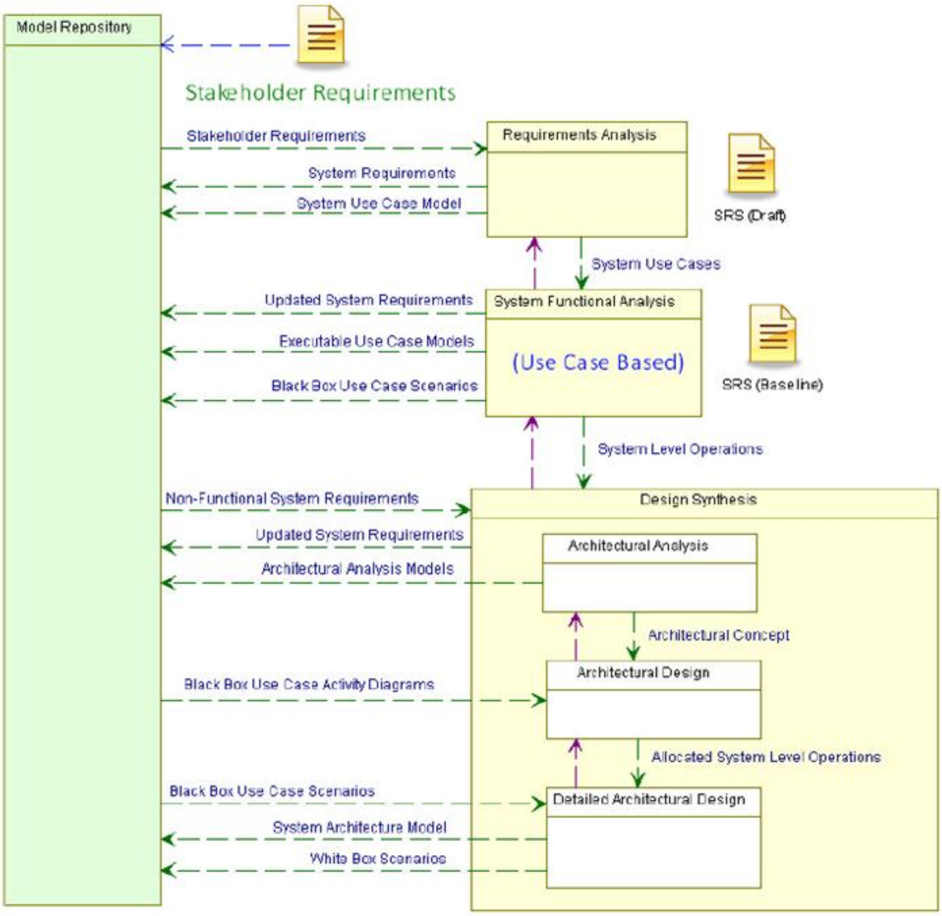


Figure 5-6: Harmony process workflow

The top level functional model is represented by the allocation of functional and operational requirements over the use cases defined for the ice protection system, regardless of system solutions (Boots, Electrothermal de-icing, etc…). In order to go through this process, the linking of top level requirements from DOORS to Rhapsody is required, as well as the definition of top level use cases and actors. Each use case and actor needs to be connected, considering the functional relationship between the two entities.

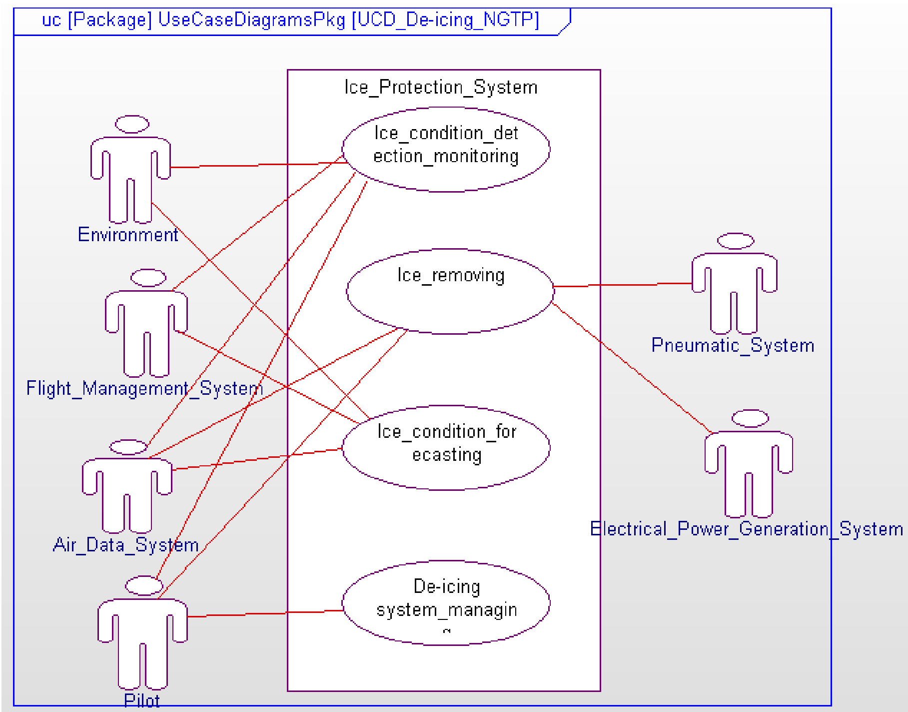


Figure 5-7: Use case Diagram with Use Cases and Actors connected

The Use Case Diagram is fundamental for the next phase, the functional one, where every top-level actions and actors for a generic ice protection system are represented through the Activity Diagram. In those type of diagrams, the set of actions and sub activities are descripted by a workflow. Both “sunny” and “rainy days” are represented in Figure 5-8, showing the different behaviour of the system under certain conditions.

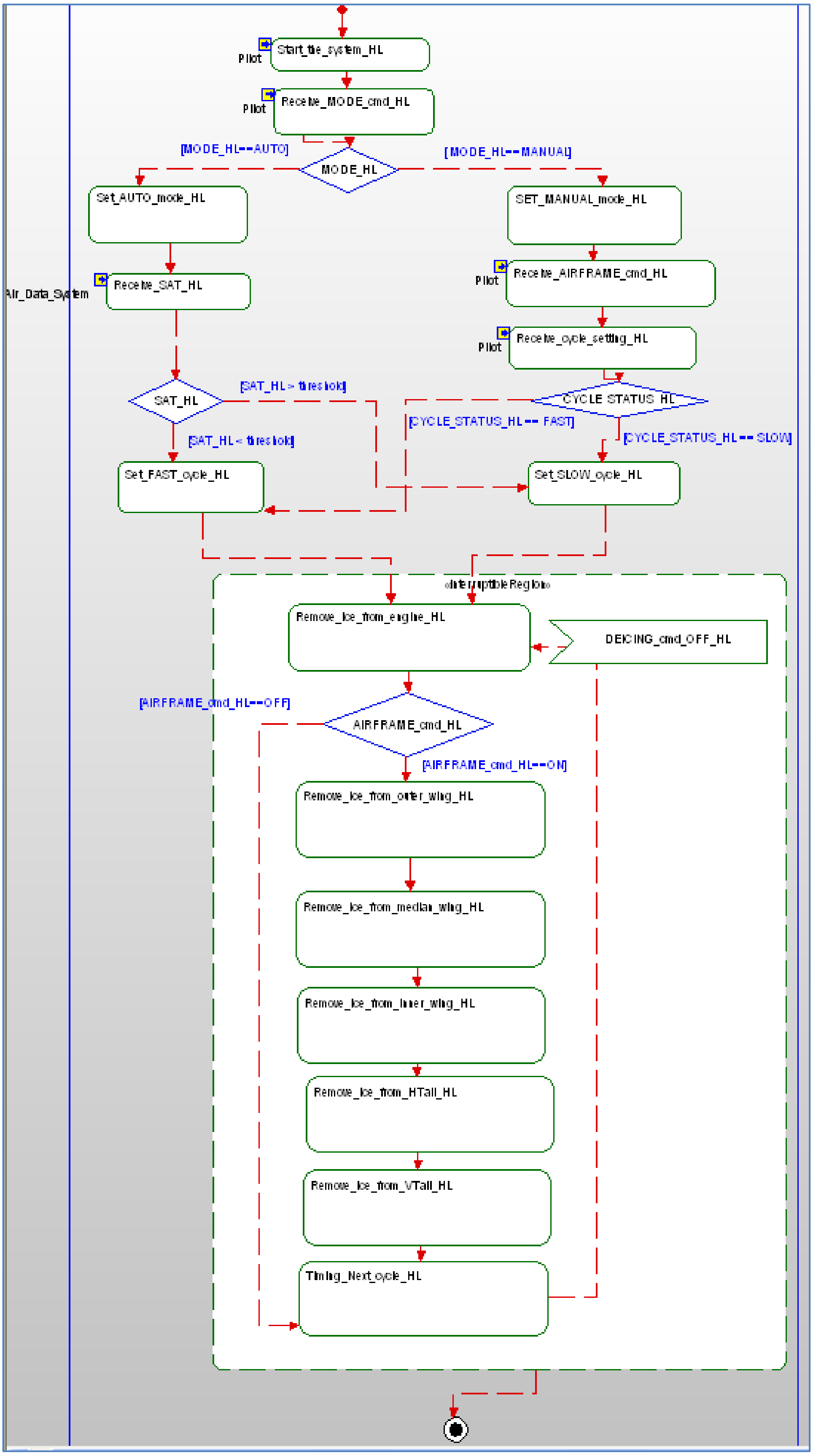


Figure 5-8:Top Level Black Box Activity Diagram for a generic Ice Protection System

The sequence Diagrams represent how actors and blocks of actions collaborate in some behaviour. One activity diagram can generate multiple sequence diagrams and thus different scenarios, depending on the complexity of the system and the number of decision nodes in the activity diagram.

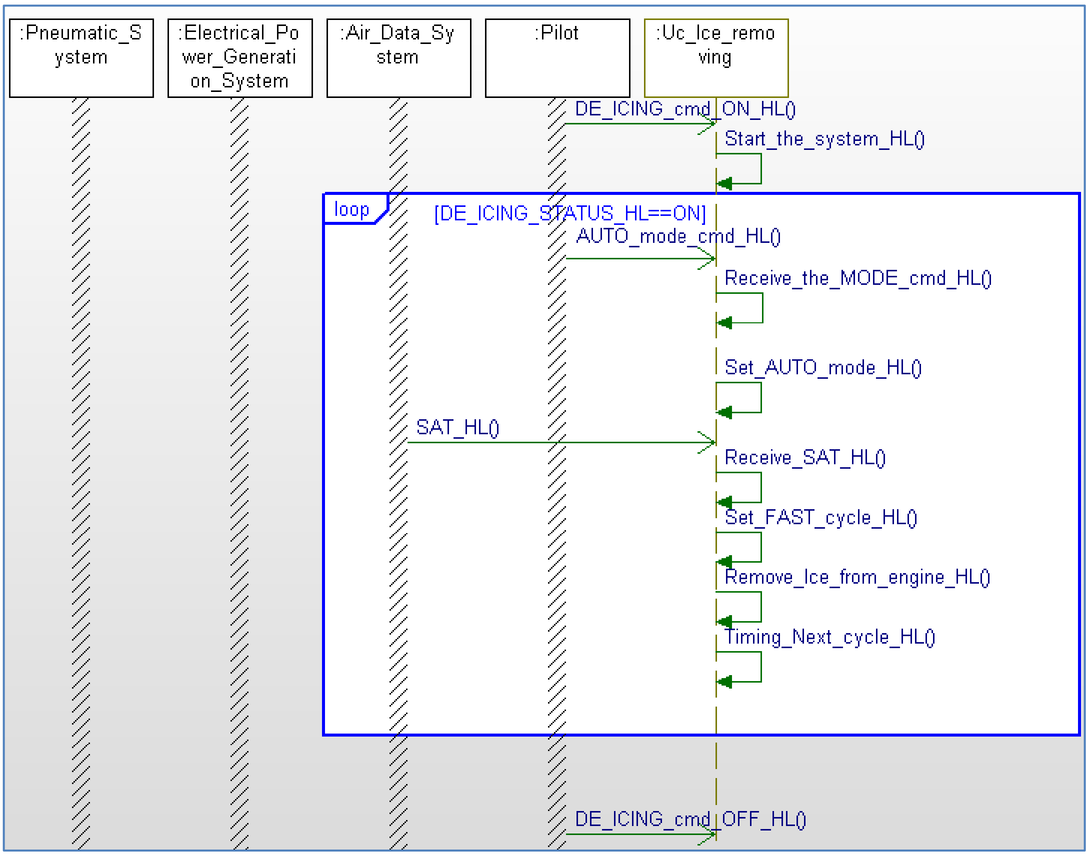


Figure 5-9: Sequence Diagram in Auto Mode, SAT sensed in order to set the FAST de-icing cycle, and Airframe command off.

5.1.3 Derived functional model per concept

Once the very top level view has been elicited using the Functional analysis phase, each possible concept should be identified, and the optimum one modelled using, for example, the MBSE Harmony Methodology, as it has been done so far.

In fact, System functional analysis defines what the system should do but not how it is to be done, this means that each system function should be then allocated into a defined system architecture. In order to define the optimum architecture, a parametric Trade Off study is necessary to identify the best means of achieving the capability of a particular function in a rational manner.

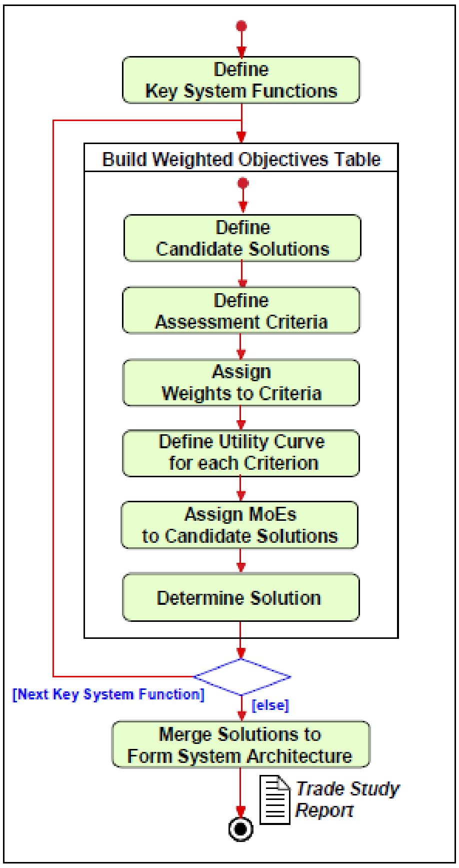


Figure 5-10: Workflow for the Trade off study

Feasible system solutions are then envisioned and fitted with a number of Assessment Criteria. Typically, these criteria are based upon customer constraints, required performance characteristics (i.e. safety, power consumption, etc…), and/or costs. Those performances can be derived from other tools and other analysis.

Five different configurations have been selected as candidates for the trade off analysis, employing four different Ice Protection System types, each of these have been preliminary sized at a subsystem level, finding out the dimensions, power consumptions as well as weights of the main components. Windshields, propellers, pitot tubes and other probes are anti/de-iced with the Electrothermal system which will not be a part of the trade off study.

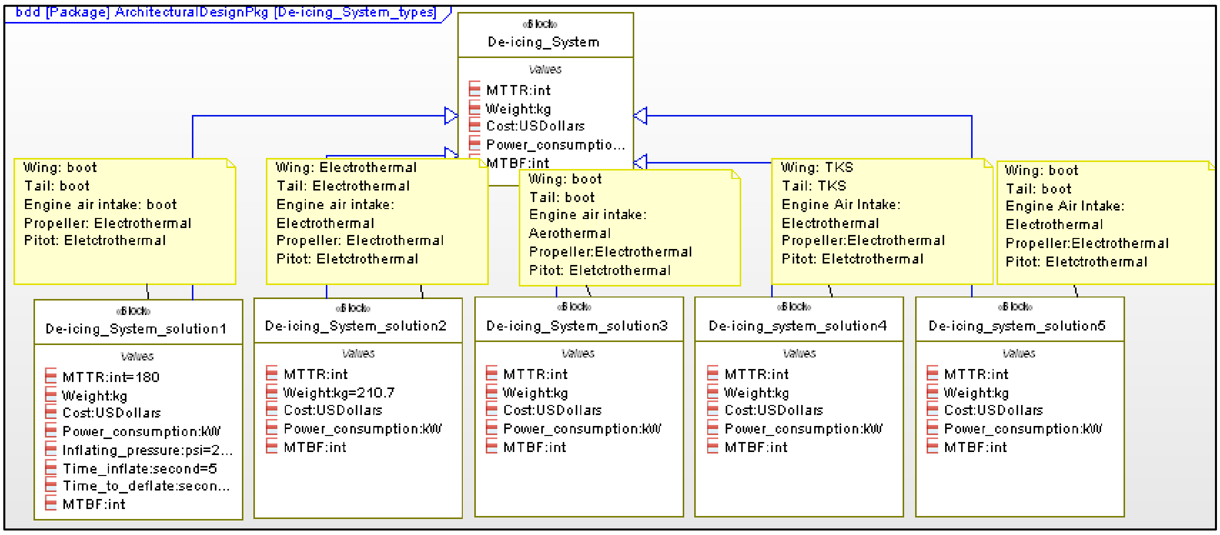


Figure 5-11: Solution taken into account for the trade-off study, attributes can be seen into the blocks at the bottom.

To compare the different solutions, each solutions is characterized by a set of normalized, dimensionless values, Measure of Effectiveness (MoE), which describe how effective a solution candidate is for a particular assessment criterion. The preferred solution is then determined by means of Weighted Objectives calculation.

In our case, we have selected the following metrics as shown in Figure 5 8:

- MTTR = Mean Time to Repair – this metrics is related to the reliability of the system
- Weight
- Cost – this value refers to both installation/production and to operating costs
- Power Consumption
- MTBF = Mean Time between Failure – this metrics is related to safety

Noteworthy, the entire Trade off study can be carried out in a tool different from Rhapsody.

Having identified the possible solutions, for example the Boots system for de-icing using on-board pneumatic power, it is possible to proceed with the architectural design phase for this solution once the functional design for the chosen candidate has been completed.

5.1.3.1 De-icing Boots system Functional Analysis Models

For the De-icing solution concept using boots, a set of refined requirements has been defined that is only applicable to this solution. Figure 5-12 illustrates the organisation of these refined requirements using a dedicated Requirements Module in Doors Next Generation. For example, a refined requirement only applicable for solution concept using boots is Requirements 106:” The time to inflate boots shall be less than 6 seconds”.

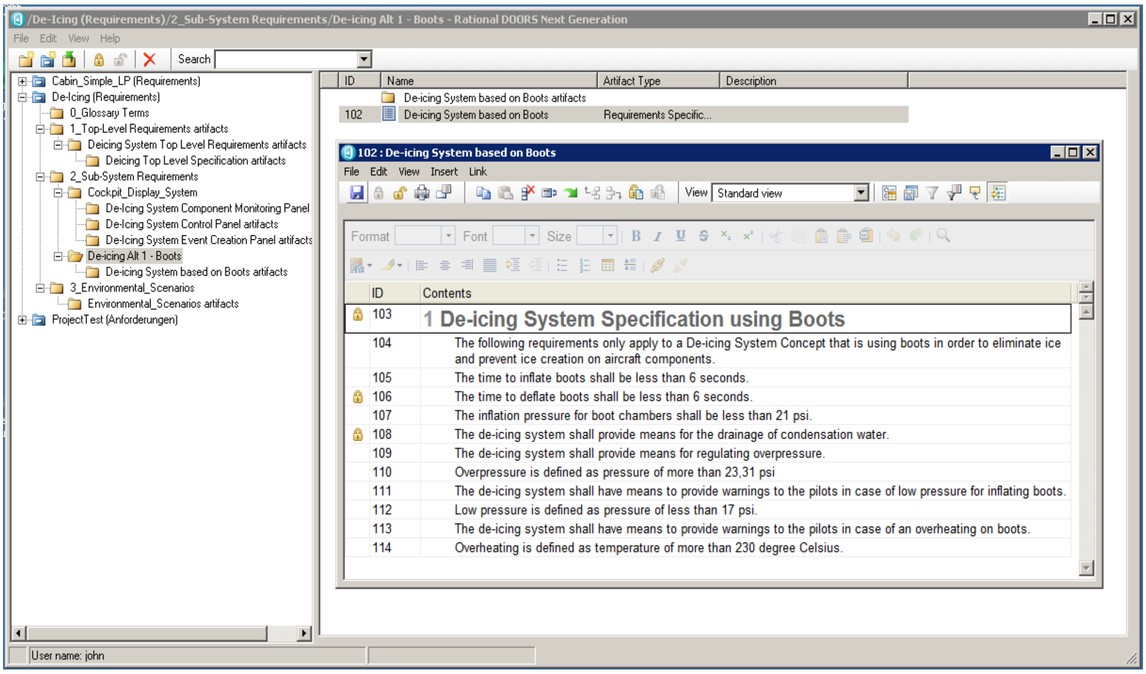


Figure 5-12: Refined Requirements for a De-Icing System using Boots concept

The boots system (Goodrich system) uses pneumatic power in order to inflate and deflate a number of geometrically-defined rubber bladders along the aircraft leading edges in order to crack the ice previously formed. This solution has been qualitatively and quantitatively preliminary sized and hereafter the functional view of this candidate, taken as an example out of the trade of study, will be explained following the Harmony methodology.

The functional analysis starts over again, considering each use case from the Use Case Diagram, black box activity diagrams, sequence diagrams, internal block diagrams and state machine chart diagrams are defined in a more detailed level, eliciting peculiar functions of the solution one, under the considered use case. Each action described in the functional analysis for the selected candidate will drive the definition of updated requirements.

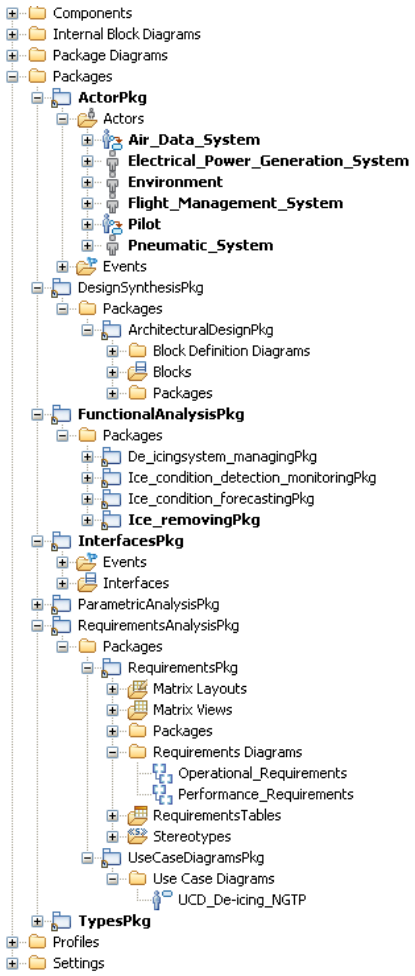


Figure 5-13: Explorer view of the Rhapsody Project

Figure 5-14 provides an overview of the Black Box Activity diagram of the Solution 1 – De-icing cycle definition. The following Figure 5-15, highlights a part of the “Ice Removing” Use case black box activity diagram.

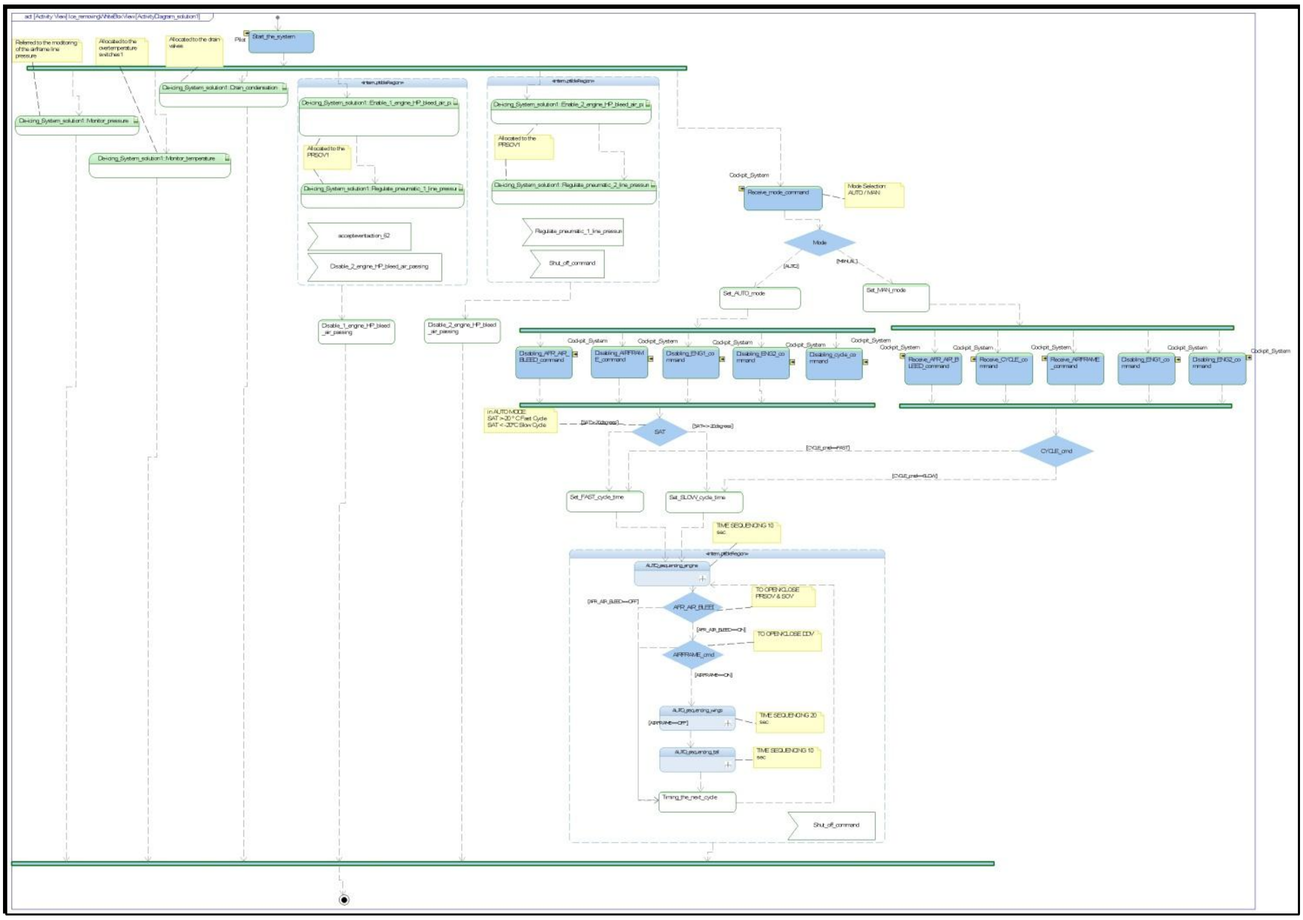


Figure 5-14: Refined Functional View for de-icing concept using boots

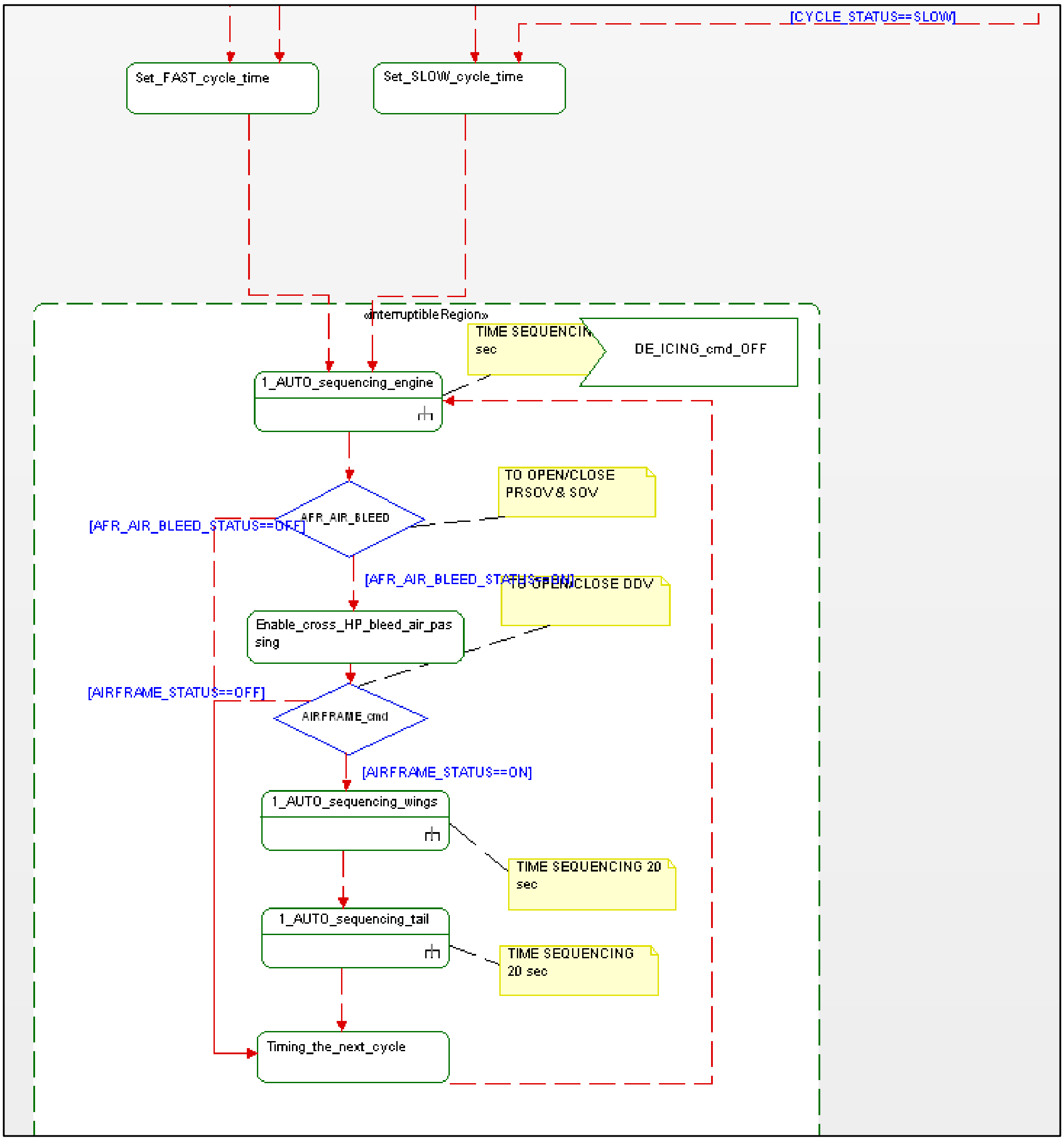


Figure 5-15: Part of the Black Box Activity diagram of the Solution 1 – De-icing cycle definition.

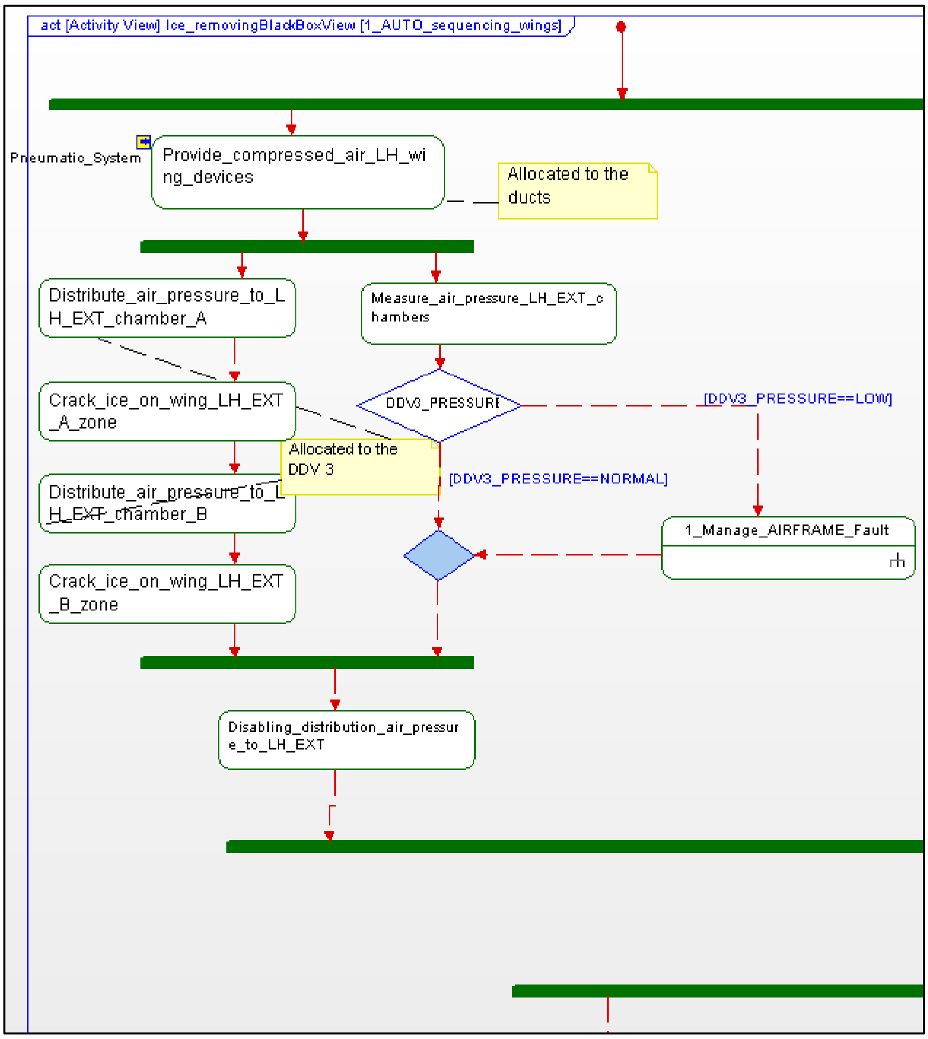


Figure 5-16: Part of the Call-behaviour embedded in the Black Box Activity Diagram of the Solution 1 – Left Hand wing ice removing.

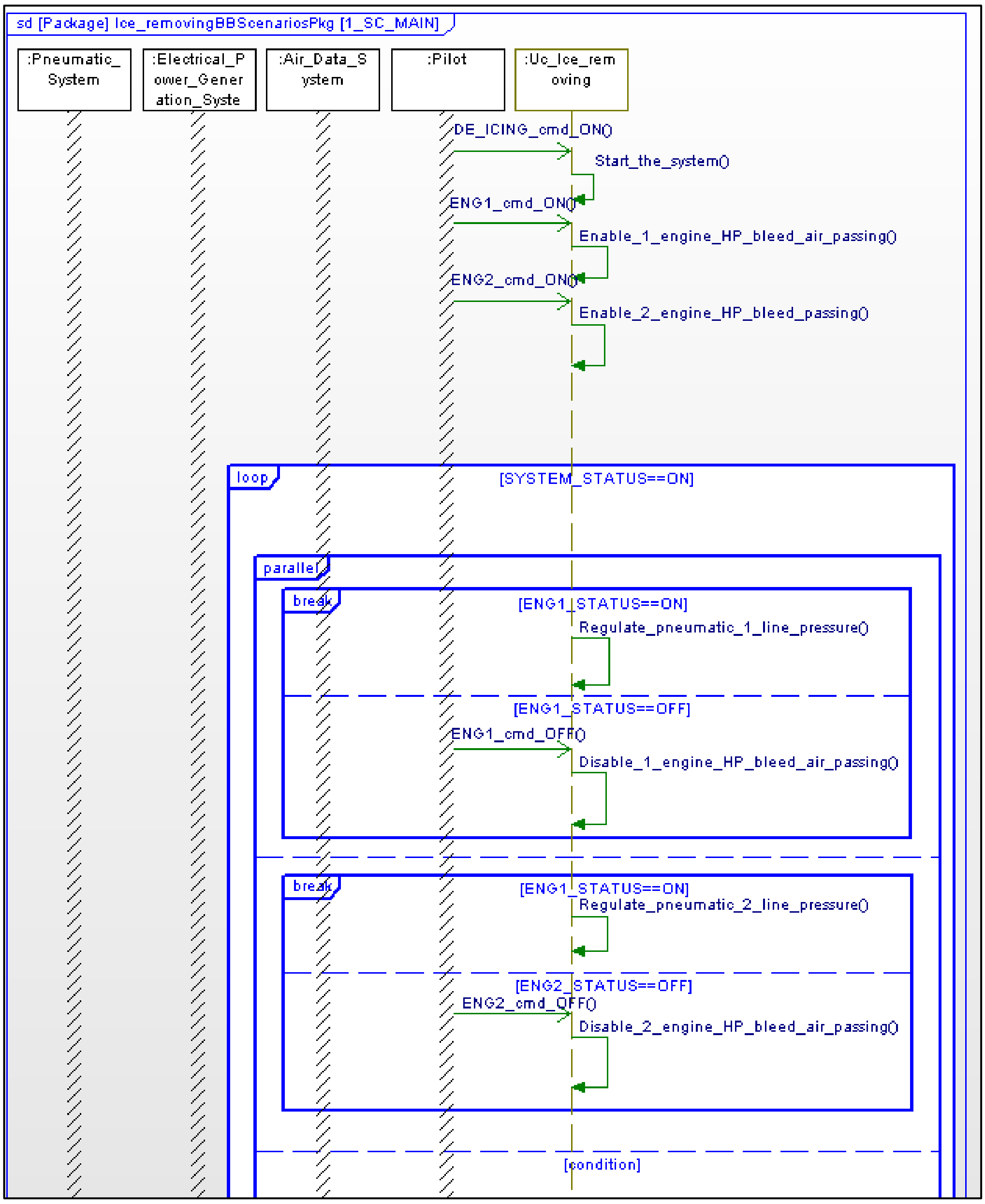


Figure 5-17: Part of the Sequence Diagram from the activity diagram of solution 1

The Internal Block Diagram, helps the Engineer to focus on the interactions between the system and the actors, showing a number of blocks representing the system and the actors involved during the system functioning. Those actors and the system are interconnected via ports (standard or flow) and connectors.

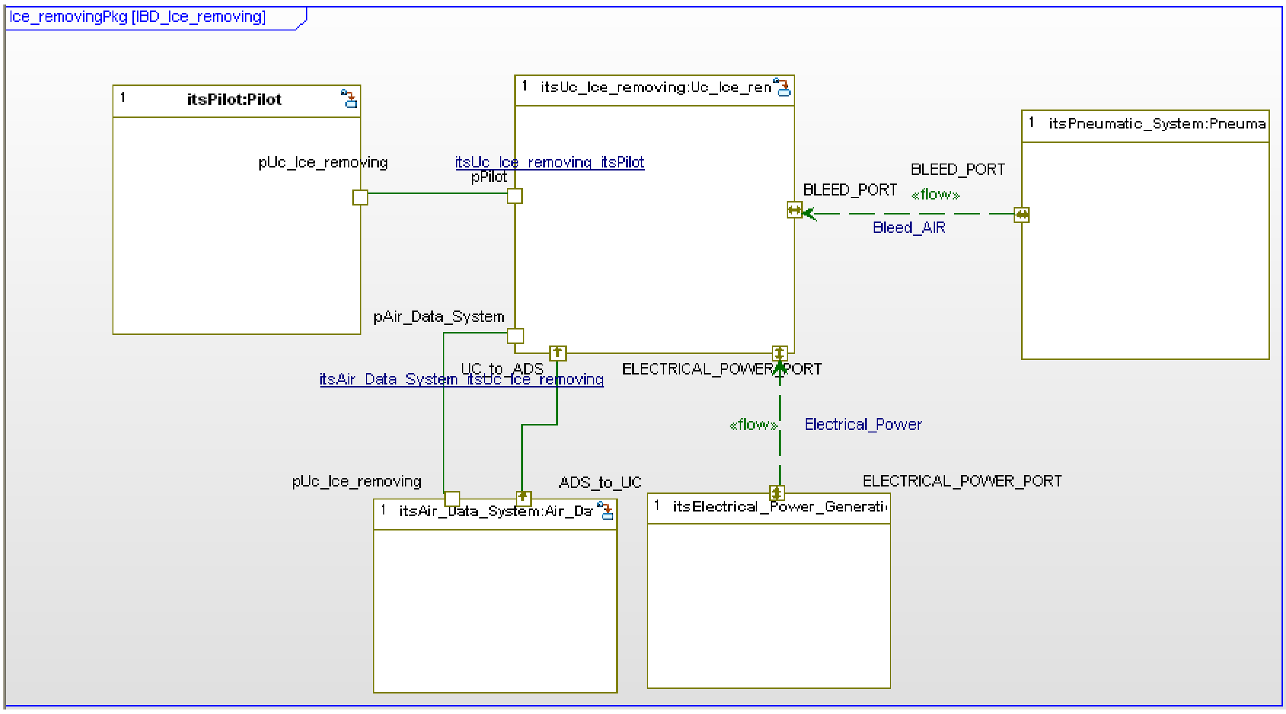


Figure 5-18: Internal Block Diagram for the solution one.

The next step in the Functional analysis for the selected solution is the modelling of the Statechart Diagram. This diagram describes the state-based behaviour of a block. It groups the information from both the activity diagram and the sequence diagrams.

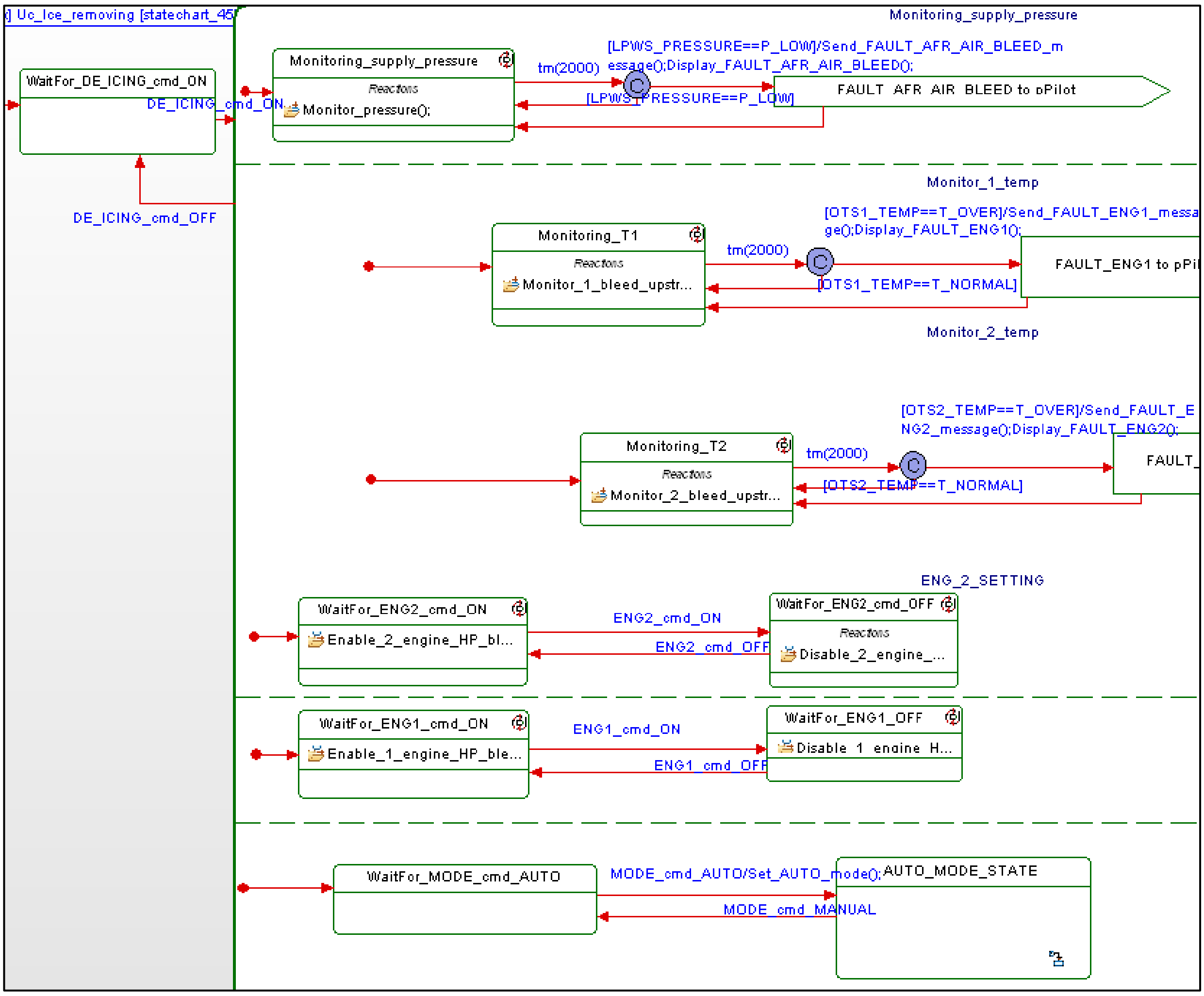


Figure 5-19: Part of the Statechart diagram for the solution one.

Once the functional analysis has been completed, the design synthesis takes over in finding the architectural setup for the solution considers, which results in pre-liminary product breakdown structure (PBS) as shown in Figure 5-20. The next step would consists in the definition of a System Architecture taking into account PBS and refined functional view.

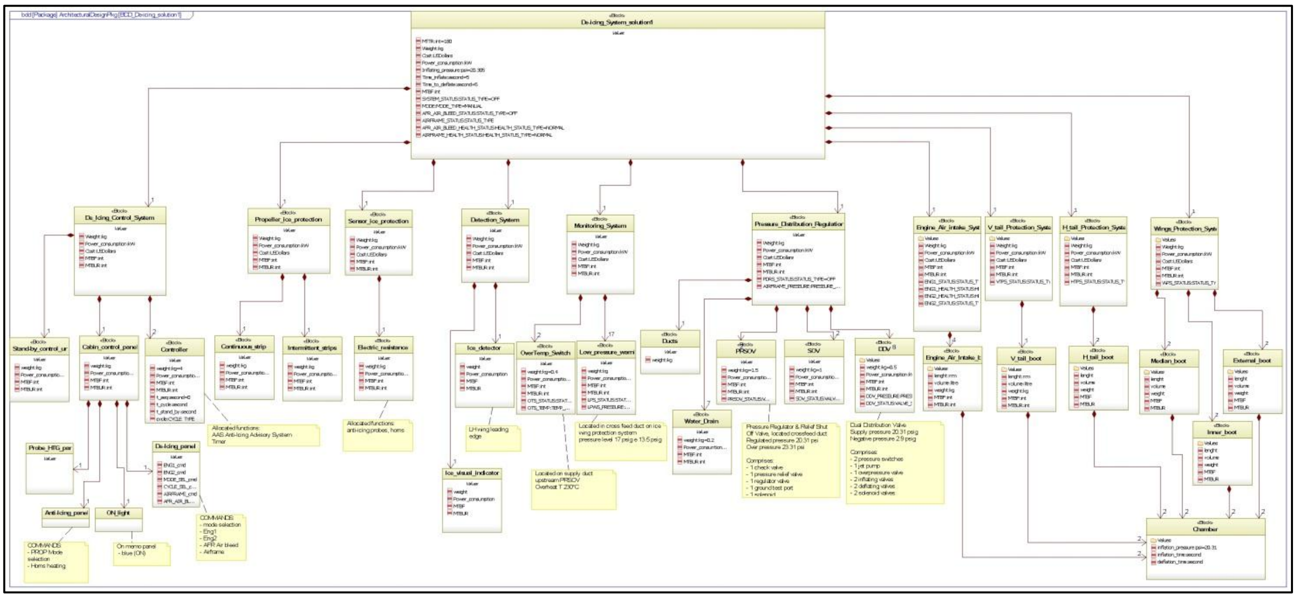


Figure 5-20: Part of the Block Definition Diagram representing a pre-liminary PBS for the solution one (boots)

5.1.4 Ice accretion and elimination physics

In order to calculate the actual values for the Measure of Effectiveness (MoE) as described in Figure 5.11, for some MoE it is necessary to define and run physical behaviour simulation models for the different solution candidates. For example, to determine the operational cost of a de-icing solution concept, it is required to compute the amount of ice that is being created on aircraft components for a given flight scenario, and to compute the consumption of variable goods (mainly electrical power) to eliminate that amount of ice. In the frame of the Public Aerospace use Case we have started to define such physical behaviour models using both Modelica and Matlab.

In order to compute the total ice formed on an aerodynamic surface in a very preliminary way, it is necessary to make a number of assumptions, referring to the simplified model descripted in "O.Meier, D.Scholz. *A Handbook Method for the estimation of power requirements for electrical de-icing systems*, Hamburg University of Applied Sciences, Aero – Aircraft Design and Systems Group. Berliner Tor 9, 20099 Hamburg.", and to the FAA/CS-25 Appendix C. A number of geometric inputs are required to run the static simulation which has been carried out in MATLAB:

- Wing's imaginary sieve surface and airfoil geometry, as an assumption for defining the impingement limits;
- Aircraft mission profile, defining altitude, speeds, time to climb, descent, and cruise (from requirements);
- FAA/CS-25 Droplet diameter (continuous maximum, intermittent maximum), and LWC definition;
- ISA Atmosphere.

The related mission profile flown by the aircraft is consistent with the initial top level requirements, and can be summarized as follow:

- Climb to operating cruise altitude of 20000 ft;
- Cruise for about 3 hours at 20000 ft at the max cruising speed;
- Descent.

The mission profile must be coupled with the temperature profile. In this context the worst case scenario has been taken into account, in which the temperature variation, during the climb, cruise and descent phase, allows for the maximum ice build-up (Intermittent + Continuous Icing Condition with the highest Liquid Water Content, using the superposition principle). A description of the Modelica Model for the worst case scenario can be found in the ANNEX Chapter 8.12. Chapter 8.13 illustrates the Modelica description of the ice accretion physics.

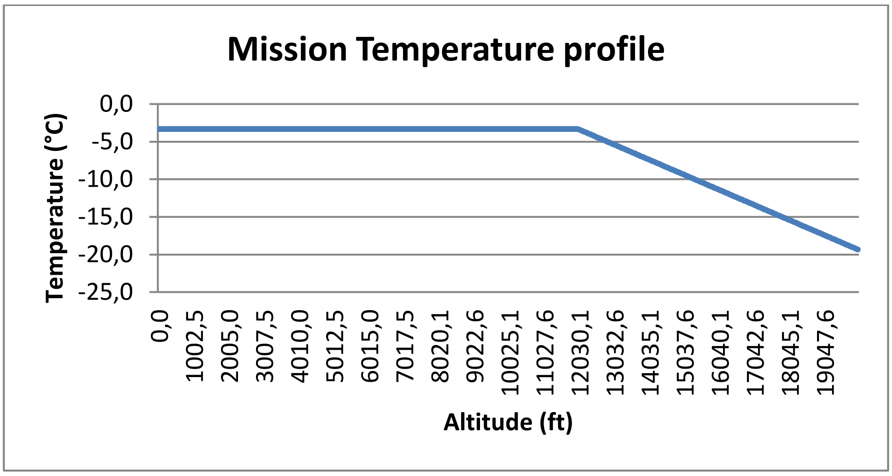


Figure 5-21: Mission temperature profile

As an example, the following results regarding the ice mass accretion for the isolated wing during the whole mission have been calculated:

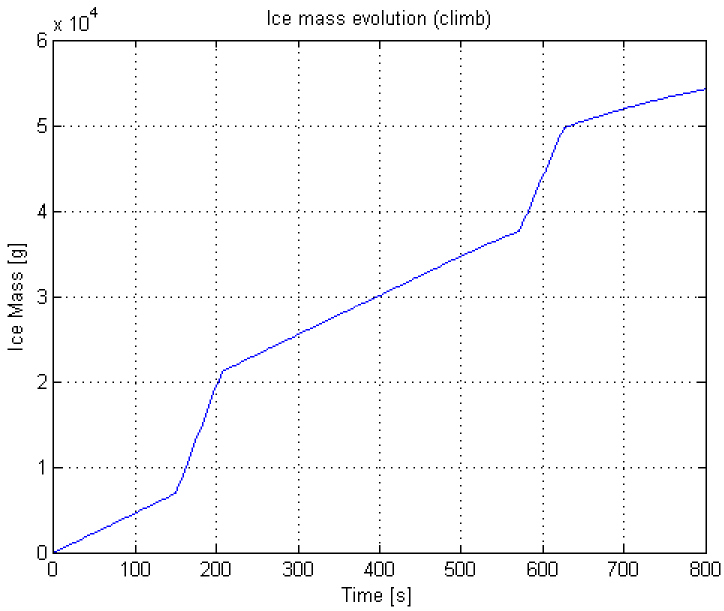


Figure 5-22: Ice mass evolution during the climb

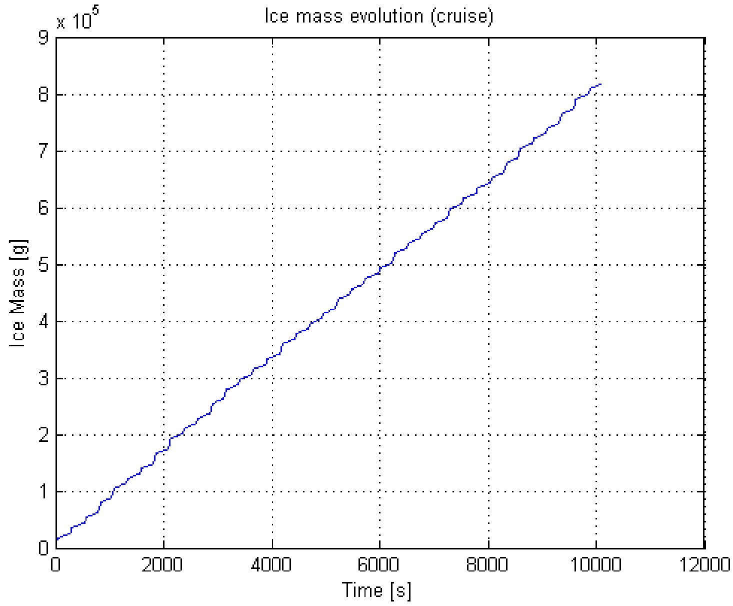


Figure 5-23: Ice mass evolution during the cruise

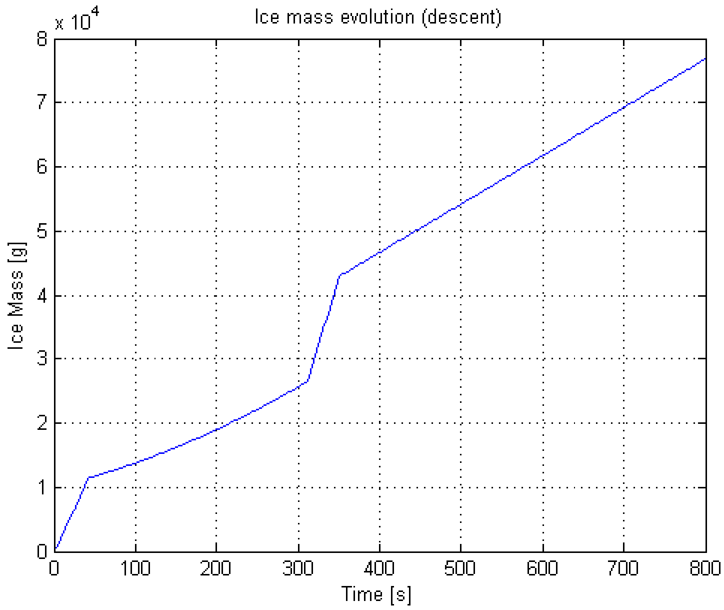


Figure 5-24: Ice mass evolution during the descent

Having the Ice mass computed as well as the airfoil geometry, the Ice thickness can be elicited (considering only the 5% of the airfoil chord affected to ice accretion as a first approximation), and the final results for the worst case mission can be illustrated with the following illustration:

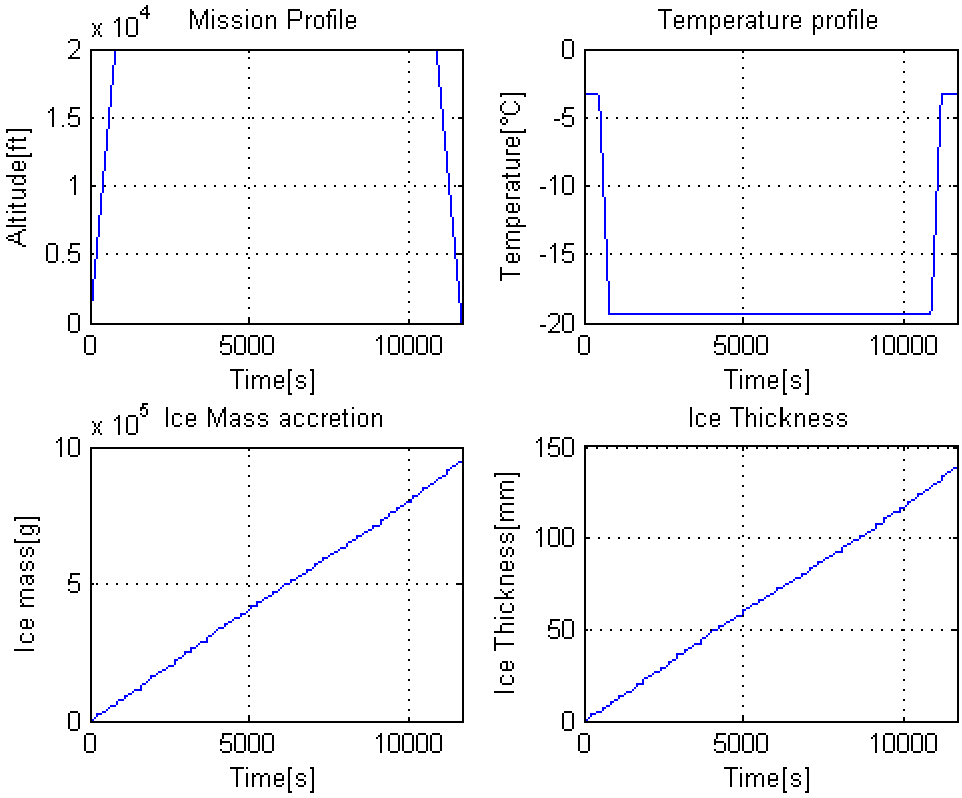


Figure 5-25: Ice accretion final results

5.2 Engineering Method implemented on Demonstrator - Change Impact Analysis / Traceability

In collaboration with WP6.11, we have started to implement the Engineering Method “Change Impact Analysis” as described in Chapter 3.7 by using the demonstrator set-up described in Chapter 4.

The main purpose of this engineering method is to support users to evaluate the impact of a change request on the data that has been defined for a System under Development.

We have decided to start with this engineering method for various reason:

- Many assets needed to implement this engineering method are already existing, at least on prototypically level
- The change impact analysis method has been rated by many partners as a very relevant scenarios, even beyond the aerospace industry.
- The change impact analysis provides some of the foundations needed for other engineering methods (especially for trade-off analysis related engineering methods)

In order to implement of this engineering method, we have first defined a brief traceability concept that describes how to link the different types of data defined for the De-icing System.This traceability concept defines e.g. that a Function represented on a functional view has to have a traceability link of type “satisfy” with a top level requirement.

As a next step, we have then set-up concrete tracability between different data links using the IBM SSE environement described in Chapter 4.

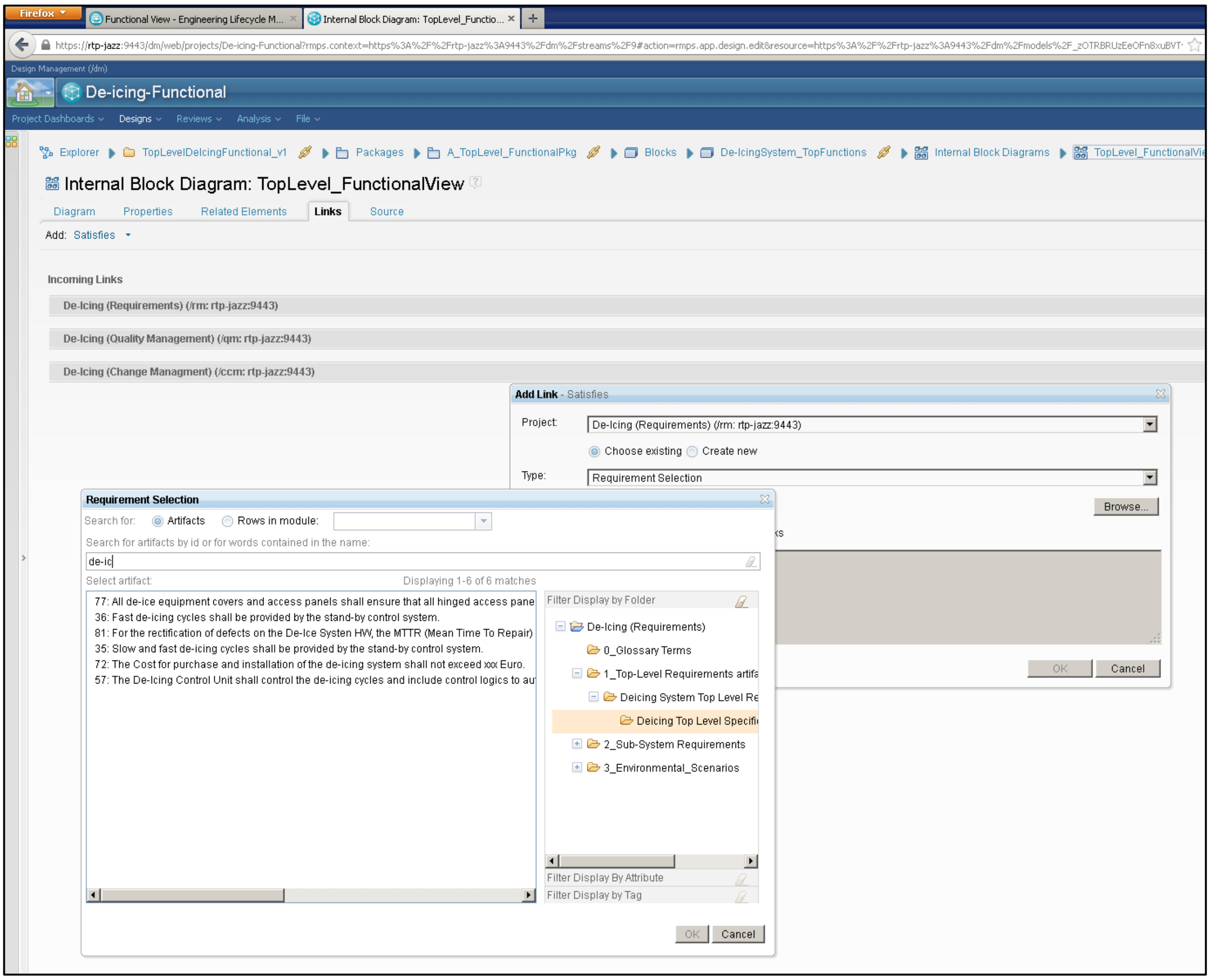


Figure 5-26: Establishing traceability links for the De-icing System using IBM SSE 4.03

Figure 5-26 illustrates the setting-up of a link between two data elements of the De-icing System. The source of the link is an Internal Block Diagram that represents the top level functional view (see upper part of the figure). By selecting “add links”, a new window appears that allows to select an application that is connected the System Engineering Environment (see window “add Link” on the right side of the figure). In this example we have selected the Requirements application. As a result of this selection, a new window appears (left side of the figure) that allows to query amongst all data managed by the Requirements application. We can reduce the results of the query by filtering for requirements that contain a certain phrase, e.g. “de-ice” as shown in Figure 5-26. The link has been established when selecting one of the requirements as target.

At the time of writing of this deliverable (January 2014), we have established around 100 links in a similar way between all types of data existing for the de-icing use case.

Setting-up links between data is a pre-requisite for implementing the Change Impact Analysis Engineering Method. A second step concerns the application a tool that allows to navigate between all data, and that illustrates the relationships between the data in a comprehensive way. In our case we have used the IBM RELM 4.0.3 tool for this purpose (see also Chapter 4.2).

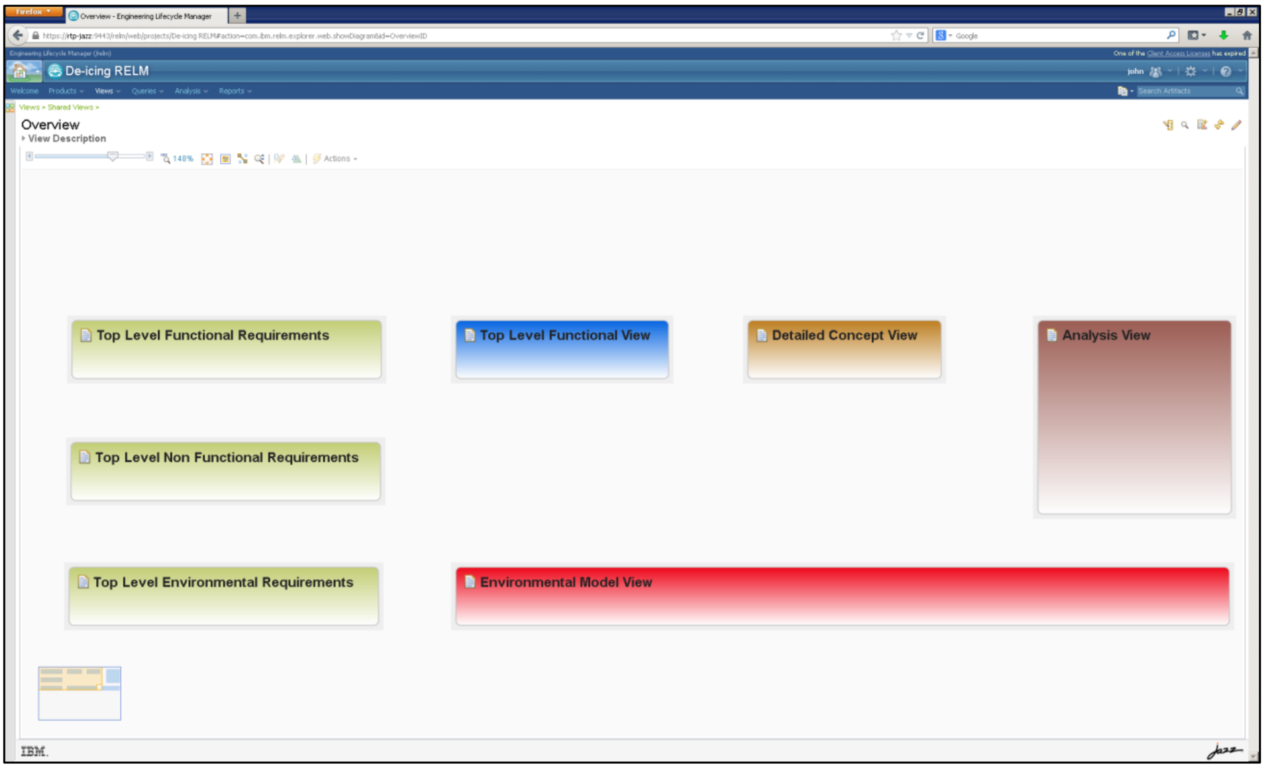


Figure 5-27: Views defined in IBM RELM for de-icing use case

Figure 5-27 gives an overview of the views that we have currently defined for the de-icing system. We distinguish on one hand between functional-, non-functional-, and environmental requirements. Furthermore, we have defined a Top Level Functional View, a Detailed Concept View, an Analysis View, and an Environemental Model View. When selecting one of these views, detailed information about data and relationships for that view will be provided to users.

Figure 5-28 illustrates the information provided to the user when selecting the Top Level Functional View. This Functional View includes all Top Level Functions, all Top Level Requirements of type "functional", all Actors, SuD, and external systems of the System Context, and all relationships between these data. We can see e.g. that the Function "Apply Anti Icing Measurements" has a satisfy relationships with several top level functional requirements, such as Requirements 48, 49, 50 (highlighted in orange). It also has relationships with the Electrical Power Generation System, the Pneumatic System, and the SuD Ice removing (right side of the figure). The information itself is coming from different tools and databases, i.e. Doors NG with JTS Database for the Requirements, and IBM Rhapsody Design Manager for the Top Level Functions and the System Context.

For each of the data elements represented on the Functional View (and on all other Views defined in RELM), additional information can be requested. Figure 5-29 shows this approach for the element "TopLevel_FunctionalView" – when hovering over this element, a request to provide additional information for this element is send to the Rhapsody Design Manager that stores the element, and the result (e.g. a preview of the related diagram) is send back to RELM.

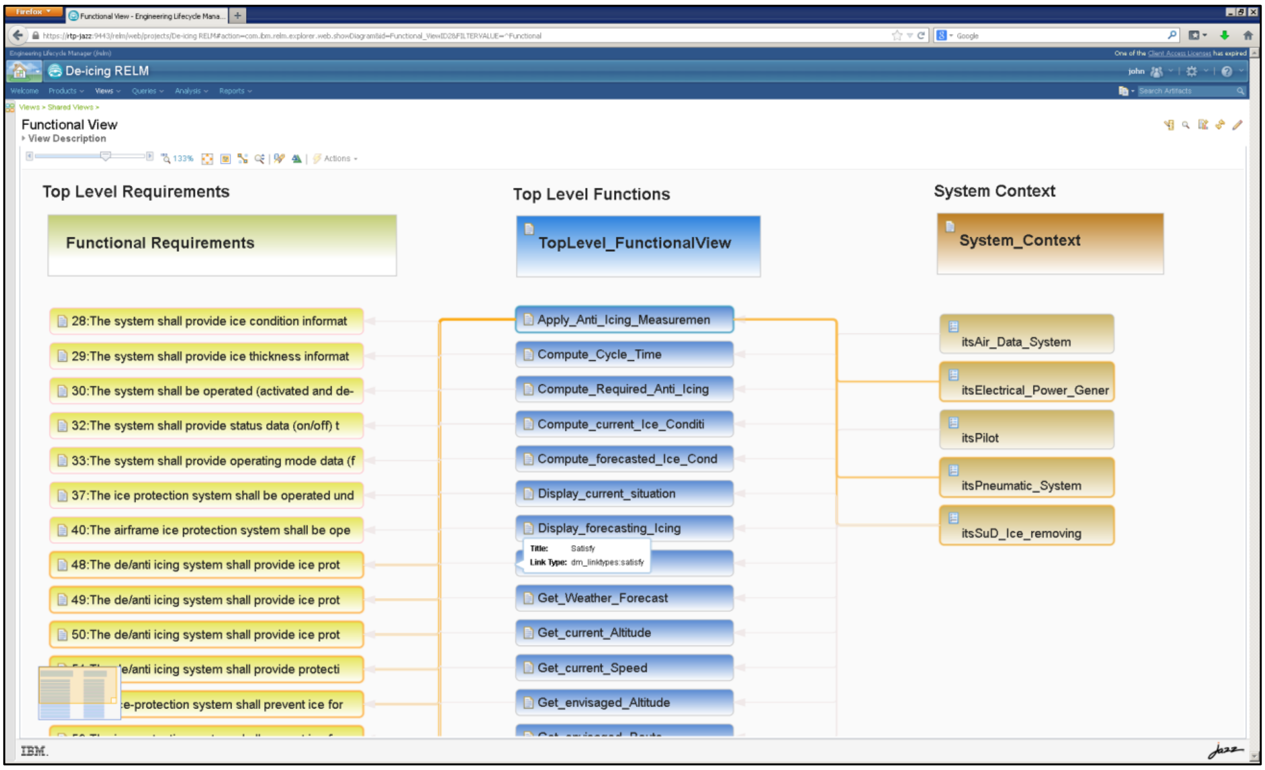


Figure 5-28: Top Level Functional View as implemented in IBM RELM

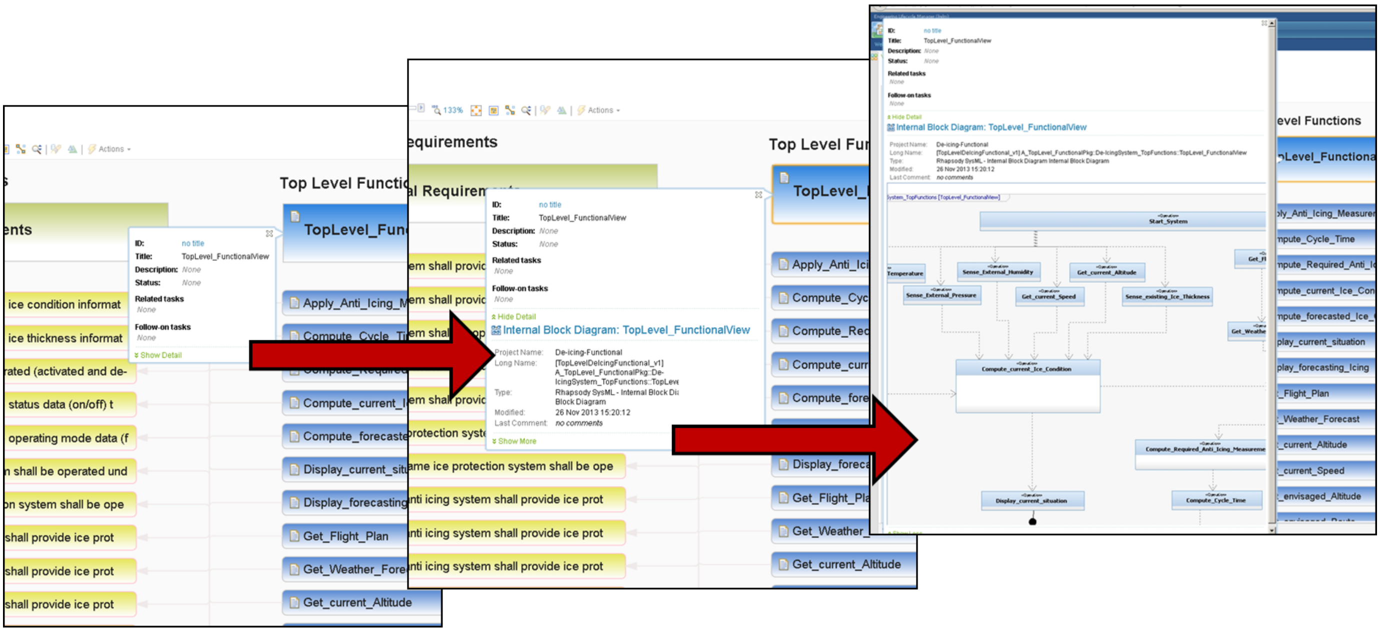


Figure 5-29: Requesting additional information for a data object using RELM

Figure 5-30: Additional Views defined in RELM for the De-icing System

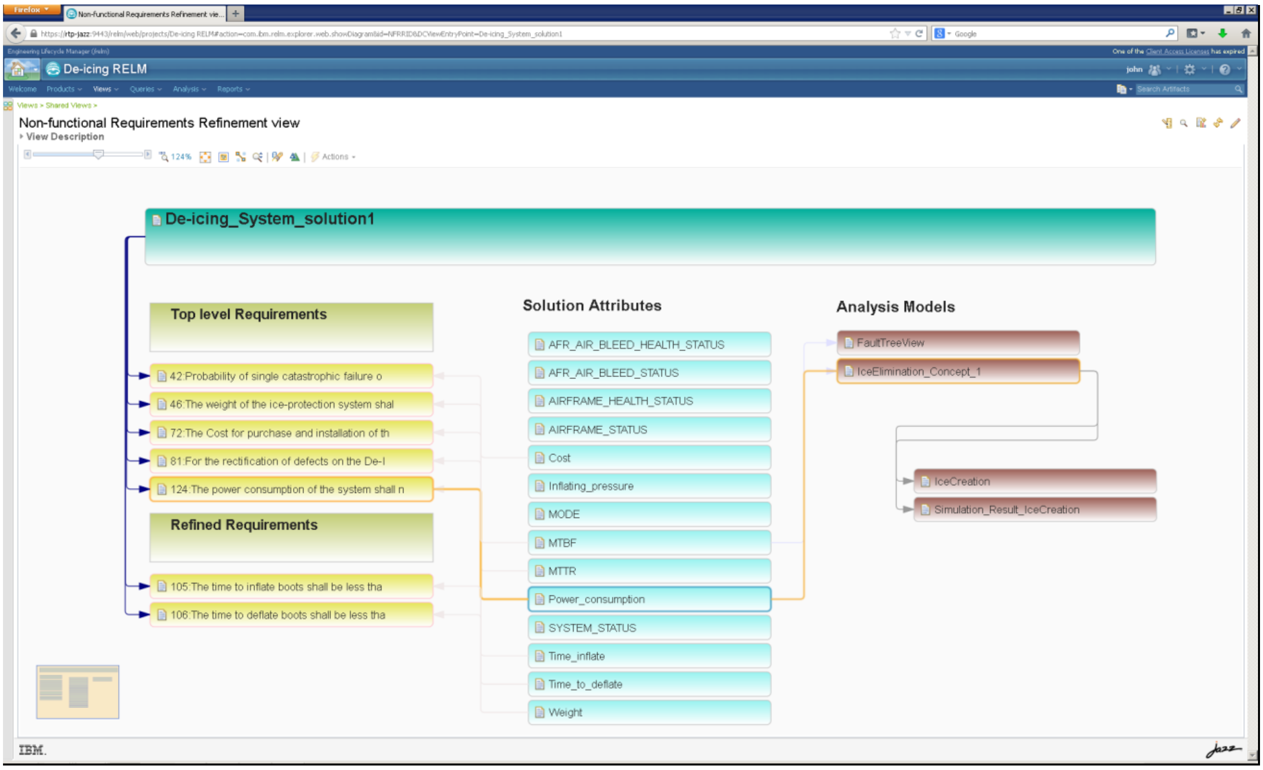


Figure 5-31: Traceability to Analysis Models

Being able to easily navigate between data helps users to better understand the impact of a change request. For example, in Figure 5-31 the links for the metric “power consumption” for the De-icing System solution 1 are highlighted. A change request on the Top Level Requirement for power consumption resulting in a reduction of the maximum allowed power consumption would impact an analysis model called “IceElimination Concept 1”. This analysis model has further relationships to an Ice Creation Model and to a

Simulation Results artefact. These are thus the data elements that would have to be further analysis before accepting the change request on the top level requirement.

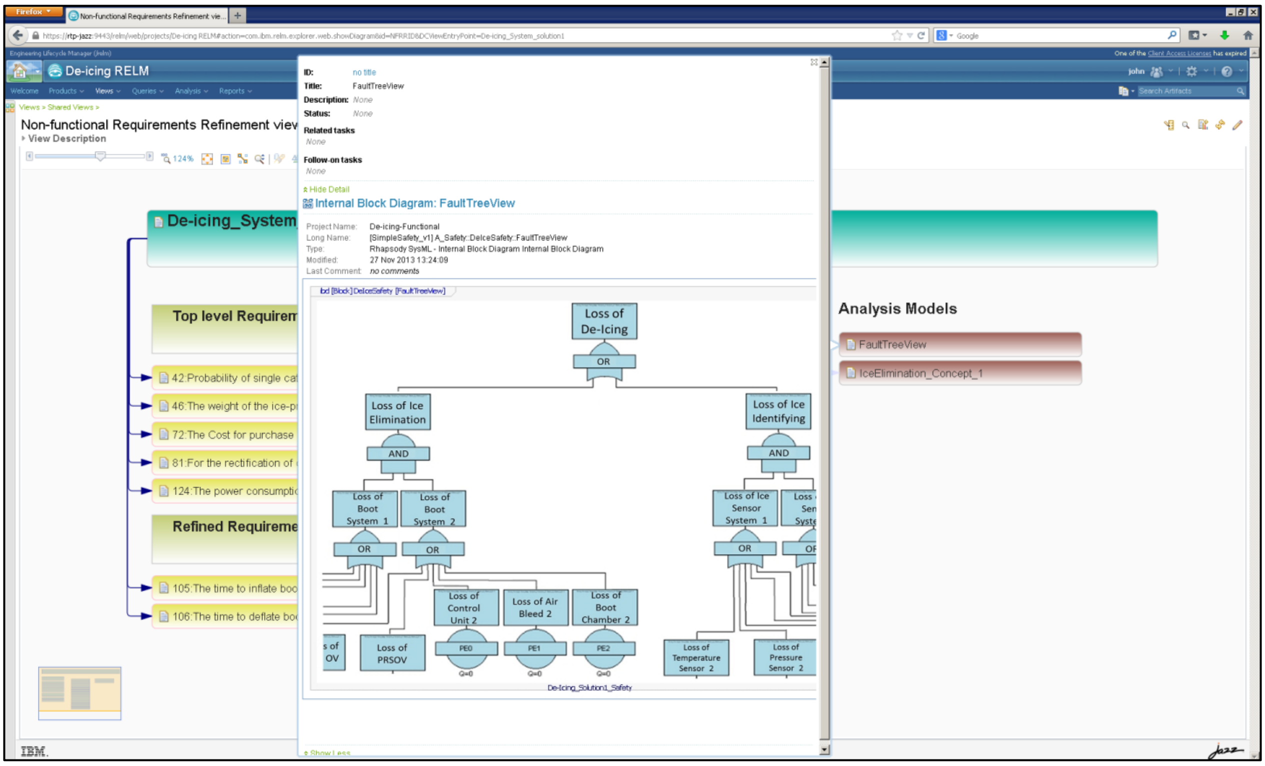


Figure 5-32: Envisaged traceability from design to safety analysis

At the moment the traceability and change impact analysis scenario described in this chapter works for tools that have already an OSLC connection to the IBM SSE environment. Figure 5-32 provides an illustration about one of the next steps that we intend to realise on this scenario. It shows the preview of a fault-tree analysis model that is linked to the MTBF (Mean Time between Failure) metric and further to a Safety Requirement for the maximum allowed failure probability of the system. However, at this moment this connection to a Fault-tree analysis model is not yet working, since it would require having an IOS connector of a safety analysis tool. Our next steps are therefore to add additional analysis tools to the demonstrator using the CRYSTAL IOS approach.

6 Conclusions and Way forward

In this deliverable we have presented the Public Aerospace Use Case study that is related to the specification, design and analysis of a de-icing system for a regional aircraft. We have further presented a first set of engineering methods that describe typical aerospace challenges with regard to interoperability between different engineering domains and the used tools as well as the exchange of data/information in the context of an extended enterprise. We have also provided information about a first demonstrator implementation of the engineering method "change impact analysis" using the IBM SSE and RELM 4.0.3 environment. The reason for using this IBM solution as basis for the demonstrator is primarily based on the fact that it provides already OSLC connectors and therefore allows us to have some first impression on the usefulness of the CRYSTAL IOS approach.

First results from the current demonstrator for the public aerospace have shown that the CRYSTAL IOS approach is practical and can enable scenarios that provide significant benefits to end-users.

We have also identified a variety of open topics that we intend to address in the upcoming months. The next short term step concerns the connection of an additional tool to the current demonstrator environment that is not part of the current IBM SSE solution, in order to assess modularity and extensibility of an OSLC based IOS connector approach.

A longer term objective is to integrate several ALM and PLM solutions with each other, e.g. IBM SSE with Siemens Teamcenter and PTC Integrity and Winchill. Such an integration scenario would allow us to address additional integration challenges related to configuration management of complex System Engineering Environement (SEE) involving many single Product Lifecycle Management solutions.

Also, we intend to address non-functional topics for SEE, such as ensuring availability and validity of data and data relationships over the entire lifecycle of an aircraft (which can be many decades). This also includes the upgradebility of the used software products (for any reasons,e.g. security, new features) with regard to the migration of data and to larger scale IT architectures (from single server to distributed server farms).

An additional open topic concerns the integration of different company networks into the demonstrator environment in order to simulate extended enterprises.

With regard to Engineering Methods, we intend to implement Engineering Methods specified in this deliverable on a step-by-step approach with support of SP6. We further intend to refine those Engineering Methods that have not yet been reviewed in detail with SP2 partners (especially Safety related Engineering Methods). Also, we are aware that some key topics for the aerospace industry have not yet been addressed at all in WP208, such as Feature modelling and Product Line Engineering. For those topics we intend to define additional engineering methods.

Finally, an important topic to address in the near future concerns the integration of concepts around domain ontologies as defined by WP209.

7 References

CRYSTAL consortium; 2013	Annex I - "Description of Work"
Prel_Sizing_rev_2013	Public use case anti-ice systems for the next generation turboprop
Estimation of power requirements for de-icing systems	"O.Meier, D.Scholz. *A Handbook Method for the estimation of power requirements for electrical de-icing systems*, Hamburg University of Applied Sciences, Aero – Aircraft Design and Systems Group. Berliner Tor 9, 20099 Hamburg.", and to the FAA/CS-25 Appendix C.

8 Annex

8.1 Minute of Meeting Nov 12th 2013 – Review of Engineering Methods

Minutes of meeting

Date:	**November 12th, 2013**	**Time:**	**14:00 – 16:00**
Subject:	**Review of Engineering Methods to be covered by WP208**		
Location:	**N/A**		
Type of Meeting:	WebEx		
Author/Company:	Andreas Mitschke; EADS IW G		

Participants:	**Company Short Name**	**Cc:**	**Company Short Name**
Andreas Mitschke	EADS IW G	Andreas Keis	EADS IW F
Luciana Loverde	ALA	Maurizio Morisio	POLITO
Francesco Brunetti	POLITO	Federico Tomassetti	POLITO
Claudio Pessa	ALA	Sergio Chiesa	POLITO
Anne Monceaux	EADS IW F	Uwe Kuehne	EADS-CAS
Ralf Bogusch	EADS-CAS		
Marc Malot	SAGEM		
Ivo Viglietti	ALA		
Odile Laurent	A-F		
Parham Vasaeili	EADS IW UK		
Gray Bachelor	IBM UK		
Thomas Kuhn	IESE		
Soeren Schneikert	IESE		
Rubén de JuanMarín	ITI		

No.	**Type**	**Topics/Remarks**	**Who**	**Until**
1.	I	The Minute of this Meeting is stored on AVL Sharepoint : AVL Project World / CRYSTAL / Data exchange / 001_Meetings / 007_SP2_Meetings / WP28		
2.	I	The purpose of this web meeting was to review the engineering methods proposed in the frame of WP208 with SP2 industrial partners and with members of the IOS Core Team. Basis for review and discussion of WP208 Engineering Method was the presentation "2013 10 17 CRYSTAL WP208 - Public Aerospace Use Case - Engineering Meth.pptx" that has been send out by e-mail to all participants prior to the workshop (AVL Project World / CRYSTAL / Data exchange / 001_Meetings /007_SP2_Meetings / WP28 /Use Case Descriptions.		
3.	I	In the following, the comments received from participants of the web meeting for each of the engineering methods are being listed. In addition, participants from		

Minutes of Meeting, 16.01.2014 page 1/4

Minutes of meeting

		EADS Cassidian have provided additional and more detailed feedback by e-mail. Not all Engineering Methods have been reviewed due to time restriction. Therefore, a follow-up meeting has to be planned.	
4.	S	Engineering Method "Trade-off Analysis 001": - Cassidian: Relevant scenario, but the scenario should be extended to show how constraints are being identified from requirements. For example, a query or filter service could be applied in the first place on a huge list of requirements in natural language format (e.g. to show only requirements of type "weight", or "safety"). - Airbus: Important and representative scenario - Sagem: Important and representative scenario - Alenia: Relevant scenario. Requirements could have attributes or types in order to support a filtering service. Other comments: - There was a discussion among participants where the semantics needed to identify constraints out of natural language requirements would be specified (e.g. in OSLC, or in a dedicated ontology, or in a dedicated natural language processing tool.	Ralf Bogusch Odile Laurent Marc Malot Claudio Pessa, Luciana Loverde
5.	S	Engineering Method "Heterogenous Simulation": - Cassidian: Less relevant scenario - Airbus: High priority, scenario is well defined - Sagem: Not important, will not use it - Alenia: Not important Other comments: - There is an error in the current word deliverable where this EM is described (Chapter 3.3), since Doors is mentioned for point 9 but not part of the EM. - I like the idea of using ontologies to extend OSLC and other standards - We should clarify in CRYSTAL what shall happen when one of the tools involved in this or another EM is offline. E.g. we could use caches or central server to guarantee available of linked data.	Ralf Bogusch Odile Laurent Marc Malot C. Pessa, L. Loverde Christian El Salloum Thomas Kuhn

Minutes of Meeting, 16.01.2014 page 2/4

Minutes of meeting

6.	S	Engineering Method "Analyse Requirement": - Cassidian: Highly relevant scenario - Airbus: Less relevant scenario - Sagem: Highly relevant. In the scenario RQA should be renamed to RQS. - Alenia: medium relevance, scenario should be modified so that requirements managed by the Requirements Database can be better filtered before sending them to a Quality Analysis tool (e.g. check quality of requirements of ATA 24, or of type "weight", or of "version 2.3"). Other comments: - Quality information managed by RQS data-base should be correlated or at least linked with other Databases, e.g. to enable query related services such as "get all data with references to quality information". - We could define an additional EM that enables to expose the REUSE ontology to other tools.	Ralf Bogusch Odile Laurent Marc Malot C. Pessa, L. Loverde Marc Malot Anne Monceaux	
7.	A	Provide an update for the scenario "Analyse Requirement"	Marc Malot	November 22nd, 2013
8.	S	Engineering Method "Verify Design against requirements": - Cassidian: medium relevance; the last step of the scenario should be modified so that the Requirement in the Doors database is not automatically given a status "passed" or "failed", but instead a verification case is created and linked to the Requirement - Airbus: medium relevance, but the EM is not part of the Airbus use cases - Sagem: medium relevance, we could address it at the end of CRYSTAL using the Ontology. In terms of tools we would prefer Doors and Rhapsody. Also, the EM should include some consistency checking between Requirement and model + impact analysis. - Alenia: Not relevant	Ralf Bogusch Odile Laurent Marc Malot C. Pessa, L. Loverde	
9.	S	Engineering Method "Provide Specification Document": - Cassidian: Highly relevant scenario, but with different tools (Doors and Rhapsody), and we	Ralf Bogusch	

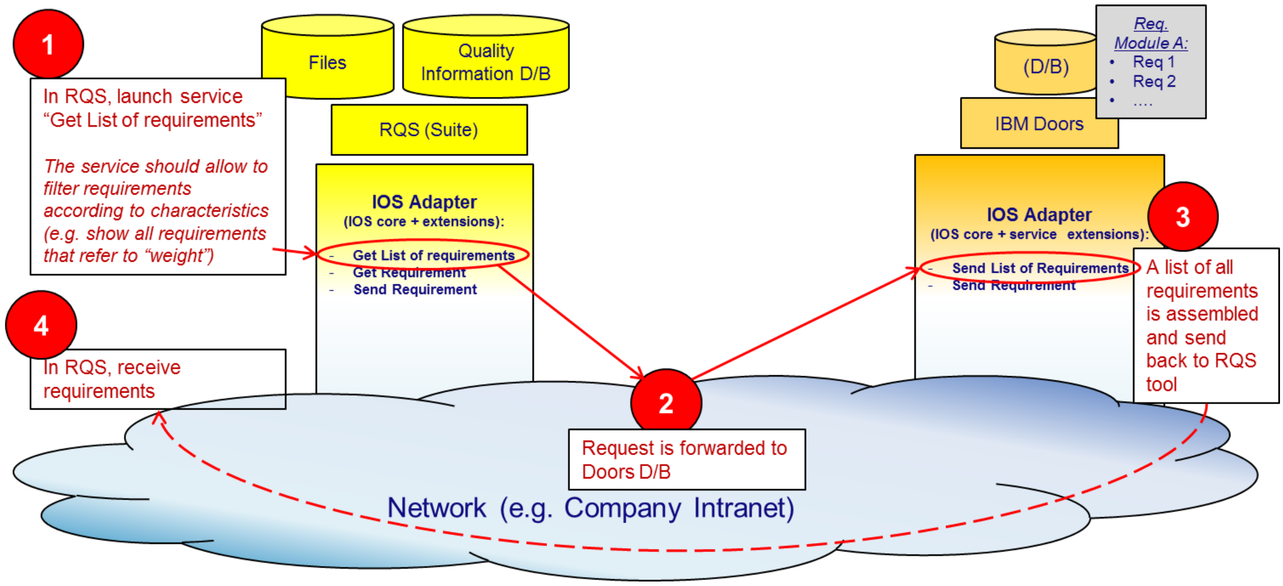
Minutes of meeting

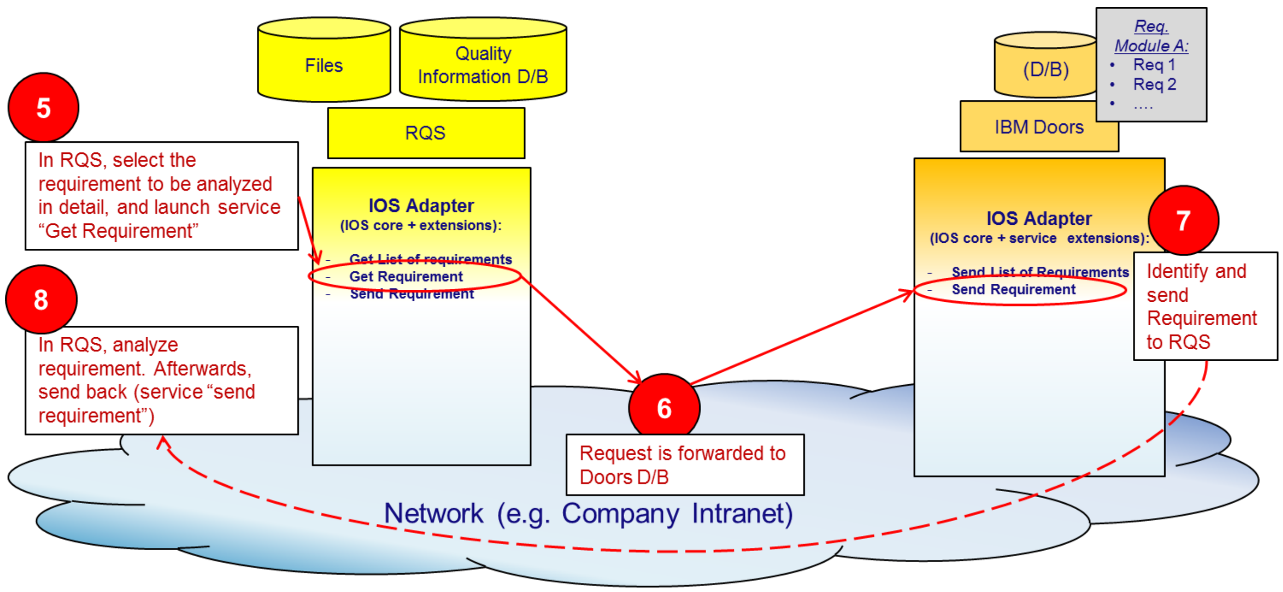
		need to clarify how to tag specification relevant data (e.g. by providing pre-defined tags, or use of containers to have well structured documents). - Airbus: Not relevant scenario - Sagem: Highly relevant.We want to use the ontology to support this scenario. Tools involved would be Doors, Rhapsody, and the ontology. - Alenia: Medium relevance. We have a similar scenario in mind which involves Doors, Rhapsody, and PTC; but the concept is slightly different.	Odile Laurent Marc Malot C. Pessa, L. Loverde	
10.	S	Engineering Method "Change Impact Analysis": - Cassidian: Very important scenario, but the workflow should be different; the Change Request should first be assessed before it is implemented. - Airbus: Interesting scenario, but not yet implemented. - Sagem: Interesting scenario, but not yet implemented. - Alenia: Interesting scenario, even if it is not part of our Use Case	Ralf Bogusch Odile Laurent Marc Malot C. Pessa, L. Loverde	
11.	I	This web conference was terminated after the review of the EM "Change Impact Analysis" on Slide 60 of the presentation "2013 10 17 CRYSTAL WP208 - Public Aerospace Use Case - Engineering Meth.pptx". The remaining Engineering Methods have to be reviewed at a follow-up web meeting.		
12.	A	Update the description of the engineering methods for WP208 according to the feedback received by participants of this web conference	Andreas Mitschke	December 20th, 2013
13.	A	Organise a follow-up meeting to review the remaining Engineering Methods of WP208.	Andreas Mitschke	December 13th, 2013

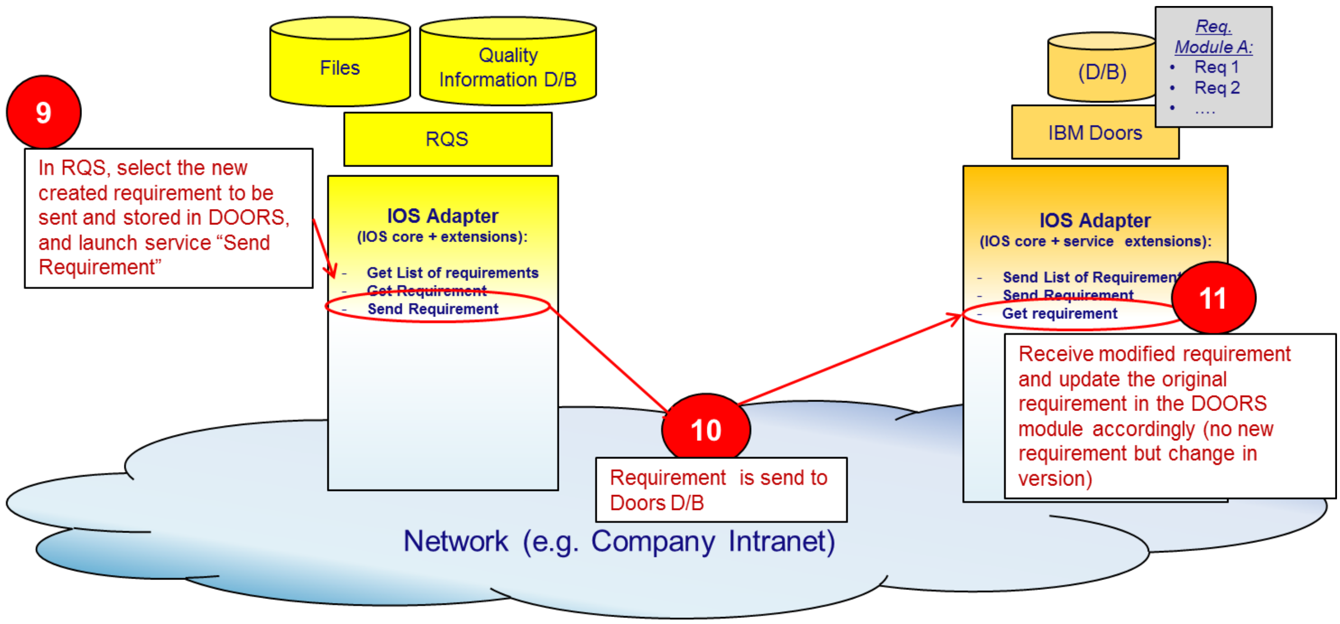
Type: **A** = Action (always with responsible partner and fixed date for completion)
D = Decision
S = Statement (always with person who made the statement)
I = Information (generally important)

8.2 Engineering Method - “Analyse Requirements”

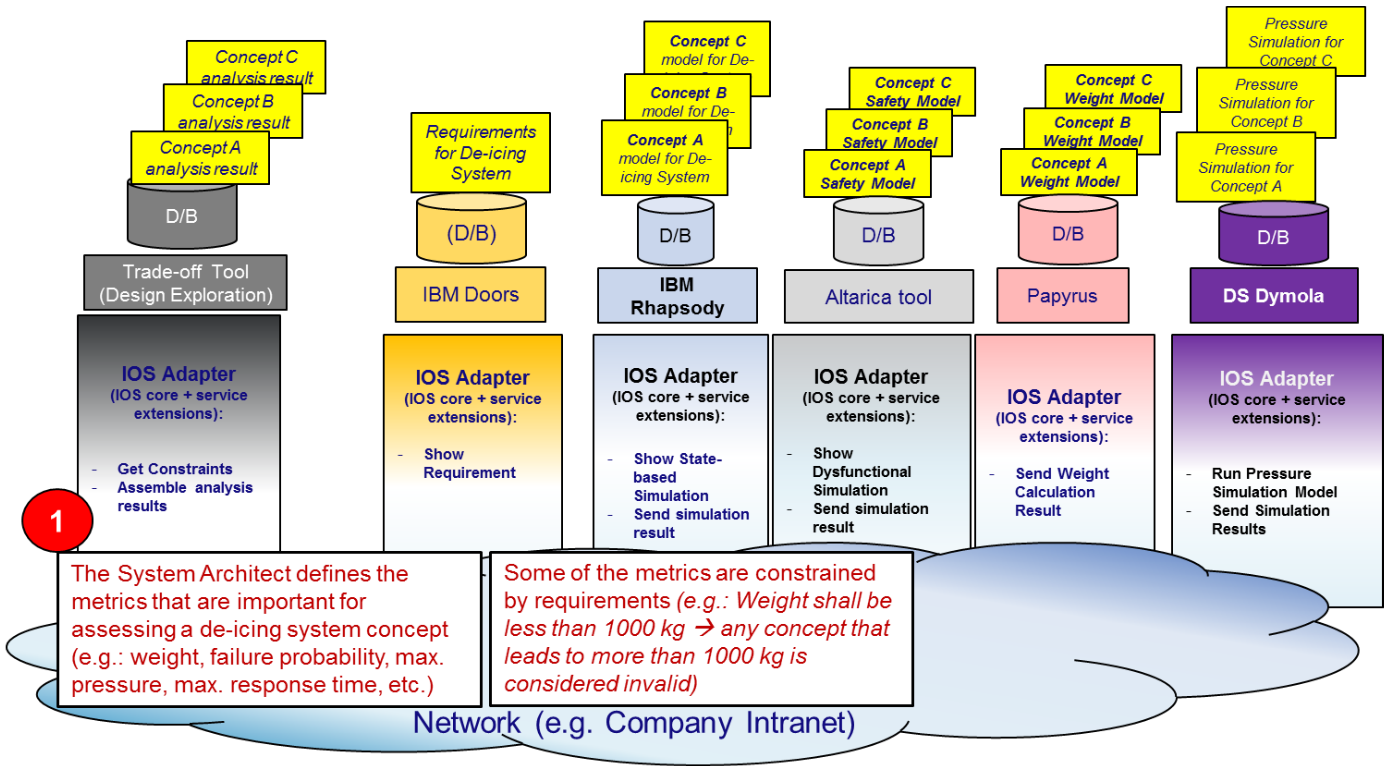
The following pictures provide an additional description of the Engineering Method “Analyse Requirements”

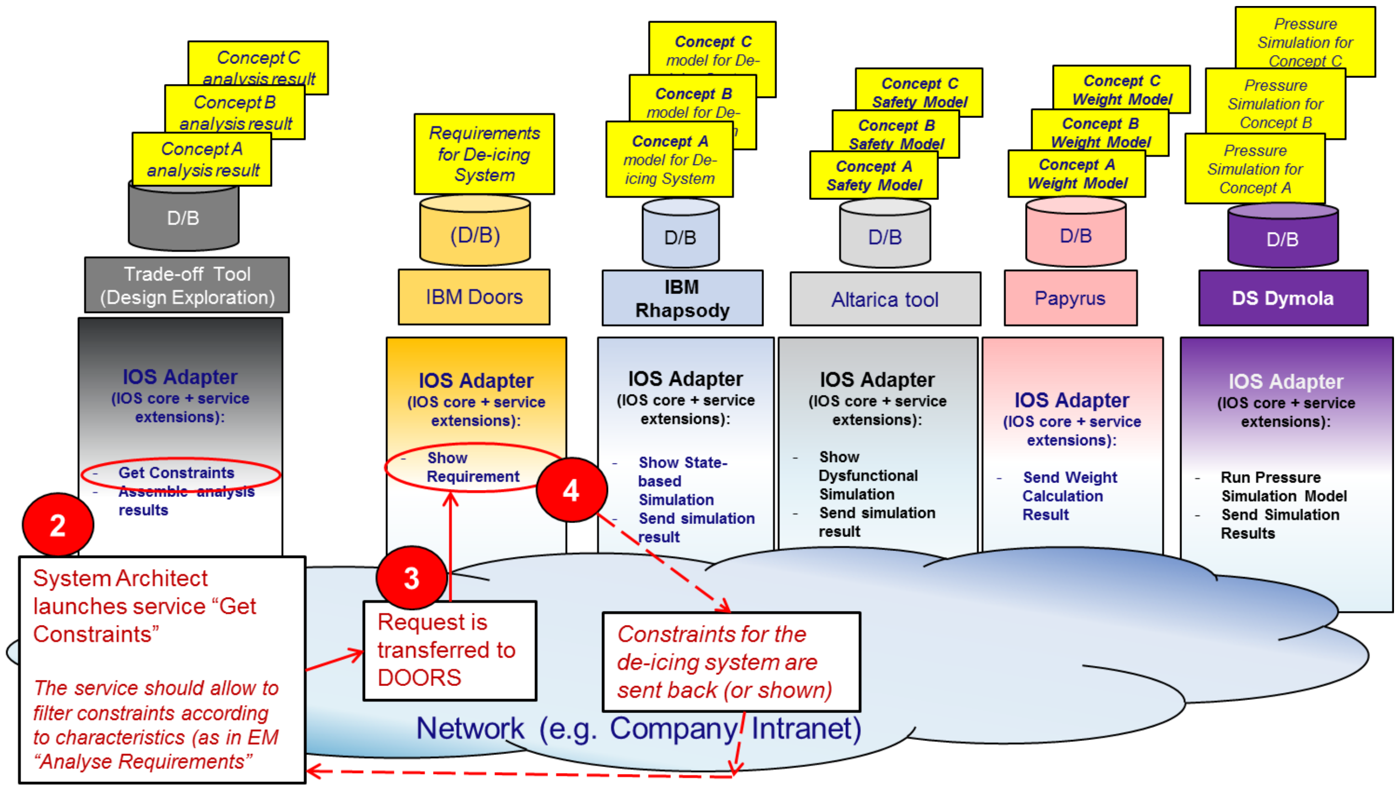


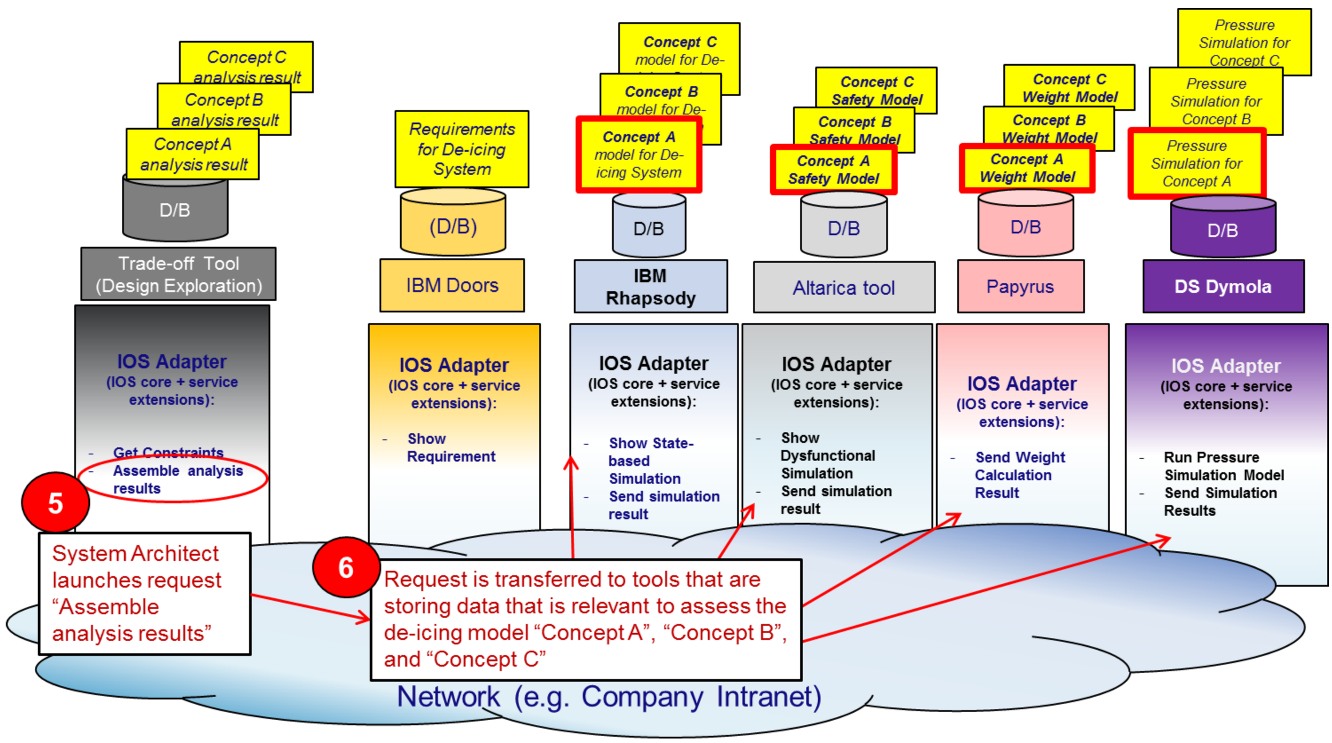


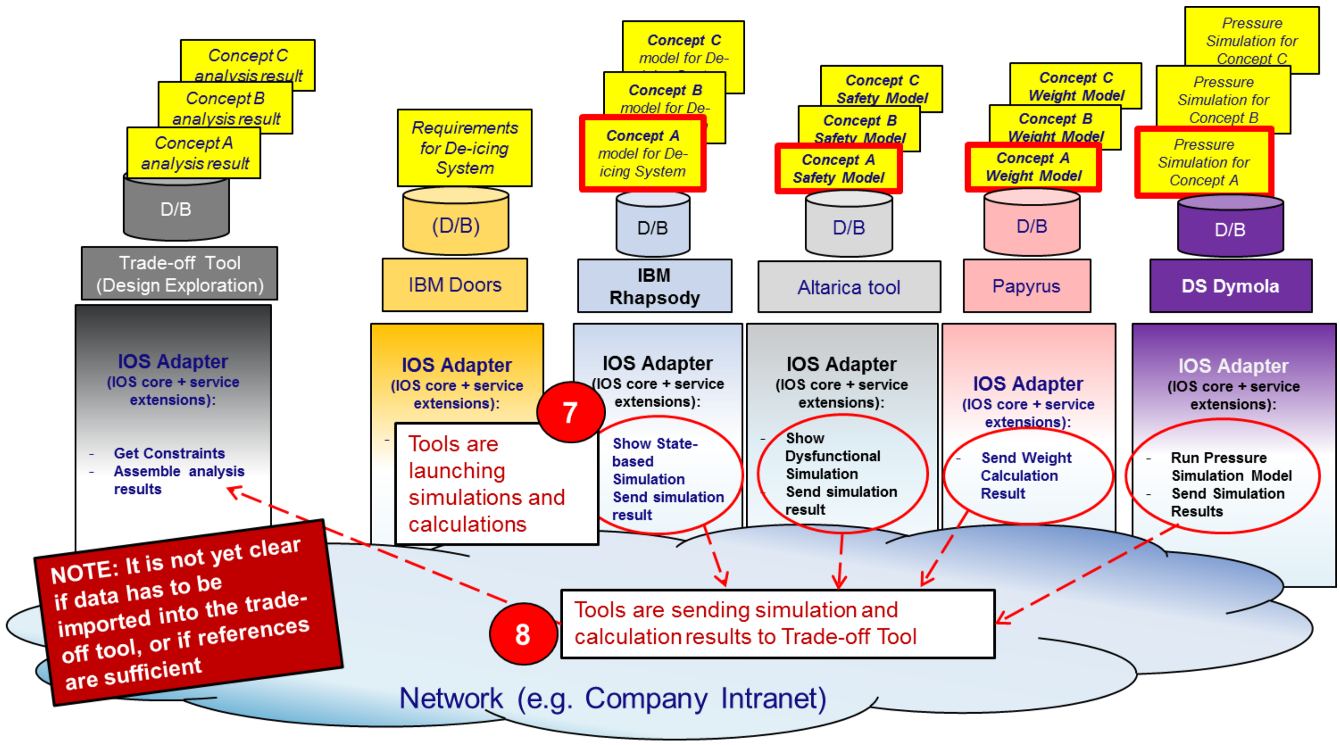


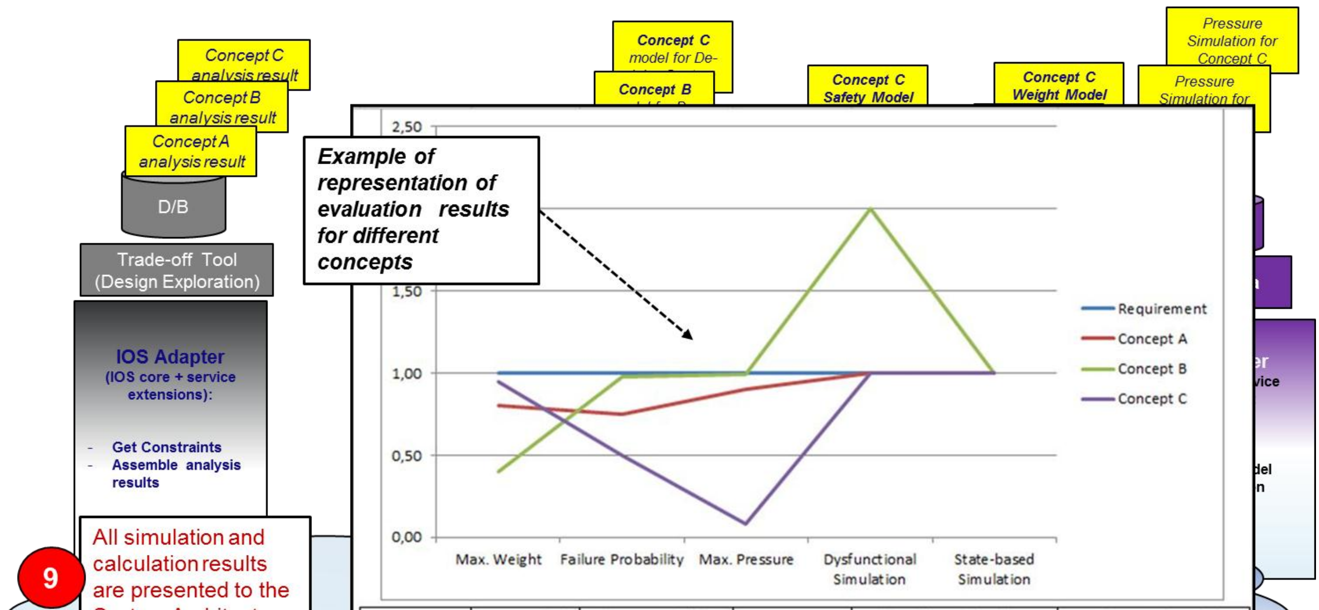
8.3 Method "Trade-off Analysis 001"




Concept C analysis result
Concept B analysis result
Concept A analysis result
D/B
Trade-off Tool (Design Exploration)
IOS Adapter (IOS core + service extensions):
- Get Constraints
- Assemble analysis results
Requirements for De-icing System
(D/B)
IBM Doors
IOS Adapter (IOS core + service extensions):
- Show Requirement
Concept C model for De-icing System
Concept B model for De-icing System
Concept A model for De-icing System
D/B
IBM Rhapsody
IOS Adapter (IOS core + service extensions):
- Show State-based Simulation
- Send simulation result
Concept C Safety Model
Concept B Safety Model
Concept A Safety Model
D/B
Altarica tool
IOS Adapter (IOS core + service extensions):
- Show Dysfunctional Simulation
- Send simulation result
Concept C Weight Model
Concept B Weight Model
Concept A Weight Model
D/B
Papyrus
IOS Adapter (IOS core + service extensions):
- Send Weight Calculation Result
Pressure Simulation for Concept C
Pressure Simulation for Concept B
Pressure Simulation for Concept A
D/B
DS Dymola
IOS Adapter (IOS core + service extensions):
- Run Pressure Simulation Model
- Send Simulation Results
2
System Architect launches service "Get Constraints"
The service should allow to filter constraints according to characteristics (as in EM "Analyse Requirements"
3
Request is transferred to DOORS
4
Constraints for the de-icing system are sent back (or shown)
Network (e.g. Company Intranet)


Concept C analysis result
Concept B analysis result
Concept A analysis result
D/B
Trade-off Tool (Design Exploration)
IOS Adapter (IOS core + service extensions):
- Get Constraints
- Assemble analysis results
Requirements for De-icing System
(D/B)
IBM Doors
IOS Adapter (IOS core + service extensions):
- Show Requirement
Concept C model for De-icing System
Concept B model for De-icing System
Concept A model for De-icing System
D/B
IBM Rhapsody
IOS Adapter (IOS core + service extensions):
- Show State-based Simulation
- Send simulation result
Concept C Safety Model
Concept B Safety Model
Concept A Safety Model
D/B
Altarica tool
IOS Adapter (IOS core + service extensions):
- Show Dysfunctional Simulation
- Send simulation result
Concept C Weight Model
Concept B Weight Model
Concept A Weight Model
D/B
Papyrus
IOS Adapter (IOS core + service extensions):
- Send Weight Calculation Result
Pressure Simulation for Concept C
Pressure Simulation for Concept B
Pressure Simulation for Concept A
D/B
DS Dymola
IOS Adapter (IOS core + service extensions):
- Run Pressure Simulation Model
- Send Simulation Results
5
System Architect launches request "Assemble analysis results"
6
Request is transferred to tools that are storing data that is relevant to assess the de-icing model "Concept A", "Concept B", and "Concept C"
Network (e.g. Company Intranet)

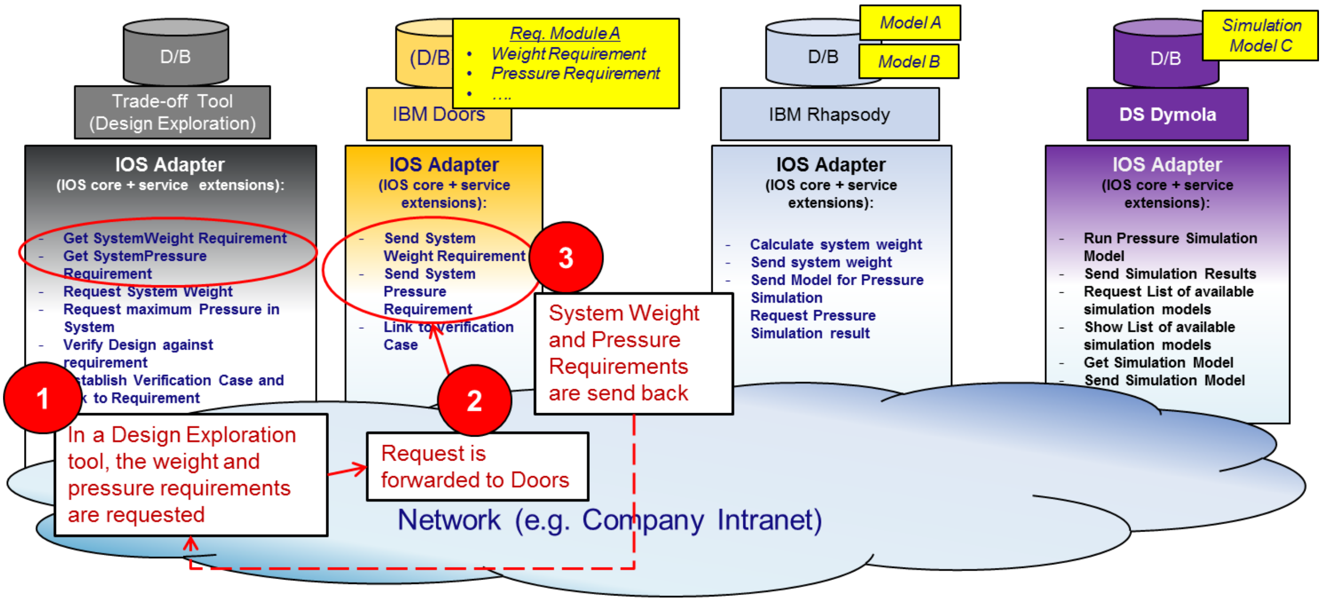


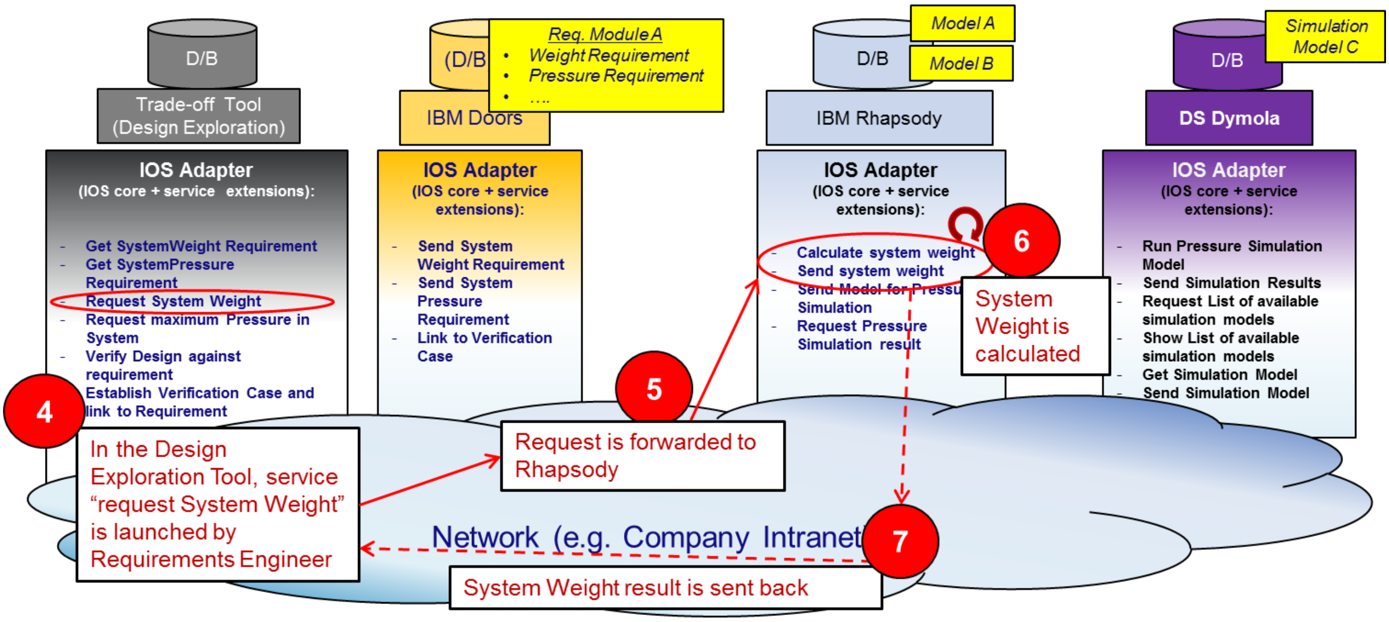


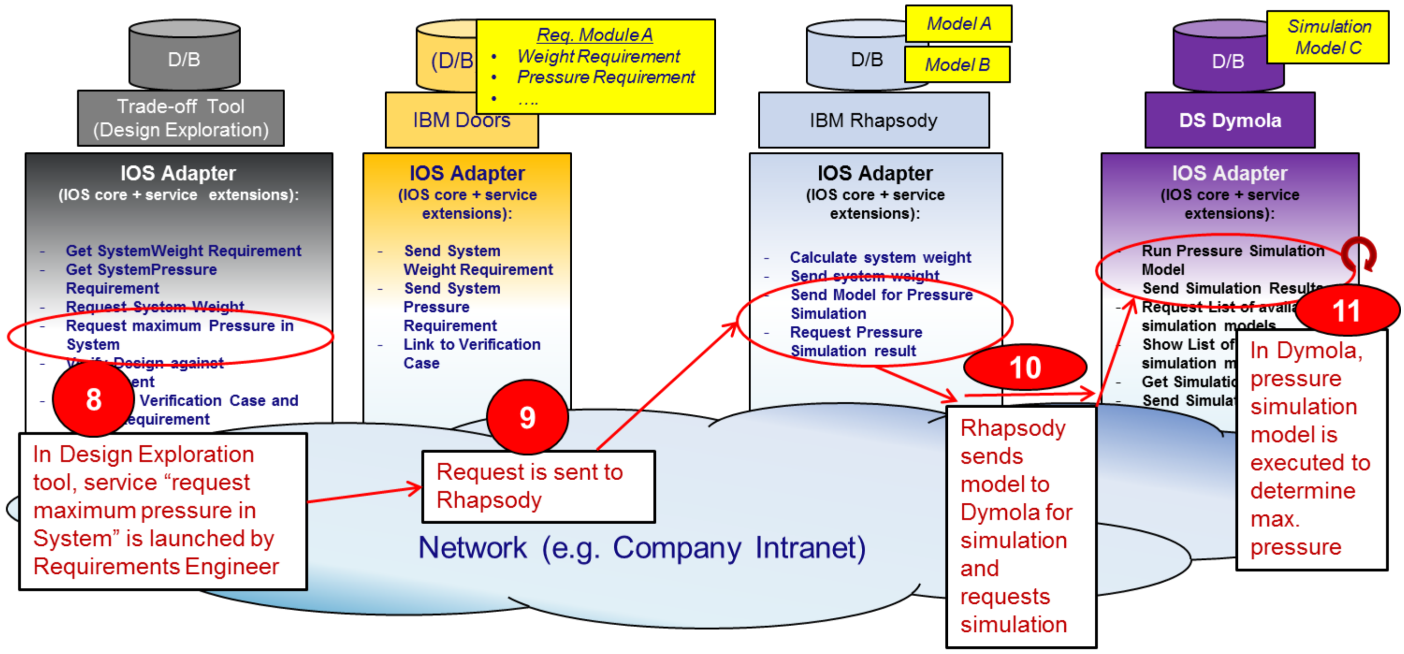
	Max. Weight	Failure Probability	Max. Pressure	Dysfunctional Simulation	State-based Simulation
Requirement	1000 kg	1.0 E-09	10 bar	Has to be OK	Has to be OK
Concept A	800 kg	7.5 E-10	9 bar	OK	OK
Concept B	400 kg	9.8 E-10	9.9 bar	NOK	OK
Concept C	950 kg	5 E-10	8 bar	OK	OK

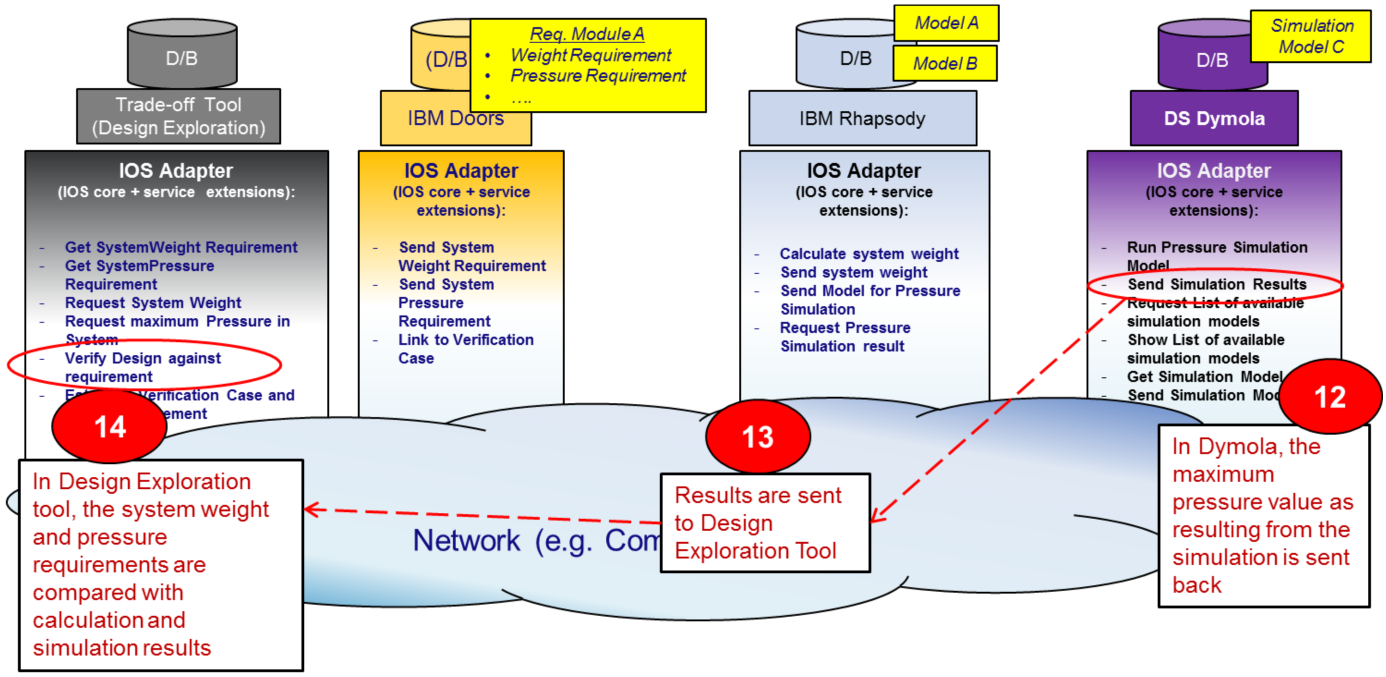
8.4 Engineering Method - “Verify Design against Requirements”

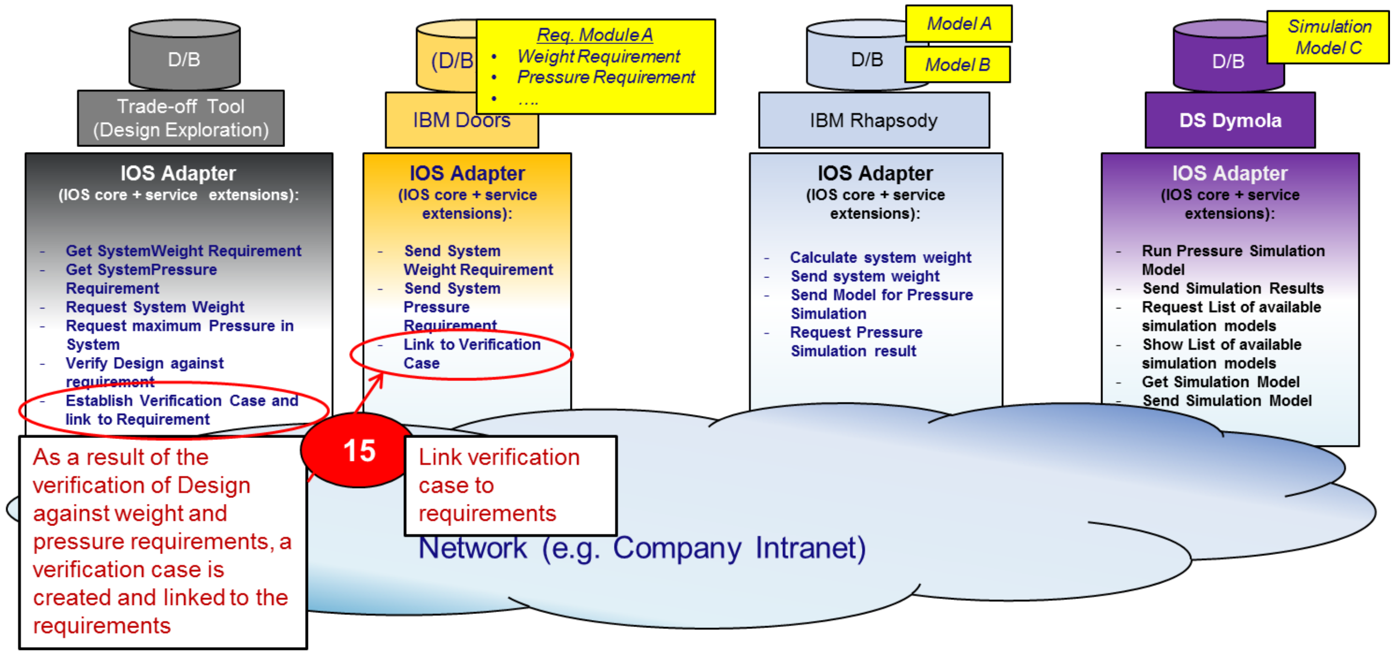
The following pictures provide an additional description of the Engineering Method “Analyse Requirements”





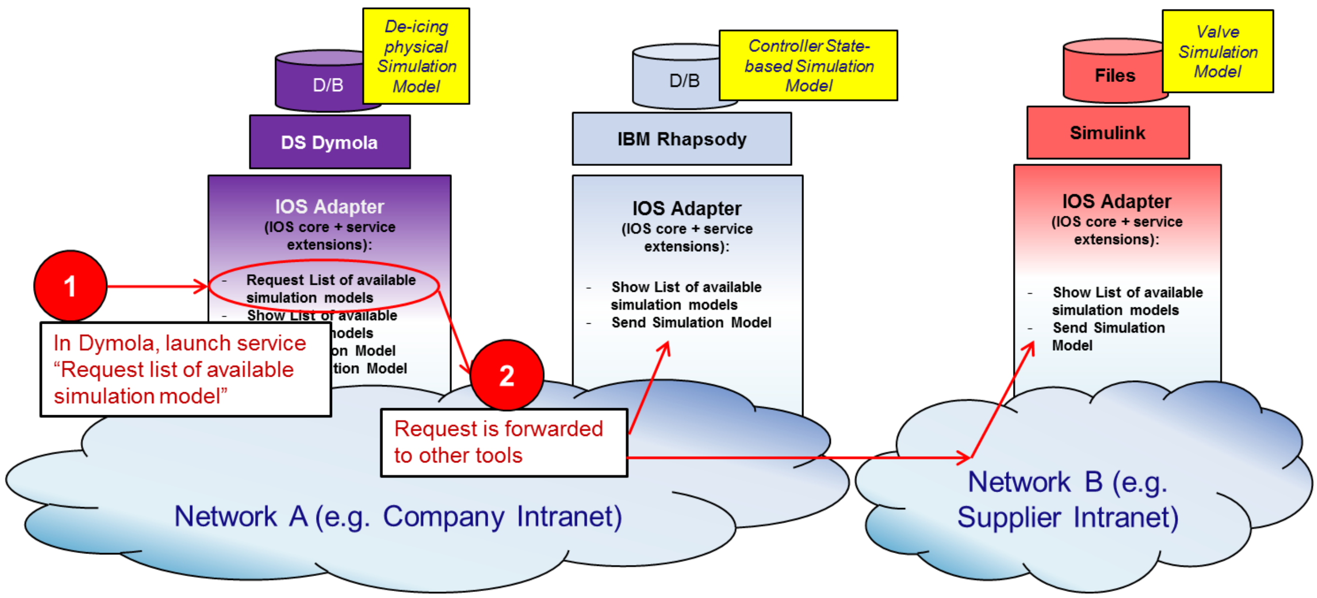

D/B
Trade-off Tool (Design Exploration)
IOS Adapter
(IOS core + service extensions):
Get SystemWeight Requirement
Get SystemPressure Requirement
Request maximum Pressure in System
Verification Case and
8
In Design Exploration tool, service “request maximum pressure in System” is launched by Requirements Engineer
Req. Module A
Weight Requirement
Pressure Requirement
….
IBM Doors
Send System Weight Requirement
Send System Pressure Requirement
Link to Verification Case
9
Request is sent to Rhapsody
Model A
Model B
IBM Rhapsody
Calculate system weight
Send Model for Pressure Simulation
Request Pressure Simulation result
10
Rhapsody sends model to Dymola for simulation and requests simulation
Simulation Model C
DS Dymola
Run Pressure Simulation Model
11
In Dymola, pressure simulation model is executed to determine max. pressure
Network (e.g. Company Intranet)

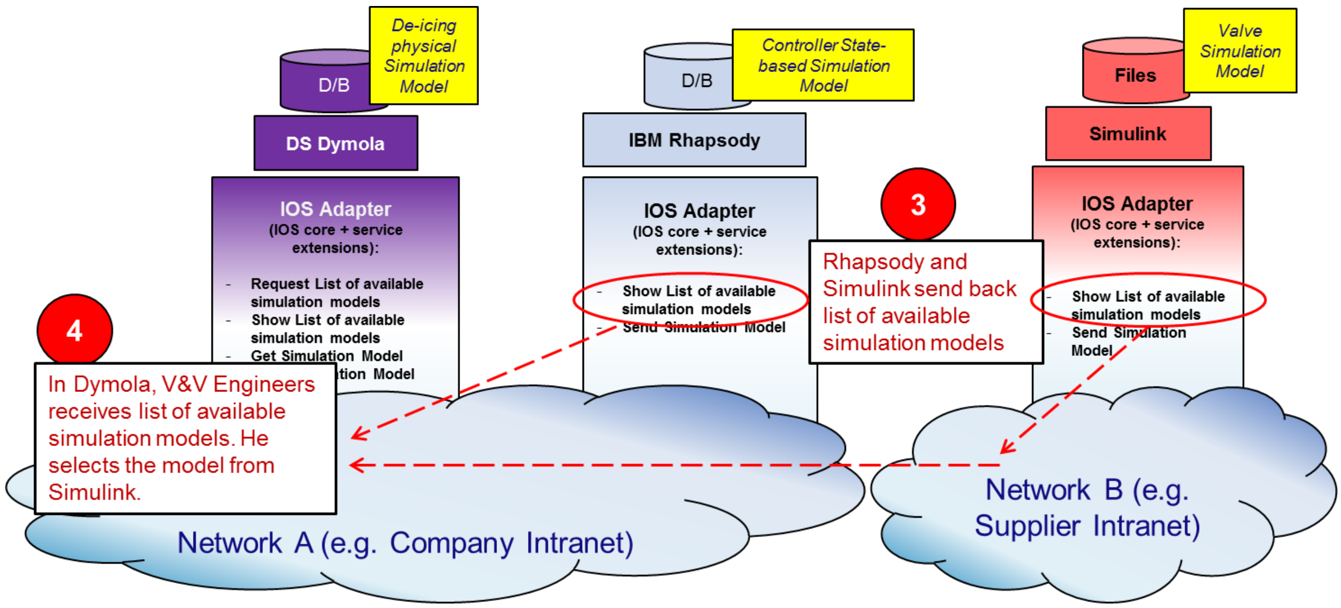

D/B
Trade-off Tool (Design Exploration)
IOS Adapter
(IOS core + service extensions):
Get SystemWeight Requirement
Get SystemPressure Requirement
Request System Weight
Request maximum Pressure in System
Verify Design against requirement
14
In Design Exploration tool, the system weight and pressure requirements are compared with calculation and simulation results
Req. Module A
Weight Requirement
Pressure Requirement
….
IBM Doors
Send System Weight Requirement
Send System Pressure Requirement
Link to Verification Case
Model A
Model B
IBM Rhapsody
Calculate system weight
Send system weight
Send Model for Pressure Simulation
Request Pressure Simulation result
13
Results are sent to Design Exploration Tool
Simulation Model C
DS Dymola
Run Pressure Simulation Model
Send Simulation Results
simulation models
Show List of available simulation models
12
In Dymola, the maximum pressure value as resulting from the simulation is sent back

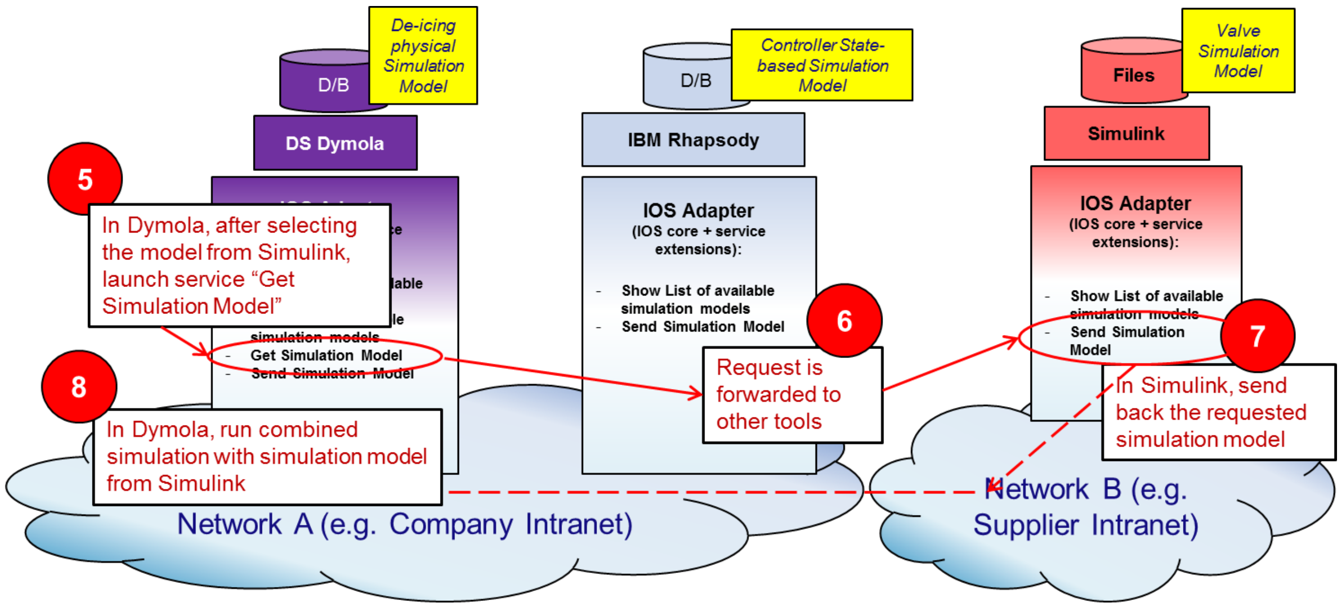


8.5 Engineering Method - "Heterogenous Simulation"

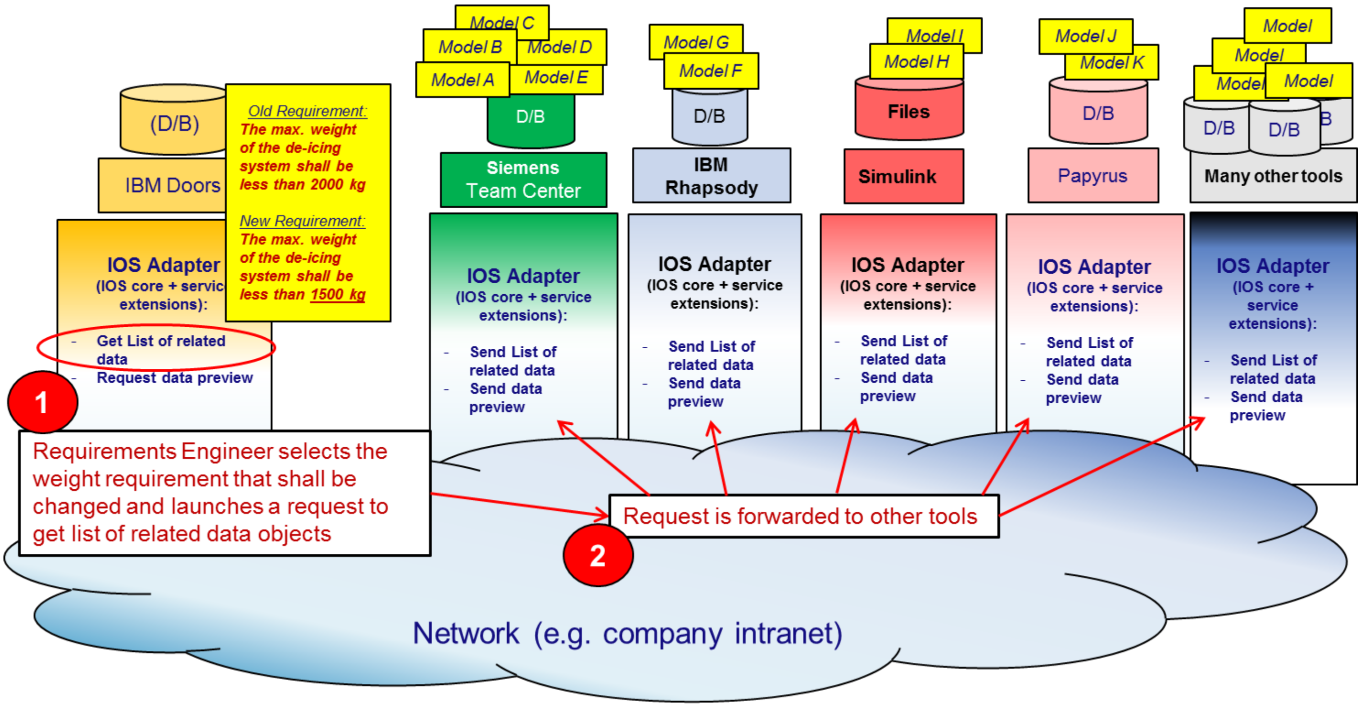
The following pictures provide an additional description of the Engineering Method "Analyse Requirements"

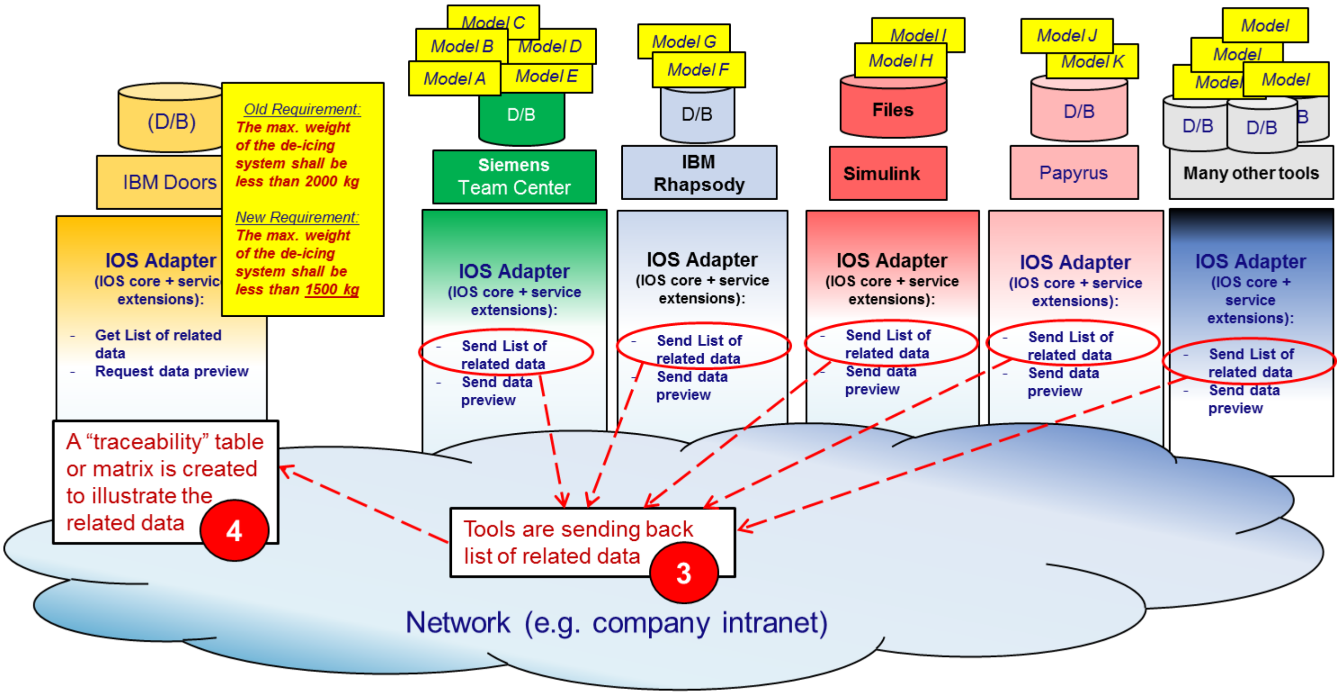


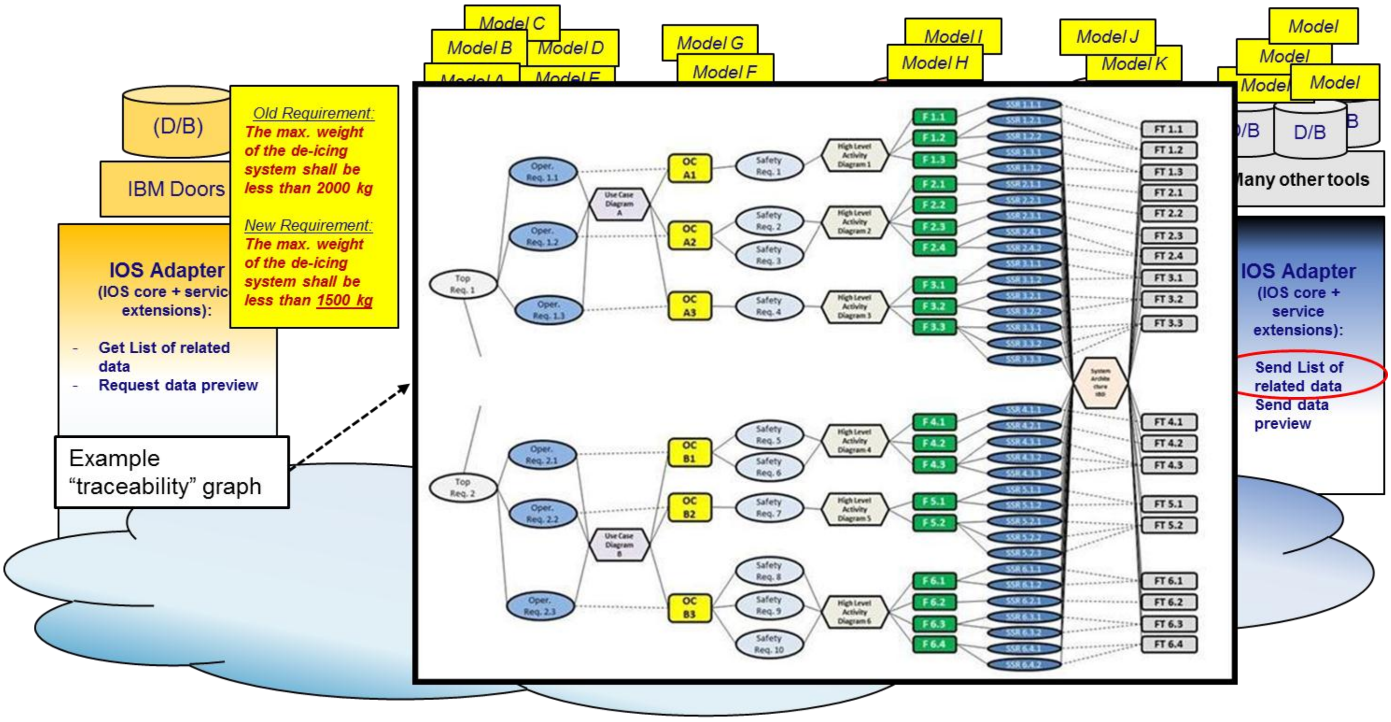

De-icing physical Simulation Model
D/B
DS Dymola
IOS Adapter
(IOS core + service extensions):
- Request List of available simulation models
- Show List of available simulation models
- Get Simulation Model
Controller State-based Simulation Model
D/B
IBM Rhapsody
IOS Adapter
(IOS core + service extensions):
- Show List of available simulation models
- Send Simulation Model
Files
Valve Simulation Model
Simulink
IOS Adapter
(IOS core + service extensions):
- Show List of available simulation models
- Send Simulation Model
3
Rhapsody and Simulink send back list of available simulation models
4
In Dymola, V&V Engineers receives list of available simulation models. He selects the model from Simulink.
Network A (e.g. Company Intranet)
Network B (e.g. Supplier Intranet)

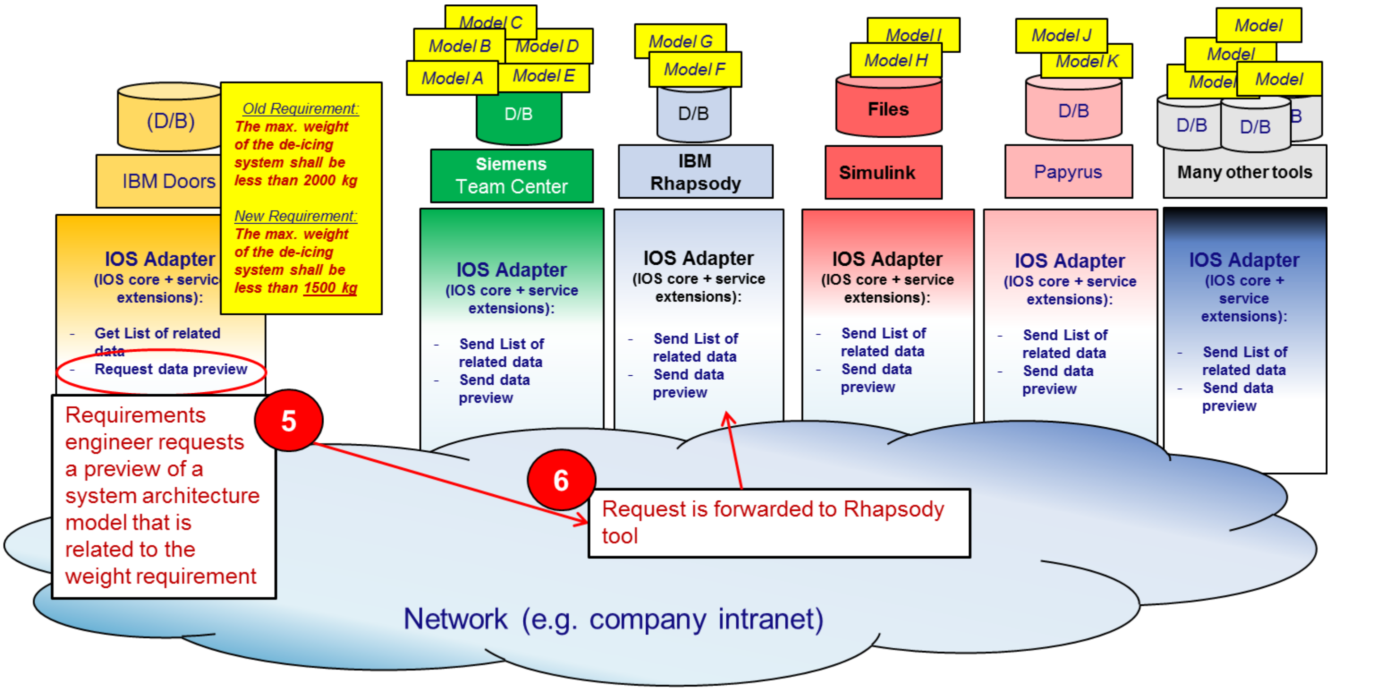

De-icing physical Simulation Model
D/B
DS Dymola
5
In Dymola, after selecting the model from Simulink, launch service "Get Simulation Model"
simulation models
- Get Simulation Model
- Send Simulation Model
Controller State-based Simulation Model
D/B
IBM Rhapsody
IOS Adapter
(IOS core + service extensions):
- Show List of available simulation models
- Send Simulation Model
6
Request is forwarded to other tools
Files
Valve Simulation Model
Simulink
IOS Adapter
(IOS core + service extensions):
- Show List of available simulation models
- Send Simulation Model
7
In Simulink, send back the requested simulation model
8
In Dymola, run combined simulation with simulation model from Simulink
Network A (e.g. Company Intranet)
Network B (e.g. Supplier Intranet)

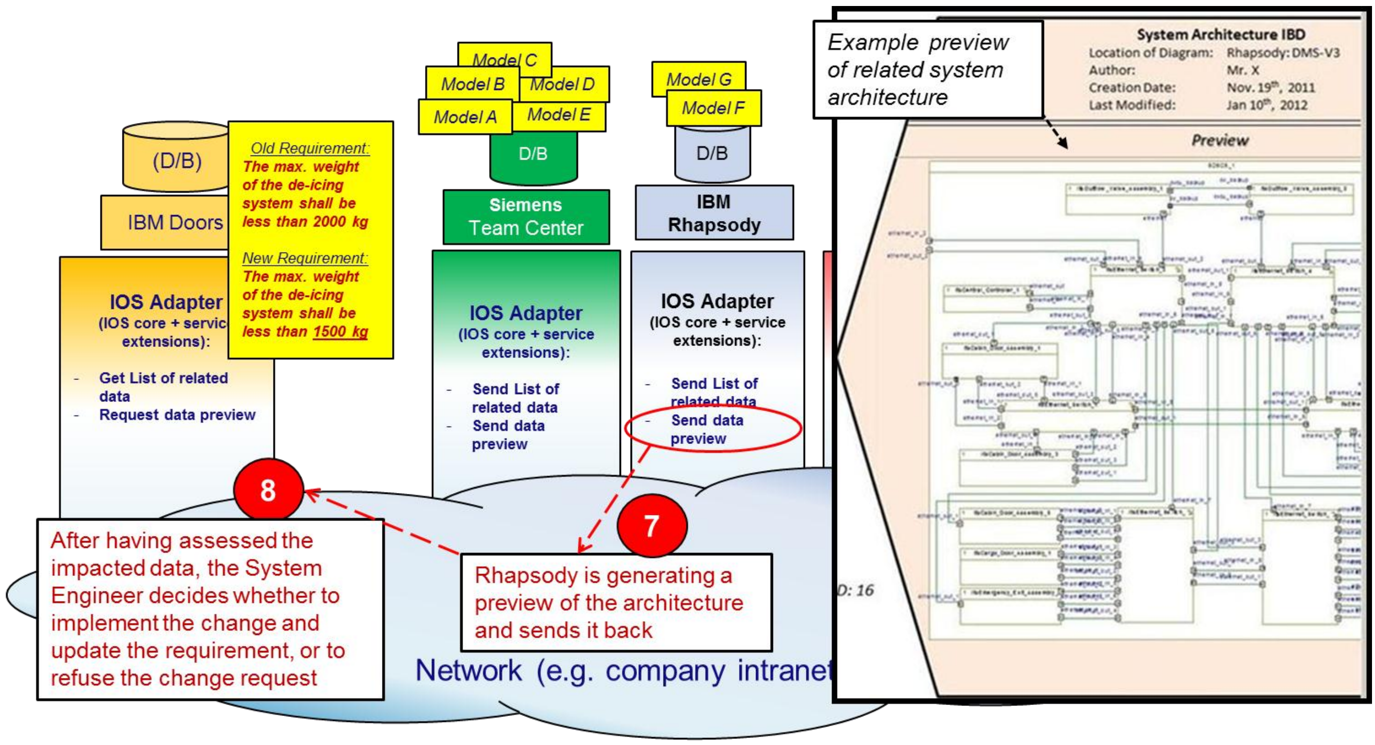
8.6 Method "Change Impact Analysis"





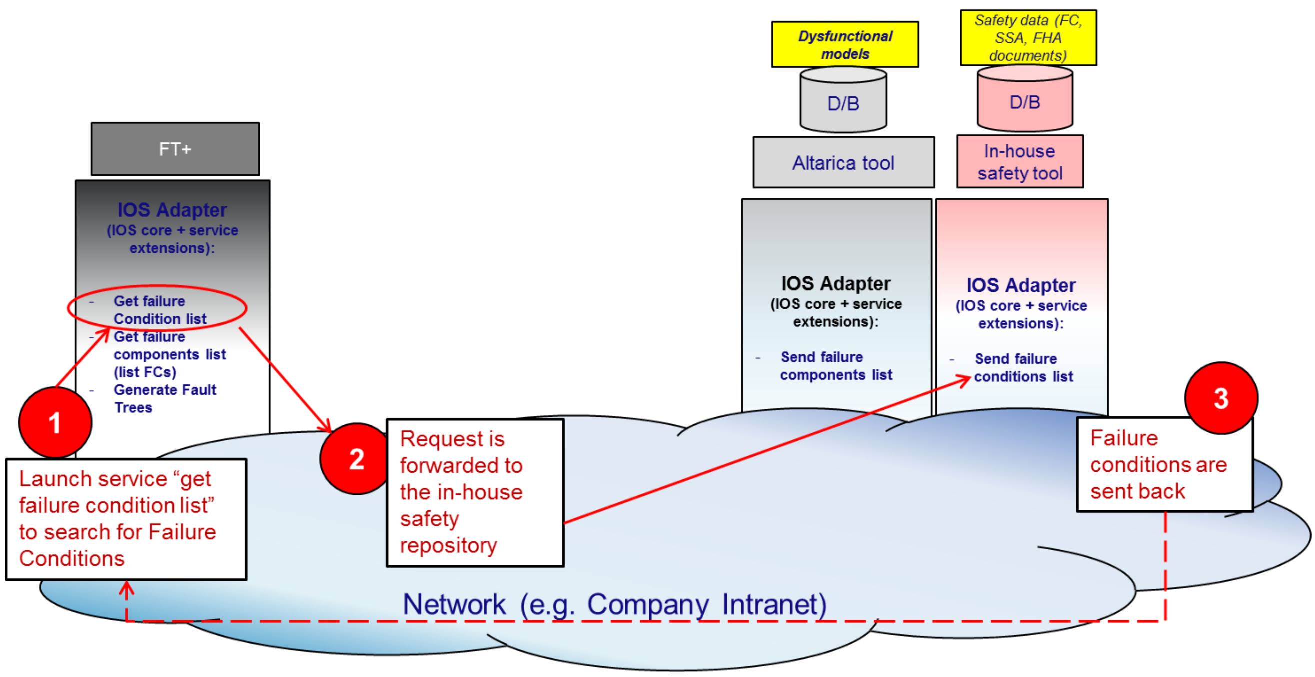

Model C
Model B
Model D
Model A
Model E
Model G
Model F
Model I
Model H
Model J
Model K
Model
(D/B)
IBM Doors
Old Requirement:
The max. weight of the de-icing system shall be less than 2000 kg
New Requirement:
The max. weight of the de-icing system shall be less than 1500 kg
IOS Adapter
(IOS core + servic extensions):
Get List of related data
Request data preview
Example “traceability” graph
Many other tools
IOS Adapter
(IOS core + service extensions):
Send List of related data
Send data preview

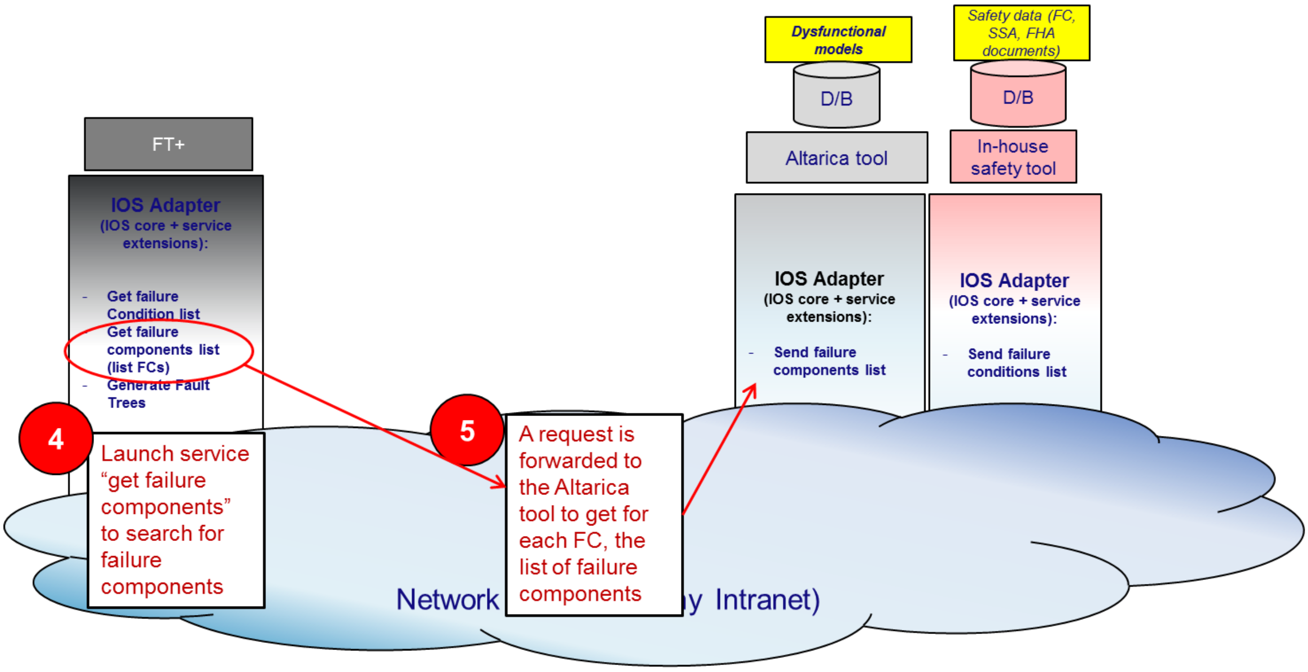

Model C
Model B
Model D
Model A
Model E
D/B
Siemens Team Center
Model G
Model F
D/B
IBM Rhapsody
Model I
Model H
Files
Simulink
Model J
Model K
D/B
Papyrus
Model
D/B
Many other tools
(D/B)
IBM Doors
Old Requirement:
The max. weight of the de-icing system shall be less than 2000 kg
New Requirement:
The max. weight of the de-icing system shall be less than 1500 kg
IOS Adapter
(IOS core + servic extensions):
Get List of related data
Request data preview
IOS Adapter
(IOS core + service extensions):
Send List of related data
Send data preview
Requirements engineer requests a preview of a system architecture model that is related to the weight requirement
5
6
Request is forwarded to Rhapsody tool
Network (e.g. company intranet)



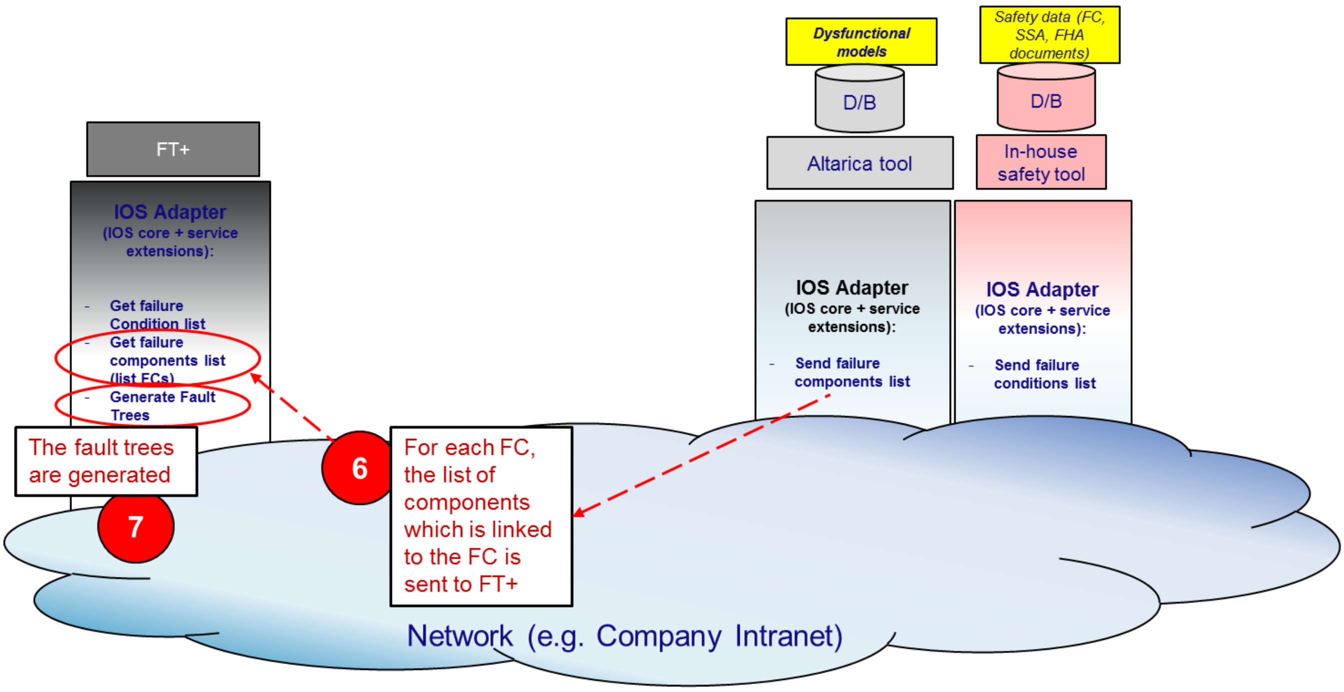
8.7 Engineering Method - "Fault-tree Generation"

The following pictures provide an additional description of the Engineering Method "Analyse Requirements"



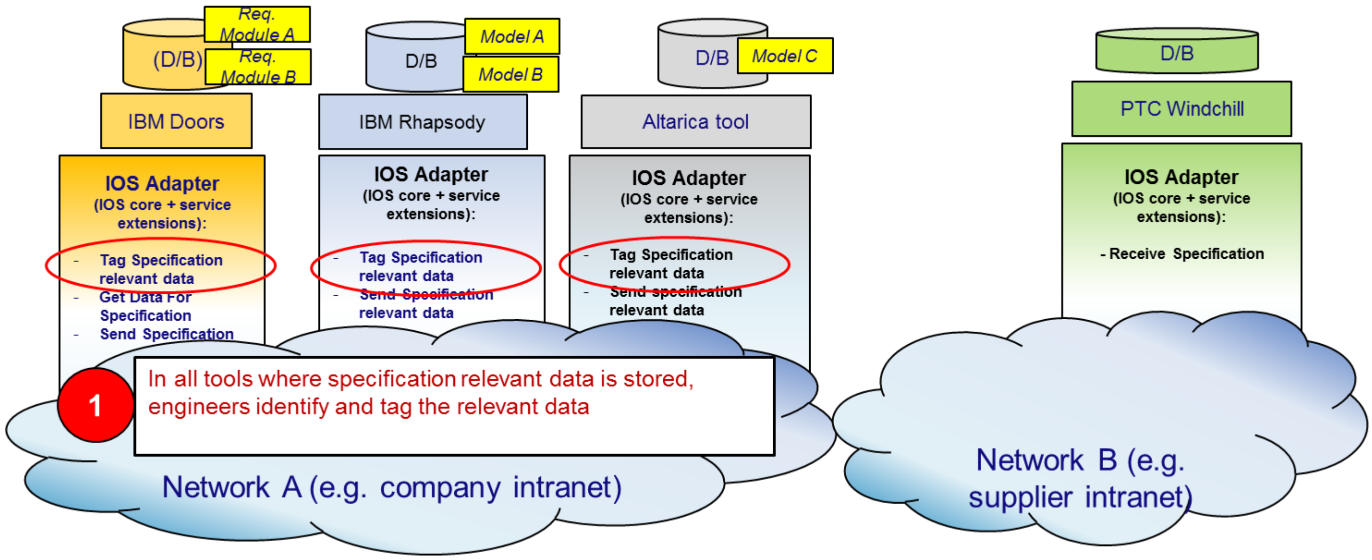


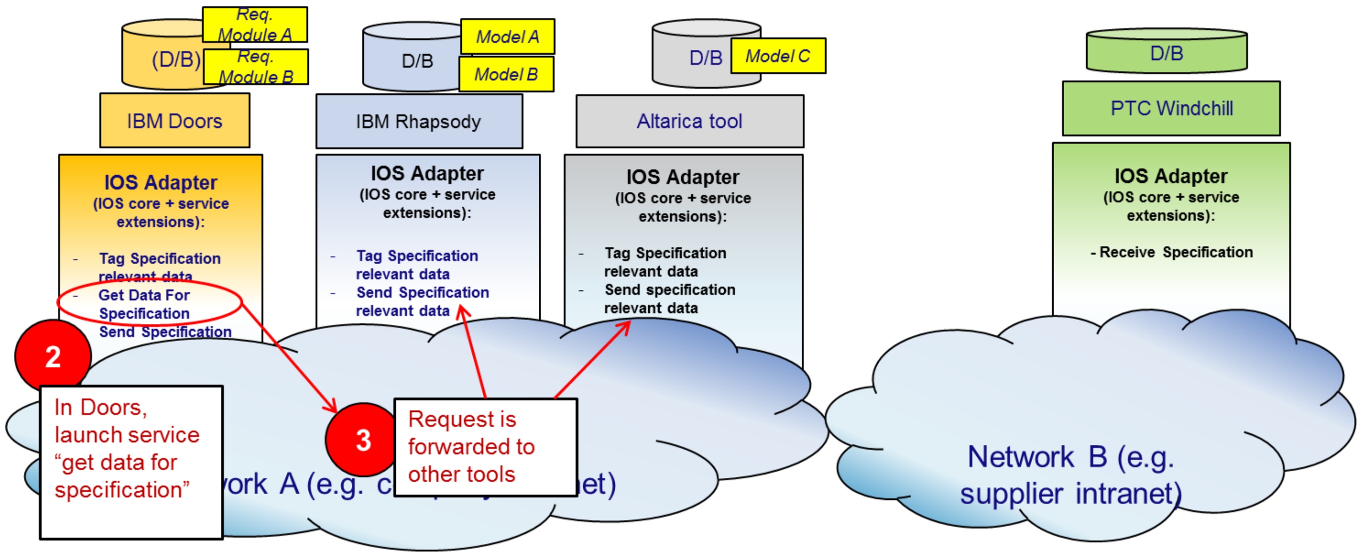
Dysfunctional models
Safety data (FC, SSA, FHA documents)
D/B
D/B
Altarica tool
In-house safety tool
FT+
IOS Adapter
(IOS core + service extensions):
- Get failure Condition list
- Get failure components list (list FCs)
- Generate Fault Trees
IOS Adapter
(IOS core + service extensions):
- Send failure components list
IOS Adapter
(IOS core + service extensions):
- Send failure conditions list
4
Launch service "get failure components" to search for failure components
5
A request is forwarded to the Altarica tool to get for each FC, the list of failure components
Network
y Intranet)

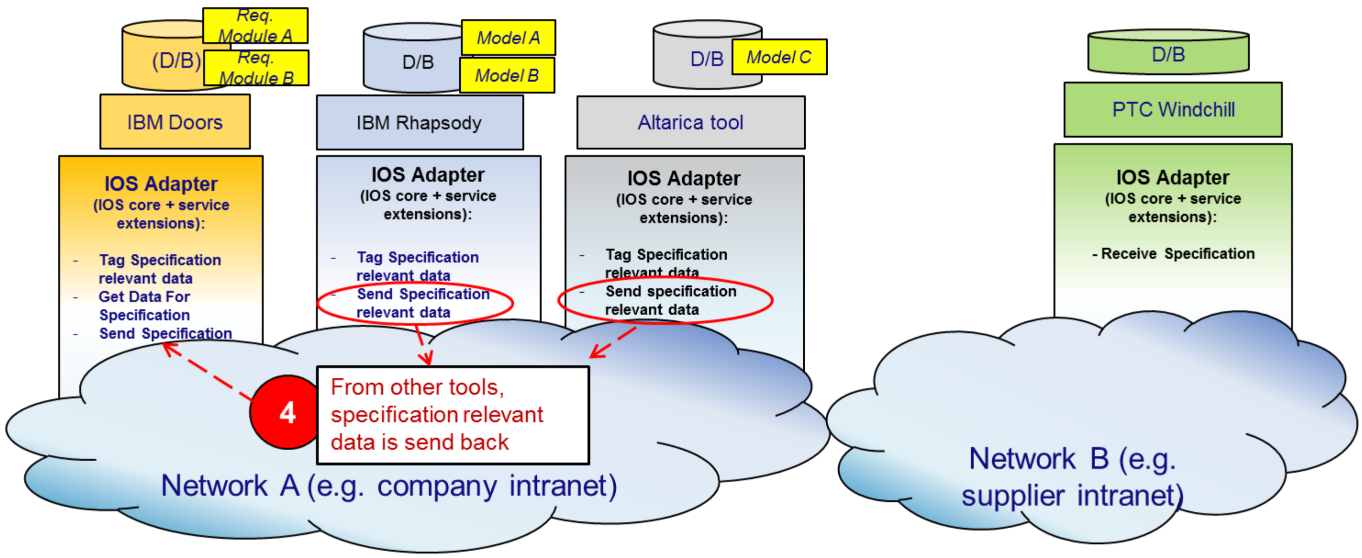


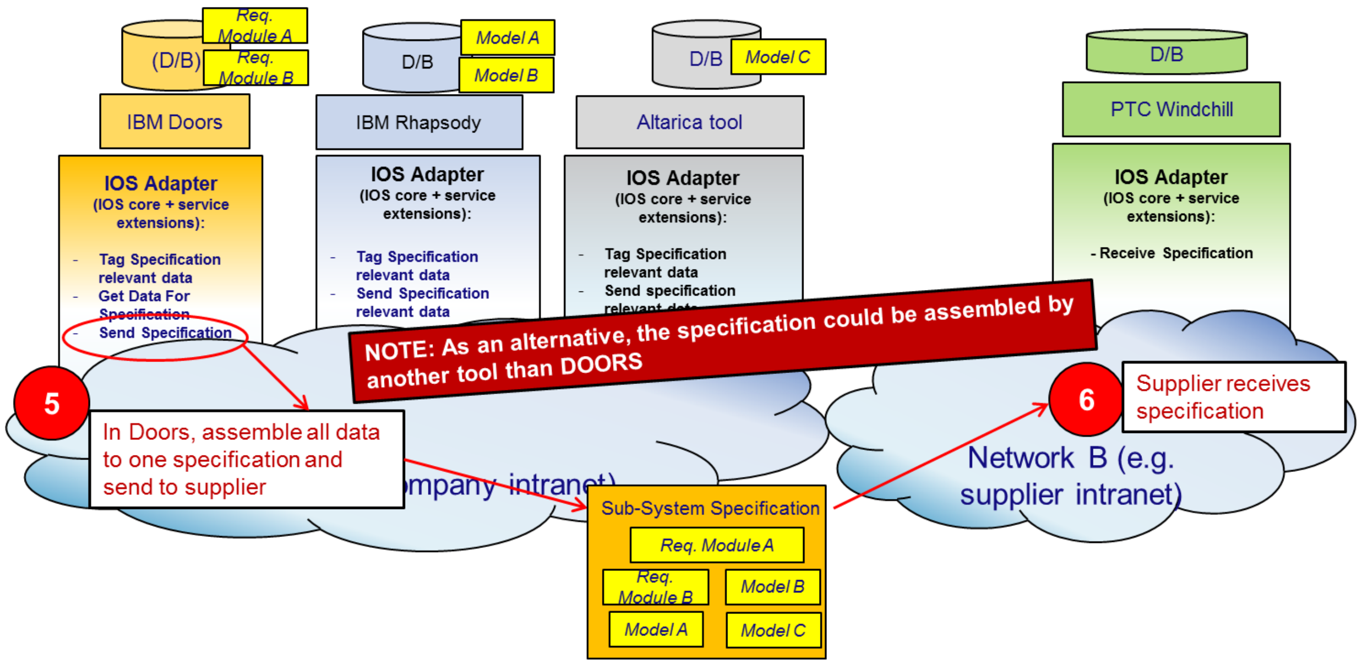
Dysfunctional models
Safety data (FC, SSA, FHA documents)
D/B
D/B
Altarica tool
In-house safety tool
FT+
IOS Adapter
(IOS core + service extensions):
- Get failure Condition list
- Get failure components list (list FCs)
- Generate Fault Trees
IOS Adapter
(IOS core + service extensions):
- Send failure components list
IOS Adapter
(IOS core + service extensions):
- Send failure conditions list
The fault trees are generated
7
6
For each FC, the list of components which is linked to the FC is sent to FT+
Network (e.g. Company Intranet)

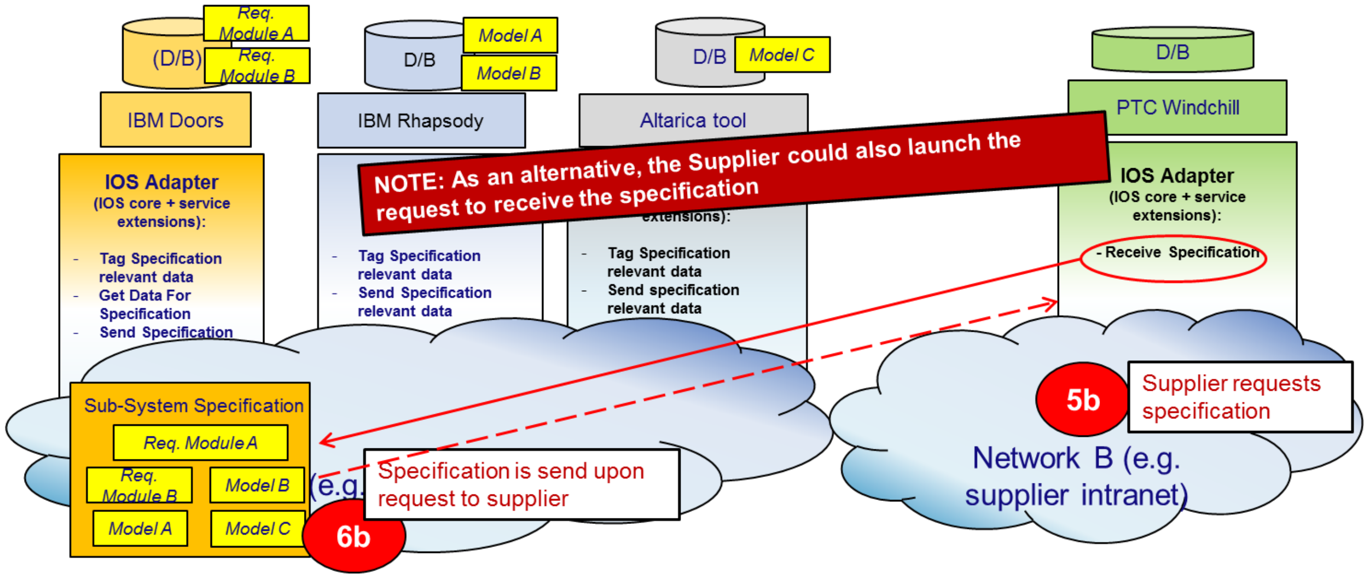
8.8 Method "Provide Specification"






Req. Module A
(D/B)
Req. Module B
IBM Doors
IOS Adapter
(IOS core + service extensions):
- Tag Specification relevant data
- Get Data For Specification
- Send Specification
D/B
Model A
Model B
IBM Rhapsody
IOS Adapter
(IOS core + service extensions):
- Tag Specification relevant data
- Send Specification relevant data
D/B
Model C
Altarica tool
IOS Adapter
(IOS core + service extensions):
- Tag Specification relevant data
- Send specification relevant data
D/B
PTC Windchill
IOS Adapter
(IOS core + service extensions):
- Receive Specification
4
From other tools, specification relevant data is send back
Network A (e.g. company intranet)
Network B (e.g. supplier intranet)

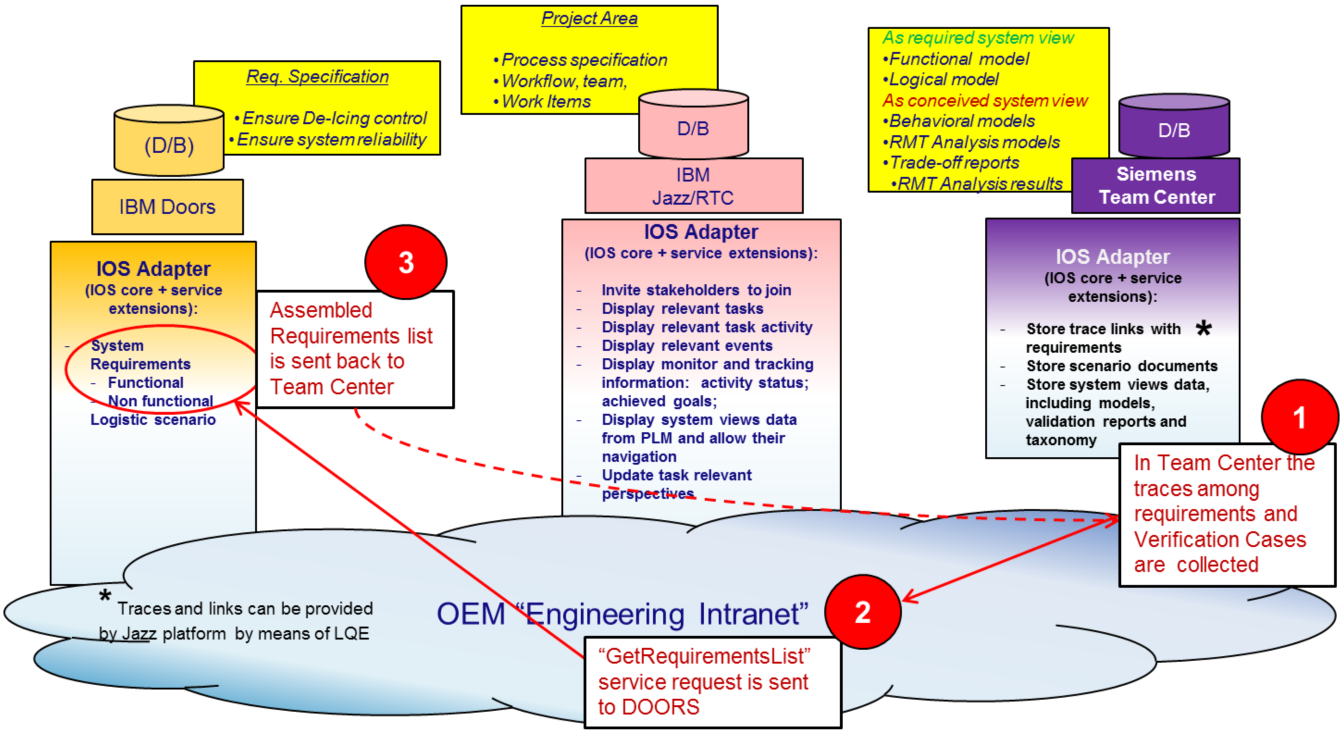

Req. Module A
(D/B)
Req. Module B
IBM Doors
IOS Adapter
(IOS core + service extensions):
- Tag Specification relevant data
- Get Data For Specification
- Send Specification
D/B
Model A
Model B
IBM Rhapsody
IOS Adapter
(IOS core + service extensions):
- Tag Specification relevant data
- Send Specification relevant data
D/B
Model C
Altarica tool
IOS Adapter
(IOS core + service extensions):
- Tag Specification relevant data
- Send specification relevant data
D/B
PTC Windchill
IOS Adapter
(IOS core + service extensions):
- Receive Specification
NOTE: As an alternative, the specification could be assembled by another tool than DOORS
5
In Doors, assemble all data to one specification and send to supplier
6
Supplier receives specification
Network B (e.g. supplier intranet)
Sub-System Specification
Req. Module A
Req. Module B
Model B
Model A
Model C

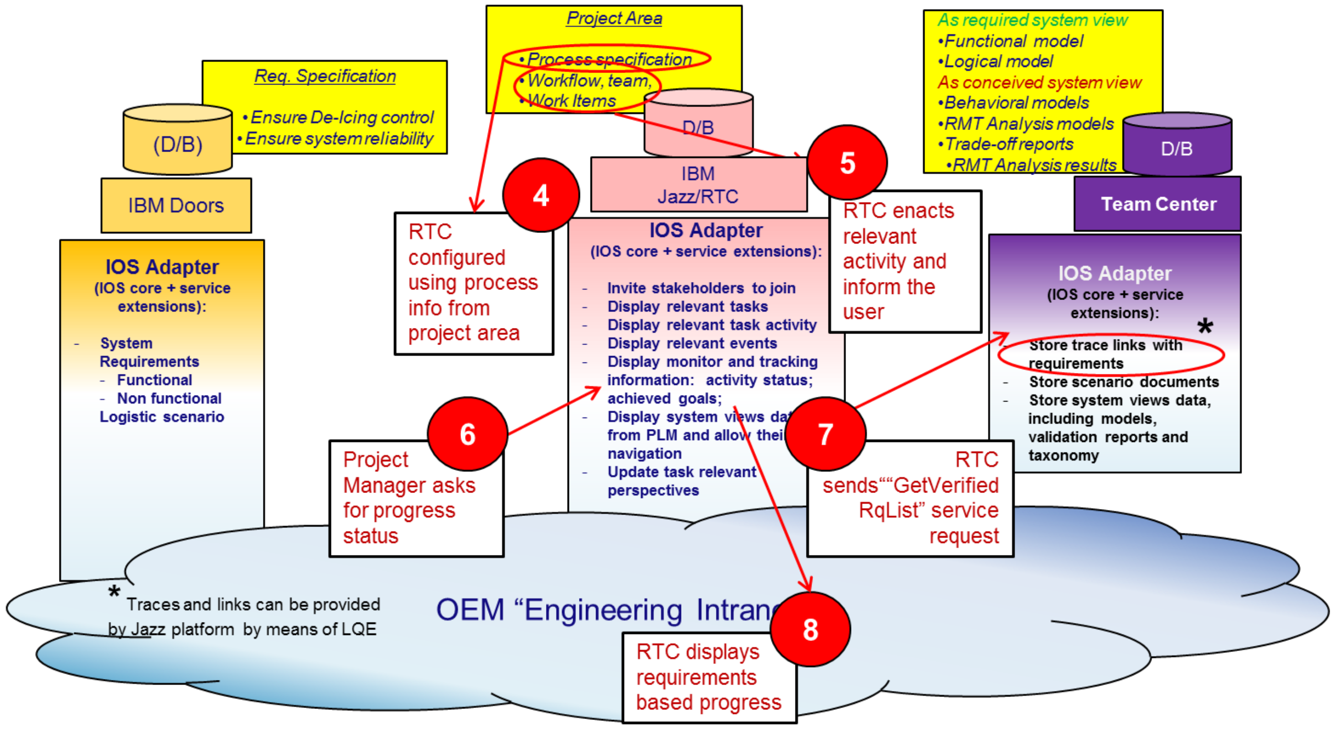


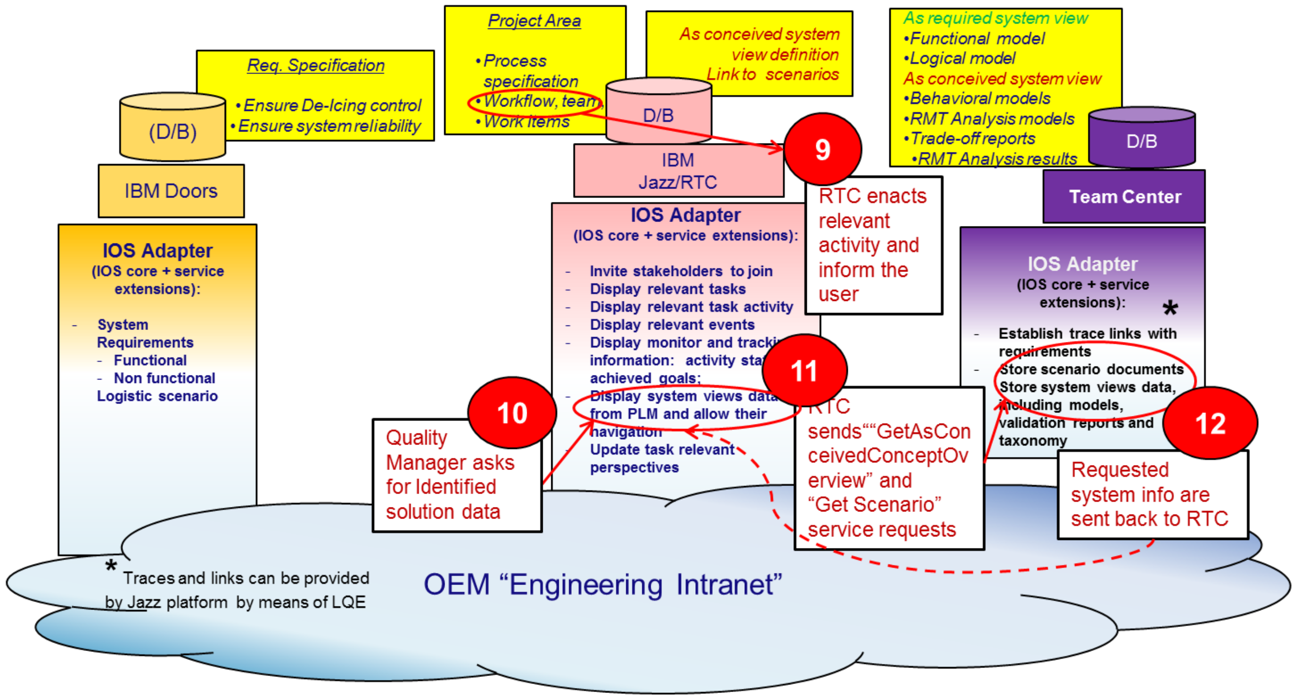
Req. Module A
(D/B)
Req. Module B
IBM Doors
D/B
Model A
Model B
IBM Rhapsody
D/B
Model C
Altarica tool
D/B
PTC Windchill
NOTE: As an alternative, the Supplier could also launch the request to receive the specification
IOS Adapter
(IOS core + service extensions):
- Tag Specification relevant data
- Get Data For Specification
- Send Specification
- Tag Specification relevant data
- Send Specification relevant data
- Tag Specification relevant data
- Send specification relevant data
IOS Adapter
(IOS core + service extensions):
- Receive Specification
Sub-System Specification
Req. Module A
Req. Module B
Model B
Model A
Model C
Specification is send upon request to supplier
6b
5b
Supplier requests specification
Network B (e.g. supplier intranet)

8.9 Engineering Method - "Provide Process Management"

The following pictures provide a graphical description of method ProvideProcessManagement instantiated on a specific task that involves Project and Quality managers. It also provides details about the envisaged tools.

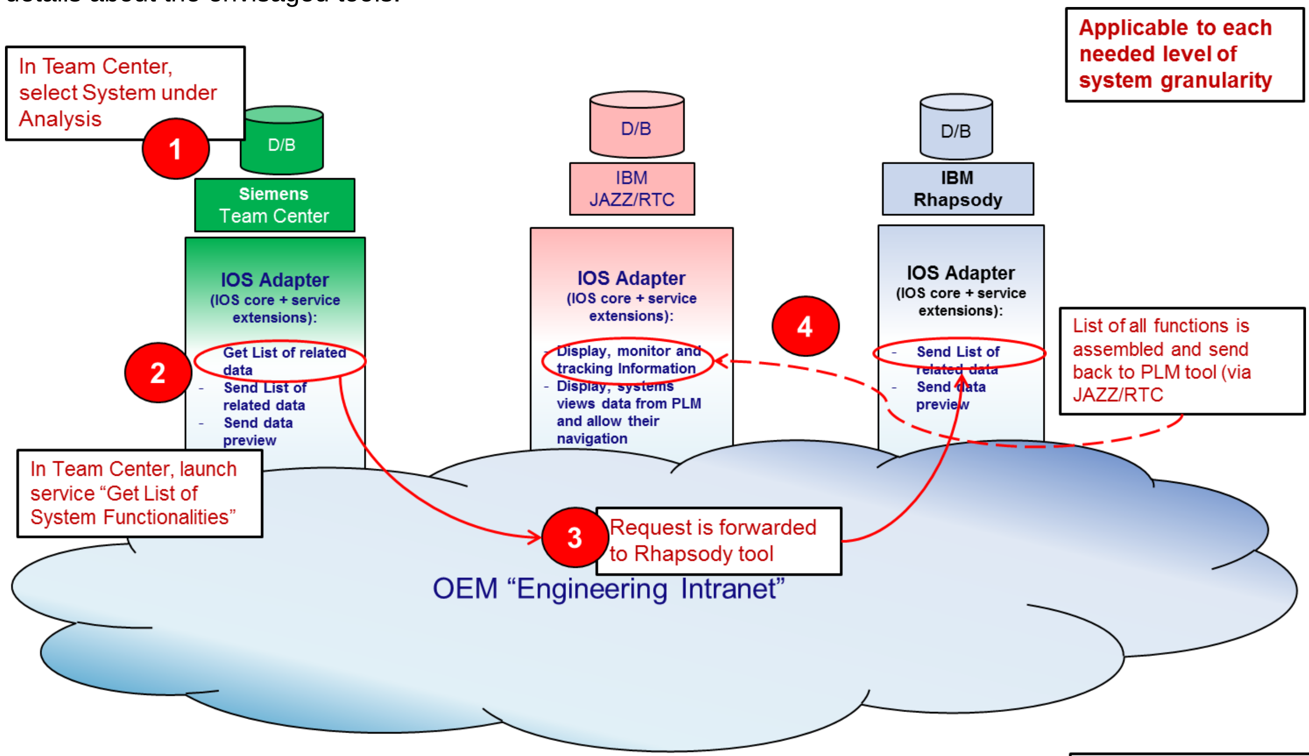


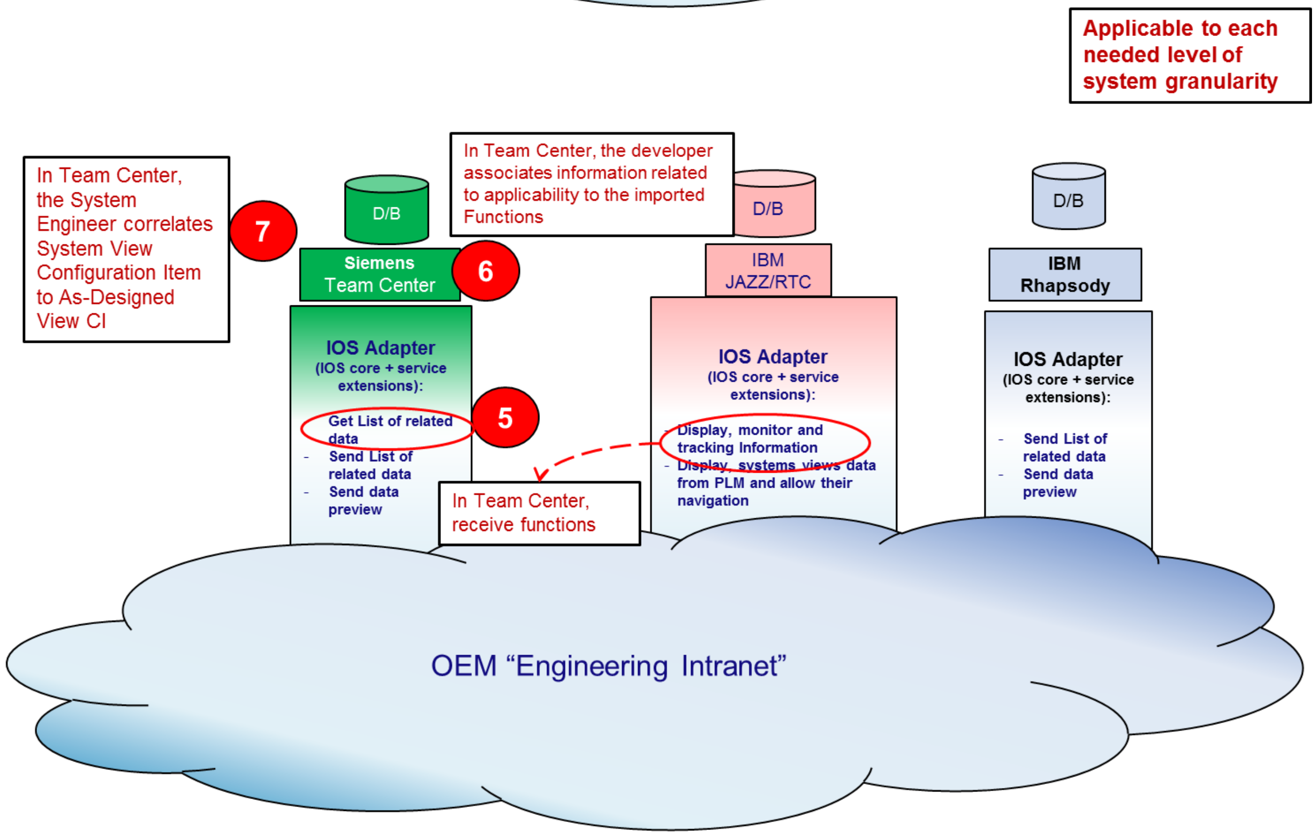

Project Area
•Process specification
•Workflow, team,
•Work Items
Req. Specification
•Ensure De-Icing control
•Ensure system reliability
(D/B)
IBM Doors
IOS Adapter
(IOS core + service extensions):
- System Requirements
- Functional
- Non functional
Logistic scenario
D/B
IBM Jazz/RTC
4
RTC configured using process info from project area
IOS Adapter
(IOS core + service extensions):
- Invite stakeholders to join
- Display relevant tasks
- Display relevant task activity
- Display relevant events
- Display monitor and tracking information: activity status; achieved goals;
- Display system views data from PLM and allow their navigation
- Update task relevant perspectives
5
RTC enacts relevant activity and inform the user
As required system view
•Functional model
•Logical model
As conceived system view
•Behavioral models
•RMT Analysis models
•Trade-off reports
•RMT Analysis results
D/B
Team Center
IOS Adapter
(IOS core + service extensions):
*
Store trace links with requirements
- Store scenario documents
- Store system views data, including models, validation reports and taxonomy
6
Project Manager asks for progress status
7
RTC sends““GetVerified RqList” service request
* Traces and links can be provided by Jazz platform by means of LQE
OEM “Engineering Intranet”
8
RTC displays requirements based progress


Project Area
•Process specification
•Workflow, team
•Work Items
As conceived system view definition
Link to scenarios
Req. Specification
•Ensure De-Icing control
•Ensure system reliability
(D/B)
IBM Doors
IOS Adapter
(IOS core + service extensions):
- System Requirements
- Functional
- Non functional
Logistic scenario
D/B
IBM Jazz/RTC
IOS Adapter
(IOS core + service extensions):
- Invite stakeholders to join
- Display relevant tasks
- Display relevant task activity
- Display relevant events
- Display monitor and tracking information: activity status; achieved goals;
- Display system views data from PLM and allow their navigation
- Update task relevant perspectives
9
RTC enacts relevant activity and inform the user
As required system view
•Functional model
•Logical model
As conceived system view
•Behavioral models
•RMT Analysis models
•Trade-off reports
•RMT Analysis results
D/B
Team Center
IOS Adapter
(IOS core + service extensions):
*
- Establish trace links with requirements
- Store scenario documents
Store system views data, including models, validation reports and taxonomy
10
Quality Manager asks for Identified solution data
11
RTC sends““GetAsConceivedConceptOverview” and “Get Scenario” service requests
12
Requested system info are sent back to RTC
* Traces and links can be provided by Jazz platform by means of LQE
OEM “Engineering Intranet”

8.10 Engineering Method - “Put all data under configuration control”

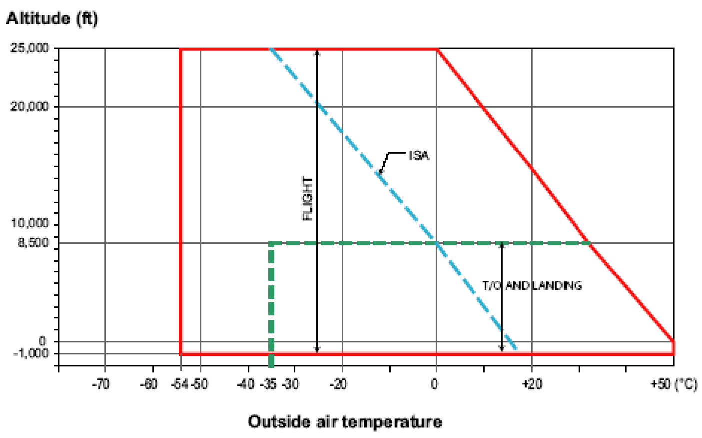
The following pictures provide a graphical description of method “Put all data under configuration control” instantiated on a specific task that involves Systems Engineers and Configuration Managers. It also provides details about the envisaged tools.





8.11 Top Level Requirements for the De-icing System

20	**1 Scope**
21	**1.1 Identification and Purpose**
22	The Specification defines the technical performance and requirements that shall be satisfied by the technical solution of the Ice Protection subsystem.
23	**1.2 Definition**
24	The word SHALL in the text expresses a mandatory requirement of the Specification. Departure from such a requirement is not permissible without a formal agreement between the Supplier and Purchaser.
25	**1.3 Operational Requirements**
26	The Ice Protection Sub-System shall permit aircraft operation without restriction in icing condition through "continous maximum" and "intermittent maximum" asd specified by FAR/JAR 25.
27	The Ice Protection Sub-System shall permit aircraft operation without restriction in icing condition within One Engine Inoperative condition.
28	The system shall provide ice condition information (visual) to the Flight Crew.
29	The system shall provide ice thickness information (visual) to the Flight Crew.
30	The system shall be operated (activated and de-activated) and set up (fast / slow cycle modes) by means of a dedicated control panel located in the cockpit.
31	The system shall support the Flight Crew by providing the activation request message for each operating modes.
32	The system shall provide status data (on/off) to the Flight Crew.
33	The system shall provide operating mode data (fast / slow) to the Flight Crew.
34	The system shall provide warning message (aural / visual) to the Flight Crew in case of failure event and mal-functioning.
35	Slow and fast de-icing cycles shall be provided by the stand-by control system.
36	Fast de-icing cycles shall be provided by the stand-by control system.
37	The ice protection system shall be operated under all flight phases from take off to landing when is protection is required and taking into account the following temperature envelope to which the AC is exposed:



40 The airframe ice protection system shall be operated under all flight phases from take off to landing when ice protection is required, and taking into account the following A/C performance:

- Climb rate (@SL) = 1500 ft/min
- Time to climb (@FL 200) = 13 min
- Climb speed = 170 CAS
- Operating Altitude = 20.000 ft
- Max Cruise Altitude = 25.000 ft
- Max Cruise Speed (@20.000 ft) = 320 KTS
- Max Cruise Speed (@25.000 ft) = 315 KTS
- Typical Mission Range = 1000 nm
- Cruise leg distance = 900 nm
- Cruise leg time (@ 20.000 ft) = 2.8 h
- Descent Speed = 200 - 250 CAS
- Descent rate = 1500 ft / min
- Descent time (@230 KCAS) = 13 min

126	The system shall provide forecasting data of the ice creation to the Flight Crew
41	**1.4 Safety Requirements**
42	Probability of single catastrophic failure of the system shall be less than 1*E -09 per Flight Hour.
43	Each ice protection components shall be redundant in order to guarantee ice protection functionality after single failure event.
44	The ice-protection failure which is categorized as catastrophic is the undetected malfunction of ice-detection.
45	**1.5 Physical Requirements**
46	The weight of the ice-protection system shall not exceed 150 kg.
124	The power consumption of the system shall not exceed TBD kW.
47	**1.6 Functional Requirements**
48	The de/anti icing system shall provide ice protection of the outboard and inboard wing leading edges.
49	The de/anti icing system shall provide ice protection of the horizontal stabilizer leading edges.
50	The de/anti icing system shall provide ice protection of the vertical tail leading edges.
51	The de/anti icing system shall provide protection of the engine inlets.
52	The ice-protection system shall prevent ice formation on the Windshields and Side windows by permanent electric heating.
53	The ice-protection system shall prevent ice formation on the Air Data Sensors (pitot and static tubes, AoA sensor) by permanent electric heating.
54	The ice-protection system shall prevent ice formation on the Horns by permanent electric heating.
55	The ice-protection system shall prevent ice formation on the Water Waste by permanent electric heating.
56	Windshield and side windows electrical heating shall be controlled by an independant temperature controller.
57	The De-Icing Control Unit shall control the de-icing cycles and include control logics to automatically select the type of cycle (high- or low- speed cycle) depending on Static Air Temperature.
58	The system shall detect ice condition.
59	The system shall be able to measure the ice thickness of the accretion.
60	In non-icing conditions, the equipment shall enable the aerodynamic profile.
61	The ice protection system shall provide the monitoring of its health status.
62	The ice protection system shall provide the monitoring of its operating status.
125	The ice protection system shall predict the ice formation with in the near future (TBD min).
63	**1.7 Performance Requirements**
64	The ice protection system shall prevent ice accretion on the wing greater than TBD mm (according to acceptable degradation of aerodynamic performance.
65	The ice protection system shall prevent ice accretion on the horizontal tail greater than TBD mm (according the acceptable degredation of aerodynamic performance).
66	The ice protection system shall prevent ice accretion on the vertical tail greater than TBD mm (according the acceptable degredation of aerodynamic performance).
67	**1.8 Installation Requirements**
68	The ice protection system shall be installed on the wing with the following geometry data: • Wing Aerodynamic Reference Area 80 sqm • Mean Aerodynamic Chord 2800 mm • Span 29 m • Aspect Ratio 10.5 • Taper Ration (Top Chord / Root Chord) 0.6 • Root Chord 3000 mm • Tip Chord 1900 mm • Front Spar Position (% chord) 16 • Rear Spar Position (% chord) 60 • Relative thickness (Root/Tip) 15.5% / 11%
69	The ice protection system shall be installed on the horizontal tail with the following geometry data: • Tail plane Reference Area 15 sqm • Mean Aerodynamic Chord (MAC) 1800 mm • Span 8200 mm • Aspect Ratio 4.6 • Taper Ratio (Tip Chord/C/L Chord) 0.6 • Root Chord 2250 mm • Tip Chord 1350 mm • Relative thickness (Root/Tip) 12% / 12%

70	The ice protection system shall be installed on the vertical tail with the following geometry data: • Fin Reference Area 17.0 sqm • Mean Aerodynamic Chord (MAC) 3400 mm • Span 5140 mm • Aspect Ratio 1.55 • Taper Ratio (Tip Chord / Root Chord) 0.6 • Root Chord 4150 mm • Tip Chord 2500 mm • Relative thickness (Root / Tip) 12% / 15% • Front Spar / Rear Spar Position 15% / 55%
71	**1.9 Cost Requirements**
72	The Cost for purchase and installation of the de-icing system shall not exceed xxx Euro.
123	The cost for operating the system shall not exceed xxx Euro per flight hour.
73	**1.10 Maintainability, Testability Requirements**
74	Design, installation and accessibility of the systems, equipments and LRU will be such to minimize the maintenance work load and skill level in order to achieve the objective of increased A/C availability.
75	Test points for functional testing on the ground shall be fittet.
76	All metal ice-protection components (ducts..) shall be of simple quick release type.
77	All de-ice equipment covers and access panels shall ensure that all hinged access panel shall remain fully open without hand support.
78	The design of any latching / locking mechanism shall include warning and indication means for a detection of a not secured condition after any maintenance operation.
79	The design of the system components which have weight higher than 20 kg shall incorporate specific handling device or provision for GSE usage (and relevant marking).
80	The system shall be tested on Ground without system components disconnections.
81	For the rectification of defects on the De-Ice Systen HW, the MTTR (Mean Time To Repair) shall not exceed TBD minutes.
82	The TTR (Time to Repair) (max. 95%) shall not be higher than 120 minutes.
83	The TTR represents the predicted time to recover (remove, install, re-test) in which 95% of the maintenance functions shall be completed on HW items.
84	The ice protection system shall be "maintenance free" during a storage period up to 5 years.

8.12 Modelica description of worst case flight scenario

```
model ScenarioMissionProfile1
  import Modelica.SIunits;

  // scenario parameters
  parameter Real rateOfClimb(unit="m/s") = 25 * 0.3048 "1 ft = 0.3048m";
  parameter SIunits.Duration timeToClimb = 800;
  parameter SIunits.Distance distanceTraveledInClimb = 44 * 1852
    "1 nautical mile = 1.852 * 1000 m";
  parameter SIunits.Velocity climbSpeed = 102;

  parameter SIunits.Distance cruiseAltitude = 20000 * 0.3048 "1 ft = 0.3048m";
  parameter SIunits.Duration timeToCruise = 10008;
//parameter SIunits.Distance distanceTraveledInCruise = 1651322 "Distance travelled in cruise";
  SIunits.Distance distanceTraveledInCruise "Distance travelled in cruise";
  parameter SIunits.Velocity cruiseSpeed = 165;

  parameter Real rateOfDescent(unit="m/s") = 25 * 0.3048 "1 ft = 0.3048m";
  parameter SIunits.Duration timeToDescent = 800;
  parameter SIunits.Distance distanceTraveledInDescent = 59 * 1852
    "1 nautical mile = 1.852 * 1000 m";
  parameter SIunits.Velocity descentSpeed = 137;

  parameter SIunits.Distance lowerLimitOfPolygonRange = 1219.2 "1 ft = 0.3048m";
  parameter SIunits.Temperature lowerLimitForIceAccretion = 0;

  parameter SIunits.Distance rasterUnit = 23 * 1852
    "Raster unit of 23 nautical miles";
  parameter SIunits.Distance rasterUnitForContAndIntermittentIcing = 3 * 1852
    "The first X nautical miles we have cont. and intermittent icing.";

  // scenario variables
  SIunits.Velocity speed "aircraft speed";
  SIunits.Distance altitude "aircraft altitude";
  Real altitudeInFeet =  altitude / 0.3048 "in feet";
  SIunits.Distance horizonalDistanceTravelled;
  Real horizonalDistanceTravelledInNMI = horizonalDistanceTravelled/1852
    "In nautical miles";

  // data obtained based on scenario variables
  Real LWC(unit="g/m3") "Liquid Water Contents";
  Real isInBlackIcingConditionsPolygon;
  Real isInGreyIcingConditionsPolygon;

  // aux variables
  SIunits.Distance distanceInBlackPolygon
    "distance flown from the point when the black polygon was entered last time.";
  Real distanceInBlackPolygonInNMI( unit="nmi") = distanceInBlackPolygon / 1852
    "in nautical miles";
  SIunits.Distance distanceStartInBlackPolygon
    "distance when the black polygon was entered last time.";

  Boolean isInFirst3MilesOfRasterInBlackPolygon;
  Boolean isContAndIntermittentIcing;
  Boolean isIntermittentIcing;
  Boolean isContIcing;

  /* **************************************************** */
  // black area (cont. icing) on the icing cond. diagram in feet.
  /* It is the scnenario with cruising in icing cond (in black polygon). This scnario is the worst case.  */
//  parameter Real vertx_black[:]={3700,   12000,   22000,   22000,   13500,   3700,   3700};
//  parameter Real verty_black[:]={-3,    -3,      -22.5,   -30,     -30,     -10,     -3};

  // grey area (intermittent icing) on the icing cond. diagram.
//  parameter Real vertx_grey[:]={13500,   22000,   22000,     31000,    18500,   13500};
//  parameter Real verty_grey[:]={-30,    -30,      -22.5,     -40,      -40,     -30};
  /* **************************************************** */
```

```
// black area (cont. icing) on the icing cond. diagram in meter.
/* It is the scnenario with cruising in icing cond (in black polygon). This scnario is the worst case.  */
parameter Real vertx_black[:]={1127.76,   3657.6,   6705.6,   6705.6,   4114.8,   1127.76,   1127.76};
parameter Real verty_black[:]={-3,    -3,      -22.5,   -30,     -30,     -10,     -3};

// grey area (intermittent icing) on the icing cond. diagram.
parameter Real vertx_grey[:]={4114.8,   6705.6,   6705.6,     9448.8,    5638.8,   4114.8};
parameter Real verty_grey[:]={-30,    -30,      -22.5,     -40,      -40,     -30};
/* ************************************************** */

Real distanceResidualInRaster
  "inidcates the residual of the distance after 23 mile";
//Boolean isInCruise = time > timeToClimb and time < (timeToClimb + timeToCruise);

Modelica.Blocks.Interfaces.RealOutput p_LWC "LWC"
                                     annotation (Placement(
     transformation(extent={{100,-30},{120,-10}}),iconTransformation(extent={{100,-26},
        {112,-14}})));
Modelica.Blocks.Interfaces.RealOutput p_v "Aircraft speed"
                                    annotation (Placement(
     transformation(extent={{100,10},{120,30}}),   iconTransformation(extent={{100,14},
        {112,26}})));

Modelica.Blocks.Interfaces.RealInput p_LWC_IntermittentContIcing annotation (
    Placement(transformation(
     extent={{-7,-10},{7,10}},
     rotation=-90,
     origin={60,107}), iconTransformation(
     extent={{-6,-6},{6,6}},
     rotation=-90,
     origin={60,106})));
Modelica.Blocks.Interfaces.RealInput p_LWC_ContIcing annotation (Placement(
     transformation(
     extent={{-7,-10},{7,10}},
     rotation=-90,
     origin={16,107}), iconTransformation(
     extent={{-6,-6},{6,6}},
     rotation=-90,
     origin={20,106})));
Modelica.Blocks.Interfaces.RealOutput p_altitude annotation (Placement(
     transformation(
     extent={{-8,-8},{8,8}},
     rotation=90,
     origin={-58,104}), iconTransformation(
     extent={{-6,-6},{6,6}},
     rotation=90,
     origin={-60,106})));
Modelica.Blocks.Interfaces.RealInput p_temperature annotation (Placement(
     transformation(
     extent={{-6,-9},{6,9}},
     rotation=-90,
     origin={-23,106}), iconTransformation(
     extent={{-6,-6},{6,6}},
     rotation=-90,
     origin={-20,106})));

equation
 // Ice conditions
 isInBlackIcingConditionsPolygon = Utilities.pnpoly(7, vertx_black, verty_black, altitude, p_temperature);
 isInGreyIcingConditionsPolygon = Utilities.pnpoly(6, vertx_grey, verty_grey, altitude, p_temperature);

 // remember the distance each time the black polygon is entered or 20 nautical miles are flown in black polygon
 distanceResidualInRaster = mod(horizonalDistanceTravelled,rasterUnit);
 when {isInBlackIcingConditionsPolygon > 0, distanceResidualInRaster < 0.001} then
    distanceStartInBlackPolygon = horizonalDistanceTravelled;
 end when;
 /* distance travelled in the black polygon
```

```
 TODO: it is not clear whether which distance it is ...*/
 distanceInBlackPolygon = if (isInBlackIcingConditionsPolygon > 0 and distanceStartInBlackPolygon > 0) then horizonalDistanceTravelled
- distanceStartInBlackPolygon else 0;

 isInFirst3MilesOfRasterInBlackPolygon  = isInBlackIcingConditionsPolygon > 0 and distanceInBlackPolygon > 0 and distanceInBlackPol
ygon < rasterUnitForContAndIntermittentIcing;
 isContAndIntermittentIcing           = altitude > lowerLimitOfPolygonRange and isInBlackIcingConditionsPolygon > 0;
 isIntermittentIcing                 = altitude > lowerLimitOfPolygonRange and isInGreyIcingConditionsPolygon > 0;
 isContIcing                        = p_temperature < lowerLimitForIceAccretion and not isContAndIntermittentIcing and not isIntermittentIcing;

 //LWC
 if isContAndIntermittentIcing and isInFirst3MilesOfRasterInBlackPolygon then // cont. AND intermittent icing
   LWC = p_LWC_ContIcing + p_LWC_IntermittentContIcing; // "superposition principle" within the first 3 nautical miles of every 20 natic
al miles.
 elseif isContAndIntermittentIcing then // only cont. icing after 3 natical miles of every 20 nautical miles travelled.
   LWC = p_LWC_ContIcing;
 elseif isIntermittentIcing then // only intermittent icing
   LWC = p_LWC_IntermittentContIcing;
 elseif isContIcing then // only cont. icing
   /* cont. icing.
   TODO: validate it. In this scenario we start with -3.3°C, so there will be ice accretion.
   However, what if the the temperature is positive at SL? */
   LWC = p_LWC_ContIcing;
 else // no icing
   LWC = 0;
 end if;

 // at the end of cruising remember the distance travelled in cruise
 when time >= (timeToClimb + timeToCruise) then
   distanceTraveledInCruise = cruiseSpeed *  (time - timeToClimb);
 end when;

 // scenario
 if time < timeToClimb then // climbing
   altitude = rateOfClimb * time;
   speed = climbSpeed;
   // ratio based on distanceTraveledInClimb and climb time
   horizonalDistanceTravelled = (time * distanceTraveledInClimb)/timeToClimb;

 elseif time < (timeToClimb + timeToCruise) then // cruising
   altitude = cruiseAltitude;
   speed = cruiseSpeed;
   horizonalDistanceTravelled = distanceTraveledInClimb + speed *  (time - timeToClimb);

 elseif time < (timeToClimb + timeToCruise + timeToDescent) then // descending
   altitude = cruiseAltitude - (rateOfDescent * (time - timeToClimb - timeToCruise));
   speed = descentSpeed;
   // ratio based on distanceTraveledInClimb and climb time
   horizonalDistanceTravelled = distanceTraveledInClimb + distanceTraveledInCruise + (((time - timeToClimb -
timeToCruise) * distanceTraveledInDescent) / timeToDescent);

 else
   altitude = 0;
   speed = 0;
   horizonalDistanceTravelled = 0;
 end if;

 // Ports
 p_altitude = altitude;
 p_v = speed;
 p_LWC = LWC;

 annotation (Diagram(coordinateSystem(preserveAspectRatio=false, extent={{-100,
        -100},{100,100}}), graphics),                                  Icon(
      coordinateSystem(preserveAspectRatio=false, extent={{-100,-100},{100,100}}),
      graphics={Bitmap(extent={{-100,90},{94,-104}},
                                      fileName="modelica://CRYSTALPutblicUseCase/img/screenshot_mission_profile.gif"),
        Text(
```

```
        extent={{-20,70},{22,28}},
        lineColor={0,128,0},
        textString="%name%")}));
end ScenarioMissionProfile1;
```

8.13 Modelica description of Ice Accretion physics

```
model IceAccretionDynamics
  import Modelica.SIunits;

  Real E "efficiency";
  Real m "total mass in g";

  Real v "speed at which the aircraft is flying at the considered mission leg";

  parameter Real t(unit="m") = 0.36
    "maximum airfoil thickness in meters, considering the wing flying at an angle of attack of 4°";
  parameter SIunits.Length b = 22.1 "protected wingspan";

  Real LWC(unit="g/m3") "Liquid Water Contents";

  Modelica.Blocks.Interfaces.RealInput p_v annotation (Placement(transformation(
          extent={{-114,8},{-100,28}}),   iconTransformation(extent={{-116,12},{
            -100,28}})));
  Modelica.Blocks.Interfaces.RealOutput p_m annotation (Placement(
        transformation(extent={{100,-10},{120,10}}), iconTransformation(extent={
            {100,-10},{120,10}})));

  parameter Real iis(unit="g/m2") = 7.43 "Ice impingement surface";
  //parameter Real is = t*b "Imaginary Sieve";

  Real specificWeight( unit="g/m2") = m/iis
    "Specific weight of ice over a square meter.";
  parameter Real iceDensity(unit="g/m3") = 916869;
  Real meanIceThickness(unit="m") = specificWeight/iceDensity;

  Modelica.Blocks.Interfaces.RealInput p_LWC annotation (Placement(
        transformation(
        extent={{7,10},{-7,-10}},
        rotation=180,
        origin={-107,-22}), iconTransformation(
        extent={{8,8},{-8,-8}},
        rotation=180,
        origin={-108,-20})));
equation

  E = 0.00324*(v/t)^0.613;
  der(m)=v*t*b*LWC*E;

  //inputs
  v = p_v;
  LWC = p_LWC;

  // output
  p_m = m;

  annotation (Icon(coordinateSystem(preserveAspectRatio=false, extent={{-100,-100},
            {100,100}}), graphics={Bitmap(extent={{-90,74},{94,-
116}},fileName="modelica://CRYSTALPutblicUseCase/img/iceAccretion.gif"),
                                  Text(
          extent={{-52,128},{50,26}},
          lineColor={0,0,255},
          textString="%name%")}),
            Diagram(coordinateSystem(preserveAspectRatio=false, extent={{-100,
            -100},{100,100}}), graphics));
end IceAccretionDynamics;
```