Gestion des interactions pour l'évaluation en phase de préconception, des architectures 3D de systèmes sous contraintes multi-physiques, application à la thermique

Romain Barbedienne

université
PARIS-SACLAY

NNT : 2017SACLC026

CentraleSupélec

Gestion des interactions pour l'évaluation en phase de préconception, des architectures 3D de systèmes sous contraintes multi-physiques, application à la thermique

Thèse de doctorat de l'Université Paris-Saclay
préparée à CentraleSupélec

École doctorale n°573 INTERFACE
Spécialité de doctorat: Science et technologies industrielles

Thèse présentée et soutenue à Saint-Ouen, le 21 avril 2017, par

Romain Barbedienne

Composition du Jury :

Omar Hammami Professeur à l'ENSTA ParisTech	Président
Stanislao Patalano Associate Professor, à l'université de Naples Federico II	Rapporteur
Stephan Rudolph Privatdozent à l'université de Stuttgart	Rapporteur
Marc Budinger Maître de conférences HDR à l'INSA Toulouse	Examinateur
Eric Duceau Professeur à l'Ecole des Ponts Paris-Tech	Examinateur
Alain Rivière Professeur à l'Institut supérieur de mécanique de Paris	Directeur de thèse
Jean-Yves Choley Maître de conférences HDR à l'Institut supérieur de mécanique de Paris	Encadrant de thèse
Olivia Penas Ingénieure de recherche à l'Institut supérieur de mécanique de Paris	Co-encadrante de thèse

Thèse de doctorat

This thesis has been funded by the Research Institute of Technology IRT SystemX within the SIM project (French acronym for Multidisciplinary Simulation) and has been carried out under the supervision of Jean-Yves Choley and Olivia Penas from the Quartz Laboratory at Supmeca.

Remerciements

First at all, I would like to warmly thank Stanislao Patalano, and Stephan Rudolph first to have accepted to review my PhD manuscript, even with the short time limits accorded, and also for the quality of their reports, and their interesting questions during the defense. I address my thanks to Omar Hammami for presiding the defense, as well as Eric Duceau and Marc Budinger to examine this thesis, I appreciated their encouragement, insightful comments, and their valuable questions.

Je remercie mon directeur de thèse Alain Rivière pour la liberté et la confiance qu'il m'a accordé ainsi que mon encadrant Jean-Yves Choley, à l'origine du sujet de cette thèse, pour son écoute et ses conseils qui m'ont été précieux. Je voudrais également remercier Olivia Penas, mon encadrante pour sa rigueur scientifique et son expertise dans le domaine de l'ingénierie des systèmes. Mais je tenais surtout à te remercier pour ton soutien pendant ces trois années, pour ta patience, et d'avoir été présente à chaque fois que j'avais besoin de toi.

Merci aux collaborateurs du projet SIM, et particulièrement à Laurent Gasser de m'avoir fortement soutenu, ainsi que pour son expertise en ingénierie système et informatique. A Paul Labrogère, et Etienne de Pommery pour m'avoir fait confiance. Je remercie Yves Baudier pour m'avoir impliqué dans le projet Européen TOICA, et Michel Fouquemberg pour tous les conseils, sa franchise, et l'expertise qu'il a partagée avec moi en thermique. Je remercie les membres du laboratoire Quartz, et particulièrement Régis Plateaux, de m'avoir donné l'opportunité de mettre en application mes travaux lors d'un cours avec les élèves ingénieurs du parcours mécatronique. Mon seul regret aura été de ne pas avoir pu travailler plus avec toi, notamment sur les aspects topologiques. Je remercie également Philippe Serré pour son aide, et son expertise dans le domaine de la modélisation 3D.

A tout ceux qui travaillent dans l'ombre pour que la thèse se passe bien, et notamment Christel Compagnon et Véronique Da-Silva, merci pour votre bonne humeur, vos sourires communicatifs et d'avoir toujours eu un petit mot pour me soutenir. Merci à Sophie Huet, et Natalie Limonta pour leur aide concernant l'aspect administratif de l'IRT. Merci à Suzanne Thuron, et Emmanuelle Coplo pour leur accueil au sein de l'école doctorale.

Et il y a eu tous ceux sans qui la thèse n'aurait pas été la même : mes collègues et amis de l'IRT SystemX avec qui j'ai partagé des séances de sport, mes déjeuners, certaines de mes soirées (quand je ne travaillais pas), et mes vacances pendant 3 ans. Un énorme merci à Gauthier, Hadrien, Mian, Anaïs, Kevin, Aminata, Jing, Emna et Laura pour tout ! Je remercie également Mouadh pour son soutien et sa philosophie qui m'a poussé à réfléchir. Je remercie Yves, Sulivan et Gabriel pour leurs humours. Je remercie Adrien G., Adrien C., Marco, Etienne, Aude et Kamil pour leur accueil au Laboratoire Quartz

Ceux sans qui rien de tout ça n'aurait existé, mes parents, et mes deux frères qui ont toujours cru en moi.

Enfin, je dédie cette thèse à celle qui a partagé ces trois années à mes côtés. Merci Marine pour tout ce que tu m'as apporté.

Table of Contents

List of tables xi

List of Figures xiii

List of abbreviations xviii

List of relevant terms xxi

List of variables xxii

Introduction 0

I. Context 1

A. Position at the technological research institute SystemX 1

B. Industrial challenges 3

II. Research problem 4

III. Structure of the dissertation 5

Chapter 1 – Scientific context 8

I. Systems Engineering context 9

II. Related works 12

A. Geometrical and physical modeling integration in an MBSE context for conceptual design 12

B. 3D physical architecture assessment 14

C. Interaction management during the conceptual design phase 17

III. Conclusions 29

Chapter 2 - Approach selection process 30

I. Expression of needs 31

II. Analysis of existing collaborative structures 36

III. A proposal of alternative approaches 39

A. First approach for data exchange automation of current industrial practices 40

B. Second Approach: geometrical and physical enrichments for semantic interoperability and traceability 41

C. Third approach: physical and geometrical specifications and traceability in a single 3D physical platform 43

IV. Approach selection 44

V. Demonstrator implementation environment definition 46
A. Validation of the SysML Language for the System modeling 46
B. Selection of the model transformation process 49
VI. Conclusions 52
Chapter 3 – Geometrical modeling 54
I. Expression of needs 55
II. Existing geometrical modeling approaches 56
A. Geometry modeling 57
B. Topology modeling 61
III. Analysis for the choice of geometrical modeling in SAMOS ... 64
A. Choice of geometry and topology modeling for geometrical modeling 64
B. Integration of face modeling for SAMOS 66
IV. The GERTRUDe SysML extension 67
A. Definition of the GERTRUDe metamodel 67
B. Example of a description of a finite cylinder geometrical 68
V. Geometrical information model transformation strategy......... 70
A. Different M2M transformation strategies 70
B. Analysis of the "Neutral format usage" strategy applied to SAMOS 71
C. Analysis of the "Direct translation" strategy applied to SAMOS 73
D. Selection of the model transformation strategy for SAMOS 73
VI. Conclusions 74
Chapter 4– Thermal modeling 76
I. Expression of needs 78
II. Preliminary thermal hypotheses and reminders 79
III. Description of the main existing thermal modeling approaches 81
A. Analytic calculation 81
B. Thermal resistance modeling 82
C. Finite element analysis /Finite volume method analysis (FEA/FVM) 84
IV. Thermal modeling selection for SAMOS 86
A. Selection of the thermal modeling approach 86

B. Description of the temperature calculation with the analytic method selected for thermal modeling....86
V. Thermal modeling for the SAMOS framework 98
A. Component, Medium, Interacting Face and thermal interaction network concepts....98
B. Thermal modeling according to geometrical views.... 100
C. Example of complete thermal modeling with a multiple component architecture.... 104
VI. Use of the TTRS theory for the analytical thermal modeling. 109
A. Conduction modeling.... 109
B. Convection modeling 114
C. Radiation modeling....118
VII. TheReSE SysML extension 123
VIII. Conclusions 125
Chapter 5 – Implementation of the Thermal 3D Sketcher.....127
I. Expression of needs and corresponding tool selection.... 130
A. Demonstrator implementation requirements 130
B. Tool selection 132
II. Algorithmic modeling of the Thermal 3D Sketcher.... 140
A. Description of the geometrical model transformation.... 140
B. Descriptions of the thermal model transformations and algorithms.... 149
III. Developments in SysML (Atego-PTC environment).... 153
A. GERTRUDe 153
B. TheReSE.... 157
C. Layer Management to improve readability.... 159
IV. Development in 3D CAD (FreeCAD environment).... 162
A. Geometric constraint solving 162
B. SAMOS processing.... 166
V. Development in Modelica (OpenModelica environment).... 174
A. Development of a Modelica Library related to the shell view. 174
B. Model transformation and perspectives.... 179
VI. Conclusions 180
Chapter 6 – Scenario-based validation.... 181

I. System conveyer 183
A. Description of the case study 183
B. Geometry definition in GERTRUDe 184
C. Generation of the 3D CAD architecture.. 187
D. CAD2GERTRUDe transformation 188
II. Helicopter bay 189
A. Case study description 189
B. Application of SAMOS approach 191
C. Modelica modeling.. 195
D. Verification 198
III. Airplane cab 200
A. Case study description 200
B. Geometrical requirements 201
C. Thermal behavior traceability through TheReSE....................................204
IV. Electric Power Train 207
A. Geometry modeling 207
B. Thermal requirements 208
C. Geometrical modeling 209
D. Thermal modeling 212
V. SAMOS verification case study: electric bicycle.......................215
A. Case study description 215
B. Geometry modeling 217
C. Survey results 219
VI. Verification and validation of the SAMOS and thermal 3D sketcher...221
VII. Conclusions 222
Conclusion and future work 223
I. Observation 224
II. Need...224
III. Contributions..225
IV. Perspectives 228
List of personal publications 230

Appendix 1 – Résumé étendu 231
Appendix 2 – The SIM process 235
Appendix 3 – System Modeling Language/ Unified Modeling Language 236
I. Overview of the various diagrams 237
II. Description of the diagrams used 239
A. Requirement diagram 239
B. Block definition diagram & Class diagram 239
C. Internal block diagram & Composite structure diagram 240
D. Activity diagram 241
E. State machine diagram 241
F. Sequence diagram 242
Appendix 4 – The thermal laws used 243
I. Conduction 244
A. Calculation of the Laplacian equation in a 3D space 244
B. Solving the Laplacian equation for 3D geometry 246
II. Convection 252
III. Radiation 254

List of tables

Table 1: Proposed data exchange technical solution for the corresponding collaboration models ... 26
Table 2: Requirements specification ... 34
Table 3: Description of additional selection criteria related requirements ... 36
Table 4: Comparison of each collaborative structure according to the selection criteria related requirements ... 37
Table 5: Description of the additional requirements related to the model transformation process ... 38
Table 6: Comparison of existing model transformation approaches regarding the previous specific requirements. ... 39
Table 7: Analysis of the three different approaches. ... 45
Table 8: Description of the additional requirements related to the System modeling language ... 47
Table 9: Analysis of the system modeling requirements regarding the SysML language ... 48
Table 10: Additional requirements related to the model transformation process. ... 51
Table 11: Evaluation of existing exogenous methods regarding the requirements of the model transformation implementation method. ... 51
Table 12: Additional requirements for geometrical modeling ... 56
Table 13: The 13 TTRS constraints (Clement, et al., 1998) ... 62
Table 14: Evaluation of geometry modeling approaches according to the requirement specifications. ... 65
Table 15: Evaluation of topology modeling approaches according to SAMOS requirement specifications. ... 65
Table 16: Pros and cons of using direct translations and neutral formats usage (Fowler, 1995) ... 71
Table 17: Description of additional thermal requirements ... 79
Table 18: Main thermal laws considered. ... 80
Table 19: Main thermal resistance formula ... 83
Table 20: Evaluation of the different thermal modeling approaches according to the requirements for the SAMOS implementation. ... 86
Table 21: Calculation of the eigenvalues and eigenvectors according to different boundary condition cases. ... 89
Table 22: Eigenfunctions bases associated to specific coordinate systems ... 90
Table 23: Definition of the dimensionless numbers used for the Nusselt correlation. ... 92
Table 24: Nusselt correlation for forced convection applied on various geometries of solid. ... 93
Table 25 Representation of elements used in thermal interaction network ... 100
Table 26: Various conduction phenomena and their related thermal networking representation ... 101
Table 27: Various convection phenomena for a solid and their related thermal networking representation ... 102
Table 28: Various convection phenomena for a fluid and their related thermal networking representation ... 103

Table 29: Various radiation phenomena for a solid and their related thermal networking representation. 104
Table 30: Resulting thermal analytical laws for 1D volumetric conduction according to the face symmetry. 109
Table 31: Projection bases choice for analytical thermal laws simplification for volume conduction regarding the TTRS class considered 110
Table 32: Calculation of the cube parameter according to the sphere size 114
Table 33: Levels definition for each DoE factor used for the convection simulation 115
Table 34: Simulation results with the maximal error found for each geometry....116
Table 35: Analytic thermal equations associated with the equivalence models. 117
Table 36: Associated constraints between the three TTRS classes considered for the view factor calculation. 119
Table 37: Expression of View factors according to the TTRS constraints involved. 120
Table 38: Implementation derived requirements....130
Table 39: Analysis of main tools based on SysML regarding implementation requirements. 133
Table 40: Benchmark of main 3D CAD tools. 134
Table 41: Additional requirements related to the simulation language 135
Table 42: Validation of Modelica language adequacy. 137
Table 43: Comparison of several usual Modelica tools. 138
Table 44: Complexity weight attributed to each operation or function occurring in the equations to be solved. 147
Table 45: New ports for the geometry and material definition. 175
Table 46: Modelica models developed to define simulation conditions. 177
Table 47: Conduction Modelica models according to the component TTRS class involved. 179
Table 48: Equipment alternative boundary conditions....190
Table 49: Comparison of temperatures according to the section altitude. 199
Table 50: Description of component requirements. 201
Table 51: Thermal requirements for the first architecture. 208
Table 52: Thermal requirements for the second architecture. 209
Table 53 Thermal requirements for fans 209
Table 54: Description of the additional component geometries. 215
Table 55: Thermal specifications of the electrical bicycle components 217
Table 56: Summary of the survey results given by students. 219
Table 57: Validation of SAMOS and the thermal 3D Sketcher. 221
Table 58: Description of the different diagrams....238
Table 59 Physical meaning and equation of the Grashof number. 252
Table 60 : Calculation of the Nusselt correlation for natural convection....253
Table 61: Definition of radiation parameters. 254

List of Figures

Figure 1: Methodologies, tools and processes developed in the SIM project to address simulation architect assignments. 3
Figure 2: System engineering design cycle (DODSMC, 2001) 10
Figure 3: SysML Diagrams (Object Management Group, 2006)....11
Figure 4: MBSE approach (Mhenni, et al., 2014)....11
Figure 5: Simulation issue of two 3D alternative interpretations of a 2D iconic 4 bar model (Plateaux, et al., 2009)....16
Figure 6: Main activities of the conceptual design phase (Kossiakoff, et al., 2011)....17
Figure 7: Description of the FBS framework (Gero & Kannengiesser, 2000)....18
Figure 8: System Architect tasks (Muller, 2011)....20
Figure 9: description of systems engineering domains (Kossiakoff, et al., 2011) 22
Figure 10: Extracted activities from SIM process....22
Figure 11: The different kinds of interaction environments....23
Figure 12: Close coupled collaboration used in a virtual environment (Wolff, et al., 2007) 23
Figure 13: Illustration of complexity in the Actor Network Theory 24
Figure 14: Description of the different structures of data exchange technical solutions 26
Figure 15: Current design cycle and its proposed evolution. 32
Figure 16: Position of the proposed framework in current industrial design cycle. 32
Figure 17: Research issue analysis related derived requirements....33
Figure 18: Selection criteria related requirements for choosing the collaborative structure to be implemented....36
Figure 19: Requirement diagram for the analysis of the existing model transformation process. 38
Figure 20: Principles of the first physical 3D architecture framework....40
Figure 21: Principles of the second physical 3D architecture framework. 42
Figure 22: Principles of the third physical 3D architecture framework....43
Figure 23: Overview of the SAMOS framework applied to thermal analysis. 46
Figure 24: Flow chart of system modeling language requirements. 47
Figure 25: Model transformation implementation method requirements 50
Figure 26: Geometry modeling requirements....55
Figure 27: Example of geometry and topology modeling for a finite cylinder....56
Figure 28: Example of wireframe modeling....57
Figure 29: Face identification, the ambiguity of wireframe modeling. 57
Figure 30: Half space based on surface modeling 58
Figure 31 Example of the half space concept. 59
Figure 32: Example of Constructive Solid Geometry operations 59
Figure 33: Example of B-rep modeling....60
Figure 34: TTRS and MRGE definition....61
Figure 35: TTRS modeling of a finite cylinder. 62
Figure 36: Example of the C06 constraint between the MRGE of the two plane TTRS. ...63
Figure 37: Multi-scale TTRS decomposition for an hollow cylinder....64
Figure 38: Face generation process. 66
Figure 39: Different geometrical views. 66

Figure 40: GERTRUDe metamodel used to support geometrical information in SysML. . 68
Figure 41: Example of a cylindrical component modeling with GERTRUDe 69
Figure 42: TTRS modeling of a Finite cylinder with GERTRUDe 70
Figure 43: Model transformation formalization using neutral formats (Standards) files. 72
Figure 44: Model transformation formalization using direct translation. 73
Figure 45: Thermal modeling related requirements. 78
Figure 46: Analogy of the thermal resistance regarding the electrical resistance. 82
Figure 47 Thermal modeling based on a solid geometrical discretization. 85
Figure 48: 2D elementary representation for conduction modeling 87
Figure 49: Description of variables used in the Bernoulli principle. 95
Figure 50: View factor calculation parameters. 97
Figure 51: Radiation heat exchanges between two surfaces. 97
Figure 52: Example of the definition of an Interacting Face between a component and a medium. 99
Figure 53: Medium slicing process for the example architecture. 105
Figure 54: IF definition of the components architecture for the layer l_i. 106
Figure 55: Example of thermal interacting faces (IF) and medium definition (2D view). . 107
Figure 56: Example of a thermal interaction network. 108
Figure 57: L-bar geometry cutting with TTRS extension. 111
Figure 58: Thermal interaction network modeling of volume conduction in the initial geometry and in the "cut" geometry of the 2d projection using the example of an L-bar. 111
Figure 59: Coupled contact-volume conduction modeling between two components through a partial face, based on TTRS construction cutting process. 112
Figure 60: Geometry approximation for convection modeling 115
Figure 61: Finite element modeling for the geometry approximation evaluation. 118
Figure 62: Different view factors between two spheres depending on the TTRS constraint considered. 119
Figure 63: Data model of TheReSE. 124
Figure 64: Global framework of the thermal 3D Sketcher. 129
Figure 65: Geometrical and thermal related implementation derived requirements. 131
Figure 66: SysML tool selection related requirements 132
Figure 67: Requirements related to the 3D CAD tool. 134
Figure 68: Simulation language related requirements 136
Figure 69: Modelica tool selection related requirement 137
Figure 70: The implemented SAMOS framework with Software and their related mapping links. 139
Figure 71: Detailed model transformation process presented in an activity diagram. 141
Figure 72: Model Transformation Metamodel using class diagram 142
Figure 73: Transformation of a 3D component into GERTRUDe. 144
Figure 74: 3D CAD into TTRS representation transformation 145
Figure 75: Compilation process. 145
Figure 76: State machine diagram of the lexical analysis stage. 146
Figure 77: Detailed description of the "Transform GERTRUDe2CAD model" activity, using sequence diagram. 148
Figure 78: Detailed model transformation process represented in an activity diagram. 149

Figure 79: TheReSE - Modelica model transformation metamodel represented by a class diagram. 150
Figure 80: Profile diagram of GERTRUDe 153
Figure 81: Example of AssemblyDiagram used by design actor user....154
Figure 82: Toolbar of the AssemblyDiagram. 155
Figure 83: Contextual Menu (right-click) developed by scripts. 155
Figure 84: GERTRUDe GUI: example of a component addition....156
Figure 85: Example of error generated by a script included in GERTRUDe. 156
Figure 86: OCL rules to limit the number of TTRS involved by constraint. 157
Figure 87: Profile diagram of TheReSE. 158
Figure 88: Example of a PhysicsDiagram GUI....158
Figure 89: Example of an Interaction Diagram. 159
Figure 90: TTRS layer management....160
Figure 91: Thermal Physical Layer Management. 161
Figure 92: Constructive approach used on the finite cylinder example. 162
Figure 93: Solving geometrical constraints of the finite cylinder surfaces using the constructive approach. 163
Figure 94: Degree of Freedom modeling of the Finite cylinder. 164
Figure 95 : GUI of the thermal 3D Sketcher workbench....166
Figure 96: Transformation in the FreeCAD environment of a component initially defined in SysML without geometry. 166
Figure 97: Example of material definition. 168
Figure 98: Requirement tab. 169
Figure 99: "Add new Requirement" window. 170
Figure 100: The "Physics Tab" description. 171
Figure 101: The "Add solid boundary conditions" window. 172
Figure 102: Adding boundary conditions for fluid component 172
Figure 103: Thermal media definition. 173
Figure 104: Differences between flow variables and potential variables....176
Figure 105 : Conveyor case study description 183
Figure 106 : Initial physical architecture of the conveyor....184
Figure 107 : Geometrical enrichment of physical architecture by GERTRUDe 184
Figure 108: Construction of the double cylinder support geometry based on TTRS modeling....185
Figure 109: Specification of relative positioning constraints to build the support double cylinder TTRS....185
Figure 110: AssemblyParam diagram describing the TTRS model of the double cylinder. 186
Figure 111: Generation of 3D conveyor components from GERTRUDe components. ..187
Figure 112: Initial 3D spatial architecture of the conveyor system. 187
Figure 113: Proposed 3D spatial architecture in FreeCAD tool. 188
Figure 114: Expected updated physical architecture traced back in the GERTRUDe extension. 188
Figure 115: Example of the contact constraint between the right roller support and roller 2. 189
Figure 116: Avionic bay 3D architecture. 190

Figure 117: Definition of the geometry of each component with GERTRUDe GUI. 191
Figure 118: Results of geometrical definition of architecture components with GERTRUDe. 191
Figure 119: 3D architecture definition. 192
Figure 120: Material selection for each component. 193
Figure 121: Adding a new requirement. 193
Figure 122: Adding boundary conditions 194
Figure 123: Media definition. 195
Figure 124: The resulting Modelica model generated. 196
Figure 125: Automatic verification of simulation results regarding the requirements. 197
Figure 126: Resulting plotted curve of the maximum outlet air temperature. 198
Figure 127: ANSYS CFX simulation results on the 3D architecture of the bay. 198
Figure 128: Comparison results of the temperature according to the cut plane altitude....199
Figure 129: Air Conditioning system (A.C.) model in the airplane section considered. 200
Figure 130: Model of the C06 positioning constraint between the Floor and Calculator 1 with GERTRUDe. 201
Figure 131: Evolution of 3D architecture according to change of requirements and addition. 202
Figure 132 Assembly diagram of first and second classes......203
Figure 133: Extract of some additional thermal requirements traceability. 204
Figure 134: Example of traceability of the thermal convection model of Calculator 1. 205
Figure 135: Interaction Diagram of the air conditioning system for the first (top) and second class section (bottom). 206
Figure 136: Description of two architectures of the electric bus scenario. 207
Figure 137: Alternative EPT architectures with GERTRUDe. 210
Figure 138: Generated and enriched 3D modeling of the two architectures. 211
Figure 139: The Modelica model of architecture 1 generated. 212
Figure 140: Geometrical view of media and pipes. 213
Figure 141: Modelica model of Architecture 2. 213
Figure 142: Comparison of the cooling liquid temperature near the motor for architecture 1 and architecture 2. 214
Figure 143: Traceability management for a 3D thermal architecture 1 modeled in Modelica (left), transformed into the TheReSE enriched SysML model (right). 214
Figure 144: Comparison of the different 3D architectures. 216
Figure 145: Extract of the electric bike architecture enriched with geometry modeling....217
Figure 146: Fluid cutting modeling for the convection analysis of the 3D architecture. ...218
Figure 147: The Modelica convection model generated. 219
Figure 148: Evolution of design time as a function of the number of changes to the architecture. 220
Figure 149: SAMOS framework 225
Figure 150: Thermal 3D sketcher structure 226
Figure 151 Convex hull of assembly 228
Figure 152 : Processus SAMOS. 232
Figure 153 : Plateforme outillée de SAMOS : Modeleur 3D thermique. 233
Figure 154: Description of the SIM process 235
Figure 155 : SysML-UML diagrams 237
Figure 156: Cartesian coordinate system. 244

Figure 157 : Infinitesimal volume element in the rectangle parallelepiped coordinate system 246
Figure 158 : Cylindrical coordinate system on an infinitesimal volume element. 248
Figure 159: Spherical coordinate system infinitesimal volume element. 249
Figure 160 Transfer of electromagnetic radiation by a solid. 254

List of abbreviations

Abbreviation	Meaning
BDD	Block Definition Diagram
B-Rep	Boundary Representation
C	Component
CAD	Computer Aided Design
CSG	Constructive Solid Geometry
DFA	Design For Assembly
EMC	Electromagnetic Compatibility
EPT	Electric Power Train
FBS	Functional, Behavior, Structure
FEA	Finite Element Analysis
FF	Fluid-Fluid Coupling Medium
FIM	Fluid Intrinsic Medium
fs	Faces port
FVM	Finite Volume Method
GERTRUDe	Geometrical Extension Related to the TTRS Reference for a Unified Design
GPS	Geometrical Product Specification
GUI	Graphical User Interface
IBD	Internal Block Diagram
Id	Identifier
IF	Interacting Face
INCOSE	International Council on Systems Engineering
IT	Information technology
M	Medium

M2M	Model to Model
M2T	Model to Text
MBSE	Model Based System Engineering
MIC	Model Identity Cards
MOF	Meta Object Facility
MoP	Measure of Performances
MRGE	Minimal Geometric Reference Element
OCL	Object Constraint Language
OMG	Object Management Group
PDM	Product Data management
PIM	Platform-Independent Model
PSM	Platform-Specific Model
Req	Requirement
RFLP	Requirement, Functional, Logical, Physical
RGE	Relative Geometrical Elements
SA-CAD	System Architecting CAD
SAMOS	Spatial Architecture based on Multi-physics and Organization of Systems
SE	System Engineering
SF	Solid-Fluid Coupling Medium
SI	International System of Units
SIM	Solid Intrinsic Medium
SIM project	Multi-Disciplinary Simulation and Engineering project
SiMo	Simulation Model
SS	Solid-Solid Coupling Medium
STEP	STandard for the Exchange of Product model data
STRD	composite structure diagram

SysML	System Modeling Language
TheReSE	Thermal Related SysML Extension
TOICA	Thermal Overall Integrated Conception of Aircraft
TTRS	Technologically and Topologically Related Surfaces
UML	Unified Modeling Language
US	United State
VAF	Vehicle Architecture Framework
VB	Visual Basic
VIATRA	VIsual Automated model TRAnsformation system
VR	Virtual Reality
XMI	XML Metadata Interchange
XML	Extensible Markup Language

List of relevant terms

Terms	Definition
3D Architects	These actors of the conceptual design allocate the space consumption and the initial 3D architecture of the components based on the physical architecture and the 3D design.
Component	A component is a geometrical element without any thermal behavior equation. Nevertheless, the component can have thermal properties (also named boundary conditions).
GERTRUDe	Geometrical Extension Related to the TTRS Reference for a Unified Design is a SysML extension that provides a geometry to a component. This geometry is defined thanks to the TTRS theory enriched with intrinsic parameters. This extension supports the automatic generation of geometry into a 3D CAD tool.
Interacting Face	An Interacting Face (IF) is a face defined by the common contact area of two faces of two different components (including media).
Medium	A medium is a specific component including a thermal behavior (equations). There are two kinds of medium: - Intrinsic medium (when the thermal behavior occurs inside the component (e.g. volume conduction for solids or internal fluid movement for fluids), - or coupling medium between 2 different solids or between a fluid and a solid.
SAMOS framework	- SAMOS is a framework allowing the actors to easily exchange information during the conceptual design while limiting the risks of inconsistencies and misunderstandings.
Simulation Teams	Simulation Teams are composed of experts in different disciplines (electronics, mechanics, control, hardware/software, etc.) in charge of verifying that the behavior of the physical architecture meets the performance and spatial requirements through physical simulations.
System Architects	System Architects are actors of the conceptual design. They provide the physical architecture, from customer requirements and takes the final decision concerning the architecture choice.

List of variables

Symbol	Description	Unit (S.I.)
λ	thermal conductivity	$W.m^{-1}.K^{-1}$
$\overrightarrow{j_{th}}$	heat flux vector	$W.m^{-2}$
T	temperature	K
h	heat transfer coefficient	$W.m^{-2}.K$
T_s	temperature of the solid	K
T_f	temperature of the fluid	K
$\overrightarrow{n_S}$	unitary vector normal to the surface	Dimensionless
ϵ	emissivity	Dimensionless
σ	Stephan-Boltzmann constant	$W.m^{-2}.K^{-4}$
ϕ_{er}	exchanged rate of heat flow	W
ϕ_r/ϕ_e	respectively received and emitted rate of heat flow	W
$F_{e\to r}$	view factor from the emitting to the receiving component	Dimensionless
$F_{r\to e}$	view factor from the receiving to the emitting component	Dimensionless
R_{th}	Thermal resistance	$m^2.K.W^{-1}$
R_{Coth}	Constant section conduction resistance	$m^2.K.W^{-1}$
R_{Cyth}	Cylinder section conduction resistance	$m^2.K.W^{-1}$
R_{Cpth}	Spherical section conduction resistance	$m^2.K.W^{-1}$
R_{Cvth}	Convection thermal resistance	$m^2.K.W^{-1}$
R_{Rth}	Radiation thermal resistance	$m^2.K.W^{-1}$
e	height of extruded section	m
S	constant surface	m^2
R_e	external radius	m
R_i	internal radius	m

L	length of the cylinder	m
S_R	receiving apparent surface	m^2
T_{le}	approximated temperature of the emitting component	K
T_{lr}	approximated temperature of the receiving component	K
Re	Reynolds number	Dimensionless
Nu	Nusselt number	Dimensionless
Pr	Prandtl number	Dimensionless
g	gravity acceleration	$m.s^{-2}$
z	altitude of the point considered from a plane reference	m
p	fluid pressure at the point considered	$N.m^{-2}$
$\vec{v}$	velocity of the fluid (v designates the norm of the vector)	$m.s^{-1}$
ν	kinematic viscosity	$m^2.s^{-1}$
c_p	specific heat	$J.kg^{-1}.K$
μ	dynamic viscosity of the fluid	$N.s.m^{-2}$
ρ	density of the fluid	$kg.m^{-3}$
dS_1	infinitesimal surface element of S_1	m^2
dS_2	infinitesimal surface element of S_2	m^2
θ_1	angle between the normal vector to dS_1 and the line formed by the gravity center of dS_1 and the gravity center of dS_2	rad
θ_2	angle between the normal vector to dS_2 and the line formed by the gravity center of dS_2 and the gravity center of dS_1	rad
r	distance between the gravity center of dS_2 and the gravity center of dS_1	m

Introduction

I. Context 1

A. Position at the technological research institute SystemX 1

B. Industrial challenges 3

II. Research problem 4

III. Structure of the dissertation 5

The general context of the research presented in this manuscript addresses the task of evaluating physical architectures during the conceptual design phase.

The main objective is to assess 3D system architectures under thermal constraints, following a Systems Engineering approach to manage design data and the interactions of the actors in the conceptual design process.

I. Context

A. Position at the technological research institute SystemX

Although simulation-based approaches (Graignic, et al., 2013) have proven to be effective in meeting these design challenges, producing simulations rapidly and in accordance with the designer's intention is far from obvious. Experts in each discipline tend to refine their simulation models to the best of their knowledge. A paradigmatic change is essential to reduce over-quality and improve agility: the expected properties of the simulation models must be specified upstream, before development, while the development of the simulation models, seen as a system-of-interest by itself, has to be re-engineered according to systems engineering practices.

Simulation has brought huge gains to engineering design, including early risk reduction, agility, reduced costs. It has been so successful that simulation teams have grown into entire departments. Simulations are now sometimes performed by engineers without understanding the goal for which the simulation model has been developed. Despite the economic pressure placed on engineers, a "universal" simulation model does not yet exist. Engineers are aware of this and tune existing simulation models to adapt them to a given scenario. Unfortunately, tuning is rarely appropriate. When out of context, simulation models are more a problem than a solution.

Therefore, the implementation of tools at the "system architecture" level based on behavioral models of different natures is a key point, and still represents a hurdle for effectively addressing the performance analyses and multidisciplinary optimizations necessary for systems design. Therefore, the Multi-Disciplinary Simulation and Engineering (SIM) project at the Technological SystemX research institute proposes to situate its investigation at the very beginning of the design (i.e. conceptual design). The aim of the SIM project is to imagine the tools of **"simulation architects"** and multidisciplinary collaboration methods "based on models" for engineering future vehicles (hybrid cars, "more electric aircraft"), to meet the challenge of environmental issues, energy efficiency and passenger comfort and safety. SIM will be situated at the collaborative interface between the system architect (who provides system architecture solutions) and the simulation teams (who perform the simulation to support design decisions) and will adapt the physical architecture to provide a simulation architecture, by providing a description of the simulation to be performed for each simulation element. The whole process of the SIM project is given in the annex on the SIM process, based on three main phases: initialization, where system architects and simulation architects create the architecture; collaboration, which concerns the preparation and visualization of the simulation; and, finally, capitalization, which includes the verification and validation phases.

To address these issues, the "Multidisciplinary Simulation and Engineering" project (abbreviated as SIM in French) has gathered partners from industry and research co-located at the IRT-SystemX for three years of Research & Development. The project has explored all aspects of simulation contributing to decision-making during the design of complex systems. The project started in April 2013 and ended in May 2016.

The SIM project addresses the tools and methodologies needed to support new simulation architect tasks (Figure 1). Below is a synthesis of the corresponding works developed within the SIM project.

Chen et al. proposed an architecture framework (Chen, et al., 2014) , called *Vehicle Architecture Framework (VAF)* which integrates all the architecture elements required for system architecture and simulation architecture. In order to help system architects to define the physical architecture, Ben Hamida et al. proposed a Design-to-Value framework, called *ValYou* (Ben Hamida, et al., 2016) , which identifies added values for customers (value elicitation) and design the value proposal of a system or service from various alternative systems. Finally, to facilitate the integration of simulation architects in the industrial process, Roa Castro et al. proposed a Collaborative Engineering Design Organizational System (*CEDOSy*) (Roa Castro, et al., 2015) which simulates the interactions between the different actors of conceptual design, based on the game theory.

The tools developed include the concept of Model Identity Card (MIC) proposed by Sirin et al. (Sirin, 2015) for specifying a SiMo. Model Identity Cards (MIC) describe a formalism associated with a SiMo as a black box (NASA-STD-7009, 2008). It is intended to be a support for negotiations on SiMo. MIC formalism identifies the numerical solving methods, the hardware and software required to run the simulation, all the "pedigree" elements (Sirin, 2015) justifying a level of confidence in the simulations, as well as a list of parameters and (input or output) variables. MICs are not executables, they only specify the expected SiMo (to-be) or describe a given SiMo (as-is). They will be useful for assessing the validity of SiMo for any situation not explicitly tested and for determining its suitability/compatibility for reuse with other simulations in a distributed simulation process. The MIC concept was extended by Fontaine et al. (Fontaine & Hammami, 2016) by four formalisms:

- *MIC2MO*, which allows automatically converting an MIC into a Modelica modeling skeleton;

- *MIC2V* formalism, which allows verifying temporal-based requirements regarding the simulation results associated with the equation enriched Modelica skeleton model;

- *CompMICs*, which compares two MICs and gives "a distance" related to the gap between them.

- *MO2MIC,* used to transform a Modelica model into an MIC.

Finally, the works developed in this PhD thesis (*SAMOS* framework and the corresponding tool *3D thermal* Sketcher) aim at supporting system architects by providing simulation architects a means of assessing the physical 3D architecture of a concept by taking into account predefined thermal requirements.

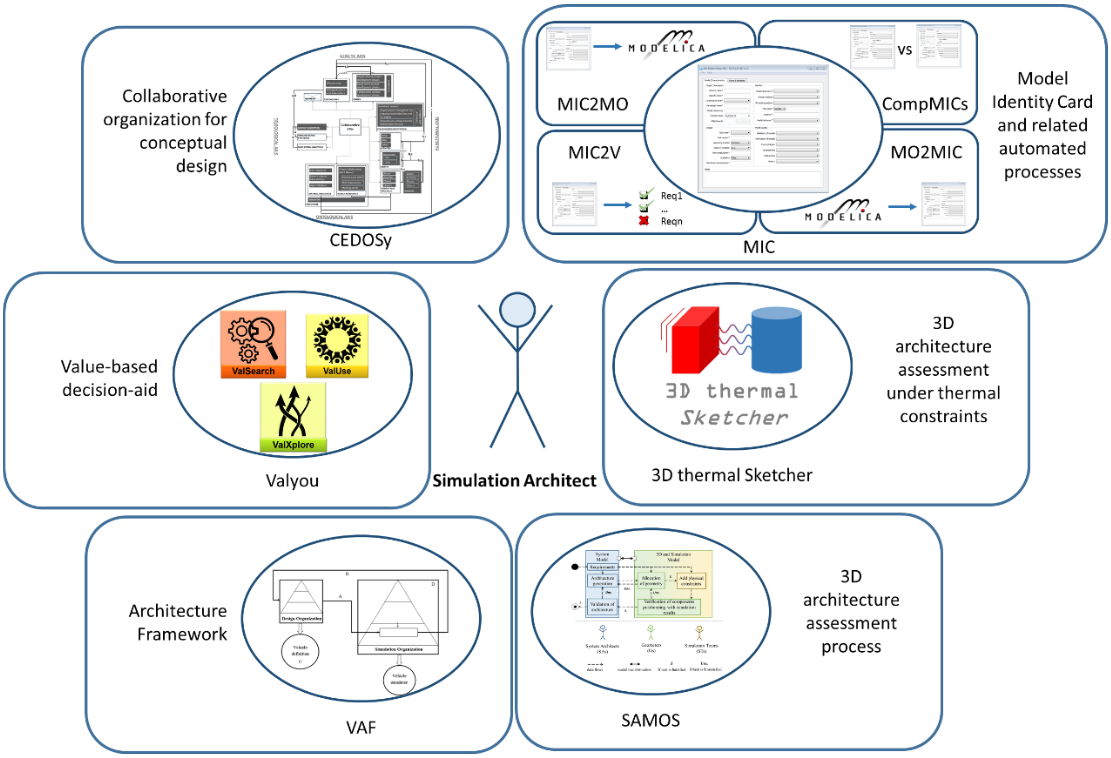


Figure 1: Methodologies, tools and processes developed in the SIM project to address simulation architect assignments.

B. *Industrial challenges*

It is commonly acknowledged that the decisions in the early stages of design (including conceptual design) impact around 80% of overall system life costs (Perry 2005). Moreover, due to fierce competition between companies, the main objectives of industry at present are to ensure good product quality while minimizing cost and design time. Therefore, the technical answers to the System Architect's questions have to be provided in the early design phase, by multidisciplinary teams (thermal, EMC, vibrations, electrical, etc.), in order to facilitate the selection of the "best" solution architecture. Indeed, as the early phase of the development process consists in exploring different architecture solutions to address various objectives like performance, innovation, costs, sustainability, etc., the choices made during this phase will affect the subsequent detailed design phases and can drastically impact the cost and even the success of a project.

Moreover, many industries are facing difficult challenges like energy transition and technological breakthroughs (Internet of Things, autonomous systems), to design new, revolutionary complex systems. Regulations impose ever-lower CO_2 emissions, forcing industries to develop new technologies, for example, electric cars in the automotive sector, and green taxiing for airplane ground maneuvers. Such technological changes imply the complete modification of system architecture, which can represent a risk for companies with no experience with this new architecture.

In parallel, the increasing complex designs of such systems results from complex interactions and relationships between their multi-domain subsystems and between the corresponding disciplines. This also includes mechatronic products, which comprise a large number of multi-physical (thermal, electromagnetic interferences and vibration) interactions between their multi-domain components. Such technological innovations lead engineers and architects to reconsider their classical single-disciplinary approaches and to turn towards a cross and multidisciplinary approach like System Engineering (SE).

However, communication between the various technical services remains quite difficult because the actors in design work with their specific tools. This leads to numerous and long iterations between the different actors, further complicated by the lack of data consistency between the stakeholders, some of which use non-interoperable tools. This makes the implementation of numerical simulations arduous, as they also have to take into account all the domains involved, including the methods and tools dedicated to the different fields in the industrial sector concerned.

II. Research problem

Architecture selection during the conceptual design is becoming a crucial step, notably with the considerable need to integrate new technologies in current systems. These systems, whose development, testing, and application require the deployment of systems engineering, are defined by the three following characteristics:

(i) they are engineered products and hence satisfy a specified need;
(ii) they consist of diverse components that have intricate relationships with each other and are hence multidisciplinary and relatively complex;
(iii) and, finally, they use advanced technology in some ways central to the performance of their primary functions and hence involve development risks and often relatively high costs (Kossiakoff, et al., 2011).

These new IT-based technologies, which often have to be integrated in mono-domain components, are now being used in certain mechatronic systems. The design of such multi-domain systems is a difficult and complex process, notably due to the increasing number of multi-domain components to be included in a small volume, in which multi-physical couplings between the components are necessary. It is also because the complexity of these multi-domain systems requires heterogeneous resources, be they human or technical. In parallel, the design of such systems also requires multi-domain simulations to evaluate the performances and multi-physical couplings of candidate physical architectures, meaning that various multi-domain or domain specific simulation environments can be used. However, these tools are typically used once the physical architecture of a concept has been chosen, without first ensuring that the corresponding 3D architecture of this concept will not generate unwanted physical interactions between the components of such systems.

Therefore, the objective of this research work is to answer the following question:

"**How can the physical architectures of these complex systems under multi-physical constraints in the conceptual design phase be evaluated, in order to limit the risk of multi-physical couplings in later design phases and subsequently a dramatic increase in design costs and time**?"

Moreover, design interaction (between data and/or humans) management will also be tackled as this research topic addresses the initialization phase of the SIM process (Appendix 1 Figure 154), in which the number of iterations between the System Architect and the Simulation Architect needed to converge to an acceptable architecture have to be reduced.

Finally, as multi-physical constraints can be various, we focus specifically on the assessment of thermal behavior, although the approach developed will be generalizable to other physics.

III. Structure of the dissertation

Chapter 1, titled "Scientific context", describes the System Engineering context and the Model Based System Engineering approaches in the conceptual design. Then, related existing works addressing the interaction management for the 3D architecture assessment under physical/thermal constraints during the conceptual design phase are examined. Thus this state of the art first focuses on the introduction of geometry and physical modeling for the conceptual design in an MBSE context. Secondly, existing methods and computer-aided-design tools for evaluating 3D spatial architectures are presented. Finally, several approaches for managing interactions between actors and data during the conceptual design phase are outlined. The analysis of these various approaches is provided in the second chapter.

Chapter 2, titled "Approach selection process", introduces the requirements derived from the research issue in this PhD work. These requirements, also defined in SysML, will serve as the basis of all the analyses and developments described throughout the manuscript. These requirements are first used as criteria for choosing the collaboration structure and the associated technical solution. Then, considering the solution selected, three approaches for the system architecture assessment during the conceptual design phase are studied regarding their capacity to facilitate collaboration between the actors addressed by our problematics (i.e. system architects, 3D architects, and simulation teams). Finally, after selecting the most adapted approach, the global structure of the framework is described through the definition of its corresponding demonstrator implementation environment, including the validation of the language used for the System modeling and the selection of the model transformation process.

Chapter 3 is titled "Geometrical modeling", in accordance with the fundamental place of geometry in the Spatial Architecture, based on Multi-physics and Organization of Systems (SAMOS) framework. Indeed, geometrical modeling is an important part of the development of the SAMOS platform, both for defining the 3D architecture, and to support thermal modeling. Therefore the related requirements are then derived from those established in chapter 2 to clearly define the needs. Then a state of the art presents the main current geometrical approaches, including geometry and topology modeling. After analyzing them regarding the previous specific requirements, the choice of the geometrical modeling for SAMOS is described. Then, the metamodel of the SysML extension, named Geometrical Extension Related to the TTRS Reference for a Unified Design (GERTRUDe) is presented. Finally, several strategies for managing the transformation of the geometrical information model from GERTRUDe to the 3D CAD tool are compared, in order to choose that which will be applied in SAMOS.

Chapter 4, titled "Thermal modeling", starts with the expression of the need related to the choice of thermal modeling. It consists in deriving the requirements proposed in Chapter 2 from the thermal modeling viewpoint. Then, several hypotheses and reminders of thermal analysis are provided, before a description is given of the three main thermal modeling alternatives: analytical analysis, thermal resistance analysis, and the finite element analysis/volume method. Each thermal model is then assessed using the criteria defined from the previous requirements. A description of the thermal model chosen and its application and adaptation to support the SAMOS framework is provided. Several proposals related to the application of the TTRS theory for the three heat transfer modes are presented. Finally, the metamodel of the Thermal Related SysML Extension (TheReSE) developed is described.

Chapter 5, titled "Implementation of the Thermal 3D Sketcher", presents the Thermal 3D Sketcher, which is the tool developed to demonstrate the ability of the approaches proposed for use in an industrial context. We first choose the tools used to develop the thermal 3D Sketcher. Then, the algorithms used to develop the thermal 3D sketchers, including the geometrical and thermal model transformation processes, are provided. Lastly, the various developments, including the Graphical User Interface (GUI), in the three environments (SysML, 3D, and Simulation) are detailed.

Finally, **Chapter 6,** titled "Scenario validation", proposes a validation of the SAMOS framework developed through various case studies. The five case studies, including four industrial ones (aircraft bay, system conveyer, helicopter bay, electric power train), and an academic scenario (developed to support an engineering training course) focus on different parts of the thermal 3D Sketcher. The System conveyer case study is aimed at verifying the geometrical model. The aircraft cab case study is a description of the TheReSE model. The helicopter bay and the electrical power train for an electrical bus serve to verify the thermal requirements for several alternative 3D architectures. The last case study consists in testing the Thermal 3D sketcher during an engineering training course to help students select the best physical architecture for an electric bicycle, based on the thermal simulation results.

The manuscript ends with conclusions on, and perspectives for, the approaches developed.

Chapter 1 – Scientific context

I. Systems Engineering context 9

II. Related works 12

A. Geometrical and physical modeling integration in an MBSE context for conceptual design 12

1. UML profiles/SysML extensions 12

2. Integration of geometry and thermal modeling in the early design phases 13

B. 3D physical architecture assessment 14

1. Metrics 15

2. Simulation modeling 16

C. Interaction management during the conceptual design phase 17

1. Detailed conceptual design steps 17

2. Need to integrate 3D visualization in the conceptual design 17

3. Human interactions during the conceptual design phase 18

4. Interactions between conceptual design actors 20

5. Design data interactions 25

III. Conclusions 29

Fierce competition between industrial companies is leading to evermore complex systems, lower costs and shorter design time. Moreover, this increasing complexity of systems implies a large number of interactions between different disciplines that have to be taken into account to ensure the consistency and traceability of data and models. This is the reason why the MBSE (Model-Based Systems Engineering) approach has been introduced in the design life cycle of companies in order meet these objectives.

In addition, a dramatic increase of design costs and time usually takes place during the embodiment or detailed design phases. This issue is, as a rule, related to the numerous and long iterations between the different disciplines in various technical services. Indeed, their respective simulations of physical behavior are not always consistent, due to the lack of data uniformity between simulation teams and the difficulty represented by design collaborations using different non-interoperable discipline tools.

To deal with these issues, a preliminary assessment of architectures under geometrical and physical constraints could reduce the risk of late changes during further design phases, and thus design time. Therefore, in this chapter we present related existing works on MBSE approaches. Studies on the introduction of geometry and physical modeling for conceptual design in an MBSE context will then be presented, before a description is given of existing methods and computer-aided-design tools used to evaluate 3D spatial architectures. Finally, several approaches for managing interactions between actors and data during the conceptual design phase will be summarized.

I. Systems Engineering context

Since the development of new modern systems is strongly driven by technological change, we must add one more characteristic to a system requiring systems engineering, namely, that some of its key elements use advanced technology (Kossiakoff, et al., 2011). The concept behind system engineering is to ensure the integrity of the overall product by imposing system-level technical specifications on all the teams and actively monitoring the development of the product as a system. The current deployment of systems engineering processes in complex systems and large projects is aimed at supporting the different engineering activities throughout the system lifecycle: requirements analysis, architectural definition, detailed design, verification and validation. The description of the process from the angle of Systems Engineering is given in Figure 2. After analyzing customers' requirements and generating the requirements specification (which relates to how well the system will work in its intended environment), the second step, "Functional Analysis/Allocation" consists in transforming previous requirements (functional, performance, interface and others) into a consistent description of system functions. Functions are then arranged in logical sequences, breaking down higher-level functions into lower-level ones and allocating performance requirements from higher- to lower-level functions. This description is often called the functional architecture of the system. This initial architecture with ever-increasing levels of details can be structured by functional partitioning which involves grouping functions that logically fit with the components likely to be used, in order to minimize functional interfaces. Finally, the next step, "Design Synthesis", is a creative activity that develops a physical architecture capable of performing the previous required functions within the limits of the specified performance parameters. Design synthesis includes trade-offs to select the best among the architectures proposed as there may be several potential satisfactory physical architectures.

These activities are commonly managed by a group of system engineers including system architects. This group "is typically established to help refine voice-of-the-customer data to system-level specifications and to work directly with the component development teams to ensure that these specifications are met. System architects are also often responsible for maintaining and updating central files of component technical specifications and design decisions" (McCord, 1993).

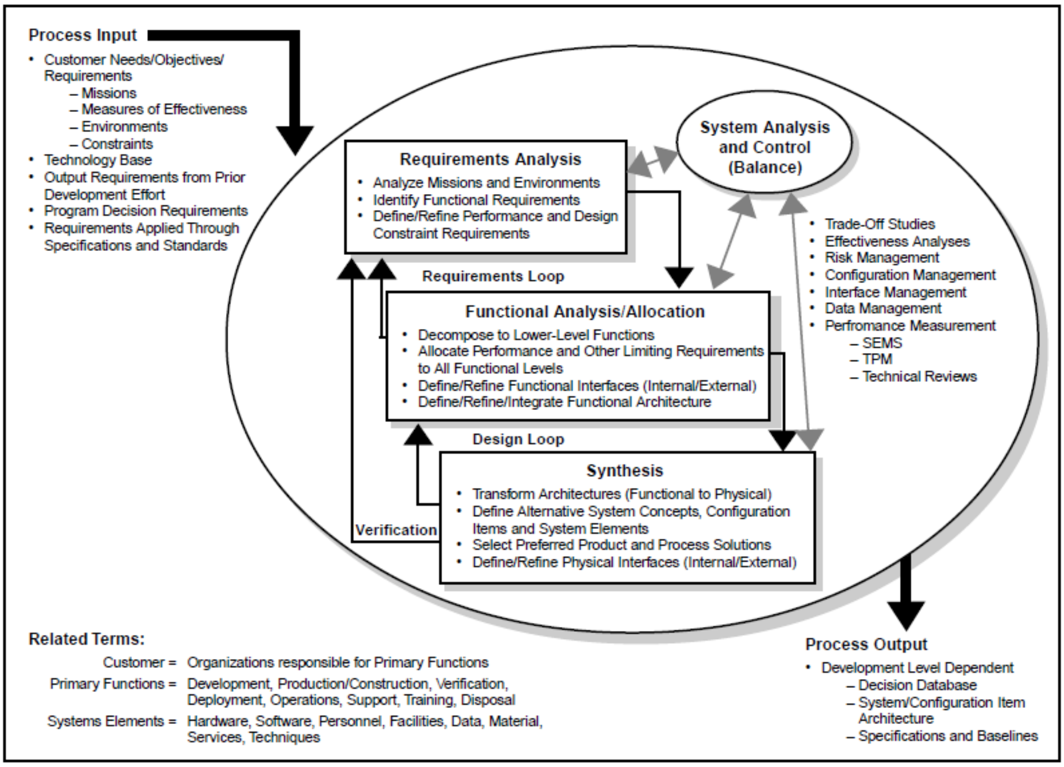


Figure 2: System engineering design cycle (DODSMC, 2001)

The systematic and comprehensive use of modeling and simulation techniques to perform these engineering activities is commonly referred to as "Model Based Systems Engineering" (MBSE). By replacing paper-documents by numerical models and data, the usage of Model Based System Engineering (MBSE) is increasingly widespread in industrial processes to improve reutilization and collaboration between different actors. These practices aim at reducing the design data inconsistencies of increasingly complex systems. Often used by system architects, MBSE efficiently encourages communication and collaboration between the experts of the different disciplines involved during the different design stages. The integration of various multi-disciplinary points of view in numerical models then helps decision trade-offs. In conclusion, MBSE approaches allow not only increasing collaboration and consistency successively between teams and models, but also improving traceability with requirements and promoting the reuse of the models.

In addition to Systems Engineering approaches, system architects need a language which allows them to specify all the requirements and system architectures, whatever the discipline or technical team they address. This aspect is essential for the MBSE approach, and is usually fulfilled by SysML (Systems Modeling Language) language. Indeed, as SysML supports the generation of complete and consistent requirements, and system architectures, it is mainly used for conceptual design. This shared design language addresses the design of multi-domain complex systems. It improves common understanding of models within the collaborative engineering design of a product, and ensures the traceability of the design process. SysML is defined as an extension of a subset of UML (Unified Modeling Language), using a profiling mechanism, for systems engineering applications. UML language is a formal object language, standardized in 1997 by the OMG (Object Management Group, 2011), and defined by a metamodel (specification of object oriented programming language) which gives it a major advantage, since it does not depend on a programming language. UML was extended to Systems Engineering applications, with the emergence of SysML initiated by the International Council on Systems Engineering (INCOSE) to support MBSE and now supported by an OMG specification since 2006 (Object Management Group, 2006). SysML provides considerable semantic richness, as it allows representing different views of the system thanks to different diagrams (described in Figure 3): requirements, functionalities, structures, processes, use cases, and more.

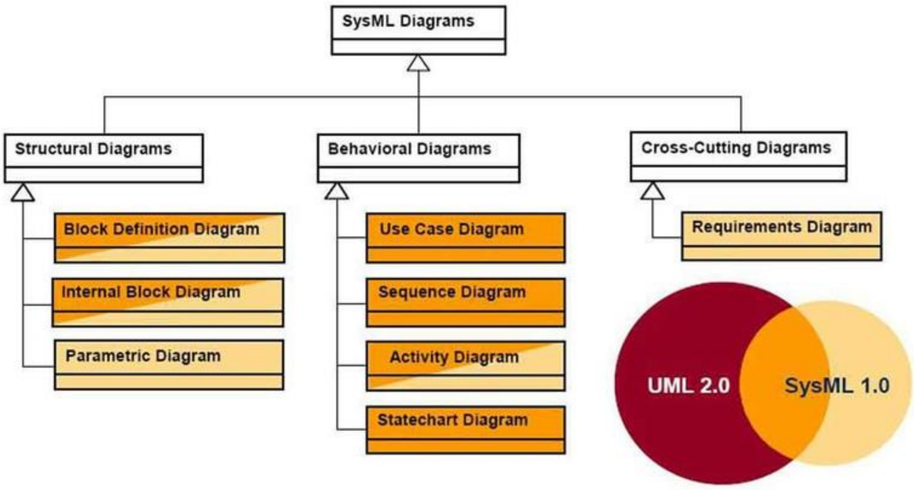


Figure 3: SysML Diagrams (Object Management Group, 2006)

As SysML does not propose any specific methodology to ensure system modeling consistency, Mhenni et al. proposed an MBSE design methodology based on SysML, which describes all the steps required to consistently generate the physical architecture of a concept according to the customer's needs (Mhenni, et al., 2014). In the first stage (named "Black box" analysis), the system is specified but not described (the aim is to describe what the system has to do): the description is obtained using an external point of view. This black box starts from the customer's requirements to define the global mission, the system life cycle, its operating modes, use cases, and functional scenarios in order to provide all the derived requirements that will then allow defining the functional, logical and physical architecture of the system during the "white box" stage, and the corresponding allocation and traceability links relating to the derived requirements. The Black box corresponds to the clarification of the task step, whereas the "White box" analysis corresponds to the conceptual design, i.e. to the description of the system's structure.

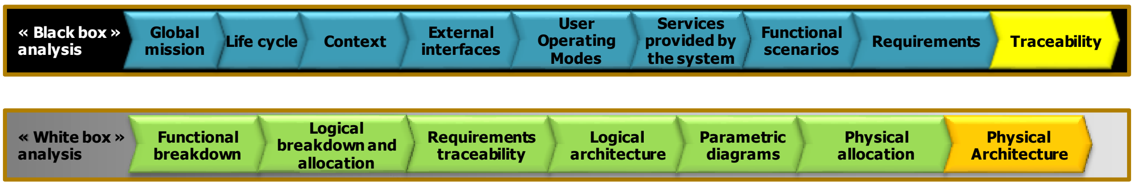


Figure 4: MBSE approach (Mhenni, et al., 2014).

Thus, MBSE approaches allow not only increasing collaboration and consistency successively between teams and models, but also improving traceability with requirements and promoting the reuse of the models.

Finally, MBSE approaches are suitable for supporting design interactions whether between the different design actors or for ensuring data consistency and traceability. Considering the conceptual design phase, MBSE will help to formalize and implement a means to assess the alternative concept 3D architecture under geometrical multi-physical constraints.

II. Related works

Firstly, we will give an overview of existing works related to the introduction of geometry and physical modeling for the conceptual design in an MBSE context, before presenting existing methods and computer-aided-design tools to evaluate 3D spatial architectures. Finally, a few approaches for managing actors and then data interactions during the conceptual design phase will be summarized.

A. Geometrical and physical modeling integration in an MBSE context for conceptual design

1. UML profiles/SysML extensions

According to the System Engineering (SE) approach (Figure 2) and its MBSE related methodologies mainly based on SysML (Ooshima & Masukata, 2013) (Estefan, 2007) (Andrianarison & Piques, 2010) (Holt & Perry, 2008), system-level modeling first aims at analyzing customers' needs, in order to derive them into engineering/technical requirements. Regarding this step, geometrical and physical information should be specified (taken into account in the system analysis process to generate physical architecture alternatives) and traced when the system design is finalized, in order to verify that all the derived requirements have been fulfilled. A common approach is to propose several UML profiles or SysML extensions to enrich system modeling with elements needed by the simulation teams. A profile in UML provides a generic extension mechanism to customize UML models for particular domains and platforms. Extension mechanisms are used to refine the standard semantics of a profile in a strictly additive manner, preventing them from being contradictory (Friendenthal, et al., 2009). Profiles define stereotypes, tag definitions, and constraints which are applied to specific model elements (metaclass), like Classes, Attributes, Operations, and Activities. A metaclass (Friendenthal, et al., 2009) describes the individual concepts of languages. A Profile is a consistent collection of such extensions that collectively customize UML for a particular domain. Stereotypes allow specializing existing modeling elements (metaclass), to give them specific properties that are suitable for a particular problem domain or otherwise specialized usage.

Some authors have already worked with the MBSE approach to introduce the geometrical point of view in the early stages of design. Baysal et al. proposed a method of geometrical modeling and positioning in UML (Unified Modeling Language), for the tolerance analysis (Baysal, et al., 2005). Nonetheless, the positioning considered is not relative and does not directly integrate any constraints, which makes it rather difficult for designers to calculate the positioning of each part. Moreover, Albers et al. proposed the Contact & Channel-Approach (Albers & Zingel, 2011) to build Contact & Channel-Models (Albers, et al., 2010) through a SysML extension (Albers & Zingel, 2013). This modeling defines Working Surface Pairs as interface surfaces that are connected by physical components or volumes of liquids, gases or spaces, named Channel and Support Structures. As their objective is to represent engineering artefacts that take into account physical flows between components, this geometry modeling is based only on the working/interacting surfaces, and does not allow generating either the whole volume of components or their relative positioning constraints. Furthermore, Bohnke et al. (Bohnke, et al., 2009) proposed a UML profile that defines the 3D geometry of components, but without managing their assembly constraints: they import in UML geometries resulting from CATIA V5 designs, but this method does not create geometry directly in UML, and does not allow the System Architect to specify any geometrical parameter specifications; they merely generate component volumes by linking different sections that have been represented by points. This method is in fact more adapted to complex detailed geometries and is thus not useful for conceptual design when the geometry of the components is not clearly specified. Finally, Warniez et al.

proposed a geometrical SysML extension that includes a library of simplified geometrical volumes to define physical integration metrics, but this extension does not manage the relative positioning between components or the addition of new geometry (Warniez, et al., 2014).

Many profiles have already been developed to ensure a consistent link between SysML/UML, simulation and 3D models,. The profiles that address the link between SysML/UML and simulation models are often developed by transforming the semantics of a simulation tool into the SysML or UML semantics. Among existing profiles linked with simulation, the most well-known are the SysML4Modelica extension (Paredis, et al., 2010) (Reichwein, et al., 2012) and the ModelicaML UML profile (Pop, et al., 2007; Schamai, et al., 2009). The link between both SysML and Modelica languages was interesting because the semantics of these two languages are very close (Paredis, et al., 2010) and both of them can be used for acausal modeling. ModelicaML was the first UML profile to be developed. It provides Modelica elements in UML, but without implementing the modeling code so the user must write the corresponding Modelica code in the UML model, in order to automatically generate the code necessary to perform a simulation in a Modelica environment. This profile provides only a UML environment to develop Modelica modeling, but it is not adapted to System Architects who are usually not in charge of multi-physical modeling in the Modelica language. It is for this reason that Schamai et al. proposed a SysML extension named SysML4Modelica (Schamai, et al., 2009). This SysML extension is easier to use for Systems Architects, because Modelica modeling in SysML is designed with SysML artefacts (internal block and parametric diagrams). The Modelica code is automatically generated from a diagram in SysML and does not have to be written directly in SysML. However, SysML4Modelica does not take into account the definition of the component geometries needed by Simulation Teams, although this is necessary to generate a 3D architecture of components whose multi-physical simulation results meet the System Architects' requirements.

Although UML profiles and SysML extensions seem to be a good way to enrich System Modeling with specific thematic artefacts, the stereotyped elements developed from the existing previous profiles only allow representing or tracing the physical modeling elements in the System model from certain simulation languages or tools. However, these profiles do not include any stereotypes for physics based on geometrical considerations. Furthermore, many of these models propose an additional system view without being linked with the other modeling artefacts of standard UML/SysML models. Finally, they are usually used to trace simulation data but not to specify them.

2. Integration of geometry and thermal modeling in the early design phases

A survey-based study (Römer, et al., 2001) showed that most designers need sketches before using 3D CAD tools, since sketches are useful to have a preliminary vision of the system's geometry (Do, et al., 2000). This study also shows that the preliminary use of sketches improves later modeling with 3D CAD (Pache & Lindemann, 2003). Although 3D CAD tools are used extensively at detailed design stages, the shortcomings of such tools for the early stage of development of mechanical products have been well documented in the literature (Tor, et al., 2002) (Cheutet, 2006). Whatever the case, some 3D tools have been developed for conceptual design, like the Open VSP software implemented by Hahn (Hahn, 2010) for NASA, which is used to design aircraft during the conceptual design phase. The author defined geometry graphically by juxtaposing shapes with several sections previously generated from points. The use of this Open VSP software is very easy, but it provides only a graphical representation of an aircraft's exterior surfaces, without enabling designers to design and visualize other internal components. This lack is critical, because some surfaces that are not dedicated to the aircraft application must still be considered; moreover the geometry of the internal components of a system is essential when taking assembly into account. Finally, this tool does not implement either dimensional parameters or assembly constraints. This open source software is therefore more adapted for 3D sketching designers who want to imagine the shape design of their system than for Systems Architects who need to evaluate component pre-positioning of various alternative architectures.

Considering virtual reality (VR) approaches, they can also address geometric considerations for conceptual design. Ye et al. claimed that as the corresponding technologies are very intuitive for supporting the visualization of a 3D CAD product, they are usually appreciated by designers, and thus can easily be used to build 3D components in conceptual design (Ye, et al., 2006). When considering the SE approach, Zwolinski et al. used VR to create a link between the 3D components modeling and the functional definition of the system, in order to provide a 3D functional model (Zwolinski, et al., 2007). Although this approach was highly innovative at the time, this VR-based functional approach did not include any geometrical constraints between elements. In this approach, the geometry of components was not directly defined on the functional level, since the 3D CAD file was attached to functions only when created. It also implied that it is not possible to create geometry requirements. Finally, the interest of this method mainly relied on component reuse, but not on the design of a new kind of product (geometry). In addition, Ng et al. also proposed to integrate geometry using an FBS (Functional, Behavior, Structure) approach with VR, which gave a description of the functional behavior of the system, with a 3D geometry representation (Ng, et al., 2014). However, the global structure of the geometry was not stored in the model, although it was possible to include design parameters and assembly constraints. But the assembly constraints defined were only limited to contact and coincidence, and it was thus impossible, for example, to use a distance constraint in order to define parallelism between two plans. Finally, the weakness of FBS modeling (Chase & Liew, 2001) and other previous VR-based 3D models is that they do not provide any formal link with any requirements, and thus with any geometrical requirements.

In order to evaluate system performances relating to a Measure of Performances (MoP), certain geometrical relationships like physical laws including geometrical shape factors are usually required for preliminary behavioral simulations of physical architecture alternatives (Lightsey, 2001).

Finally, although many research works over the last fifteen years have addressed the integration of geometrical data for conceptual design by focusing on their importance for tolerancing (Mao, et al., 2008), process planning (Feng, et al., 1999)and assembly (Sodhi & Turner, 1994) (Hsu & Woon, 1998), few studies have dealt with their implementation in the context of system engineering, especially to evaluate design concept architecture.

B. 3D physical architecture assessment

During the predesign phase, customer specifications are translated into technical specifications through functional analysis, and system architectures (functional, logical and physical) are generated. These activities are generally performed by the System Architect, who also has to evaluate the candidate architectures in order to select that which best corresponds to the customers' specifications before engaging in more detailed and often expensive analyses. This architecture analysis usually requires performing several preliminary physical behavior simulations, to quantitatively determine the design parameters that will meet performance requirements. Finally, as the simplest assessment of any physical behavior relies on the orientation and distance between the components or even on dimensional data, designers usually need to consider geometry as soon as possible in the design life cycle (notably during the preliminary design phase) in order to evaluate physical interactions. Incidentally, Clayton et al. illustrated this narrow link between requirements, functions, 3D architecture and physical behavior (Clayton, et al., 1996).

To evaluate these 3D physical architectures, two main approaches are used: metrics or simulation.

1. Metrics

The usage of metrics for the evaluation of 3D architecture in conceptual design can be performed using different models. Warniez et al. (Warniez, et al., 2014) proposed using metrics in order to evaluate the physical integration of mechatronic systems. Their metrics based on physical architecture alternatives in SysML address only component geometry, such as compactness or a convex hull, in order to decrease the space between components, but do not explicitly consider either the relative positions of components or their physical behavior.

Other approaches have been developed to address 3D architecture during the conceptual design phase, notably to tackle assembly design objectives. For example, Simpson et al. (Simpson, et al., 1995) proposed using DFA (Design For Assembly) analysis in order to develop a cost metric depending on 3D architecture, based on criteria such as accessibility, ease of handling, and complexity of assembly. The DFA technique is a method based on a product architecture, which analyzes the design of a product in order to improve its ease of assembly. This method was implemented in SysML by Demoly et al. (Demoly, et al., 2010) for use during the conceptual design phase, but this model does not include the component geometries, only their assembly constraints. Furthermore, some authors also used a DFA index for concept design, to evaluate the "fitness" of a system for assembly (Gupta & Okudan, 2008) (Hsu, et al., 1998). Concept design is therefore analyzed through certain extracted features of design concepts or models, like dimensions, symmetry, predefined shapes (round object, flat surface, etc.), weight etc.; nonetheless the assembly between different subsystems is assessed in relation to either its ease to automate assembly or its assembly time, and not specifically through the fulfillment of geometrical constraints.

In parallel, Moullec et al. (Moullec, et al., 2013) proposed to take into account the geometry and relative positioning of each component during conceptual design to validate certain performance metrics depending on the placement of components. They proposed to automatically generate the 3D architecture using Bayesian Networks and Constraint Satisfaction Problem approaches, starting from a functional architecture. Their approach is interesting regarding the automation of 3D architecture based on constraints during the conceptual design process, and notably for segregation rules. However, several difficulties exist: first, the geometry of the component is limited to a parallelepiped rectangle; moreover, the orientation of the components is limited to 6 positions. Finally, the Performance constraint is difficult to evaluate. All the components are automatically dimensioned and placed with an integrated approach based on Bayesian Networks and the Constraint Satisfaction Problem to evaluate the resulting architectures by integrating component placement optimization. Conceptual design indeed requires a declarative approach, which allows designers to specify the geometrical data which they already have, so that they can progressively complete them with further new constraints. However, this approach requires knowing all the geometrical constraint specifications, which are rarely fully available in the first stages of design, and proposes a geometrical model limited to box shapes.

2. Simulation modeling

To perform 3D physical architecture assessment, it is possible to simulate the physical behavior using the geometrical parameters transmitted from components.

Qin et al. (Qin, et al., 2003) proposed a web-based framework to share and simulate the dynamic behavior of a 3D conceptual architecture. The components of this architecture had a simplified geometrical representation, and the position of each geometrical component was calculated using several relative positioning constraints. Then the relative positioning and orientation parameters which were not imposed by geometrical constraints were calculated by an external behavior simulation tool. This tool is interesting since it proposes the collaborative verification of the dynamic positioning of components subjected to physical stress during the conceptual design. However, the main drawbacks are that all the geometrical and physical modeling has to be programmed in Javascript, and the physical simulation is limited to kinematics movements.

Komoto and Tomiyama (Komoto & Tomiyama, 2012) proposed a framework called System Architecting CAD (SA-CAD), that included a geometric modeler for visualization, based on an FBS framework. They propose to add the 3D modeling step after functional modeling, and to verify the systems requirements through a combination of functional and geometrical parameters. They proposed, for example, to roughly assess the temperature of a component by way of its geometry, its position, its power emission, its mass, and the heat capacity of its constitutive material. The physical relation is only parametric and there is no differential equation solver. The spatial relations are limited to 11 and the component position is performed manually and determined by coordinates; no spatial constraints solver is implemented. Moreover, all the (spatial and temporal) relations must be specified manually: geometrical and thermal parameters are not automatically linked with equations. Thus the specification time can be very long for a complex system.

Furthermore, Plateaux et al. (Plateaux, et al., 2009) proposed a change of paradigm for behavioral modeling. The common paradigm is "geometry in physics" in which the geometrical parameters are implicit or secondary, but the authors argue that the paradigm "physics in geometry" would be more efficient. To easily enrich a model with numerous multi-physics couplings, physical modeling must be represented in 3D. Usual simulation tools only propose a 2D object model in which 3D geometrical parameters are hidden in the components. The real geometry is only taken into account during simulation. For example, the 2D icon representation of the Modelica objects with positions and dimensions has no geometrical meaning and geometrical data are not coupled to these 2D icons, as illustrated in (Figure 5). With this new 3D paradigm, the geometrical objects will integrate their physical behavior.

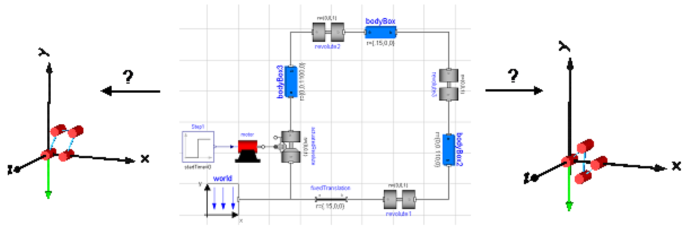


Figure 5: Simulation issue of two 3D alternative interpretations of a 2D iconic 4 bar model (Plateaux, et al., 2009)

Finally, some tools have proposed a 3D view of simplified component volumes (for conceptual design) that encapsulates functions or behaviors: the Logical 3D Architecture of CATIA Systems (Dassault Systèmes) brings 3D to logical systems for space reservation and pathway connection, but does not meet all the SE principles implemented in the SysML language.

C. *Interaction management during the conceptual design phase*

1. Detailed conceptual design steps

The conceptual design phase is also sometimes called Concept Development (Kossiakoff, et al., 2011). Systems engineering establishes the needs of the system, explores feasible concepts, and selects a preferred system concept. The concept development stage may be further subdivided into three phases (Figure 6):

1. Needs analysis: this defines and validates the needs for a new system, demonstrates its feasibility, and generates the system operational requirements. It is what Mhenni et al. (Mhenni, et al., 2014) have called "derived requirements" at the end of the black box phase of their methodology.
2. Concept exploration: this explores feasible concepts and defines the functional performance requirements and the physical architecture alternatives of the concepts.
3. Concept definition: this examines alternative concept architectures, selects the suitable concept on the basis of performance, cost, schedule and risk, and defines the system specific/induced requirements.

	Concept development		
Step	Needs analysis	Concept exploration	Concept definition
Requirements analysis	Analyze needs	Analyze operational requirements	Analyze performance requirements
Functional definition	Define system objectives	Define subsystem functions	Develop functional architecture component functions
Physical definition	Define system capabilities; visualize subsystems, ID technology	Define system concepts, visualize components	Develop physical architecture components
Design validation	Validate needs and feasibility	Validate operational requirements	Evaluate system capabilities

Figure 6: Main activities of the conceptual design phase (Kossiakoff, et al., 2011)

2. Need to integrate 3D visualization in the conceptual design

More and more studies underline the need to integrate a graphic description/3D visualization at the early stages of design (Shah & Rogers, 1988).

For example, Shen and Li explained that 3D CAD modeling improves communication between the different actors. Thus different 3D CAD tools such as Solidwork eDrawing, autodesk streamline and Actify SpineFire facilitate collaboration management (Shen, et al., 2008). A 3D CAD model is sometimes more expressive than word-based communication. Bonnema and Houten (Maarten Bonnema & Houten, 2006) demonstrated, through a survey submitted to 15 experienced conceptual designers, the use/need of sketches to meet their objectives for concreteness and to perform tasks. Boujut et al. claimed that modelling in a 3D CAD environment efficiently underpins co-operation between experts during the design phase (Boujut & Laureillard, 2002). Likewise, Li et al. (Li, et al., 2004), Qin et al. (Qin, et al., 2003) proposed a web-based framework to share and simulate the dynamic behavior of a 3D conceptual architecture, in order to

(i) rapidly develop a product model;
(ii) improve understanding of design ideas for all the parties involved;
(iii) facilitate communication so less time is spent in face to face meetings;
(iv) reduce the need to invest in more 3D CAD workstations and software;
(v) use simulation to reduce the number of costly physical prototypes.

The famous Functional Behavior Structure (FBS) framework (Umeda, et al., 1990), including six modeling types, as shown in Figure 7, has been improved by many authors in view to integrating 3D data (Al-Fedaghi, 2016) (Komoto & Tomiyama, 2012).

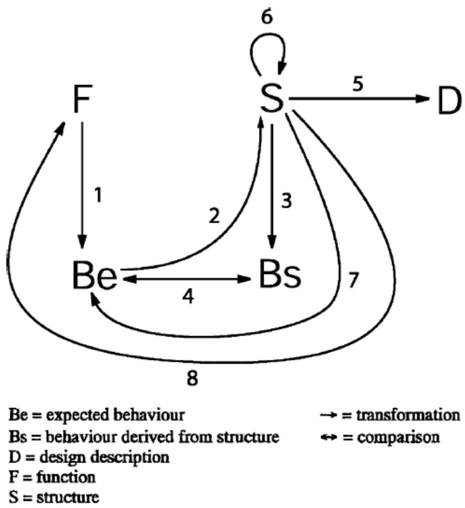


Figure 7: Description of the FBS framework (Gero & Kannengiesser, 2000)

According to Gero and Kannengiesser (Gero & Kannengiesser, 2000), each transition between different models is performed by a specific actor, demonstrating the importance of human interactions in this framework. This human interaction can be generalized for the whole conceptual design.

3. Human interactions during the conceptual design phase

Design, including conceptual design, is a process of negotiating between disciplines (Détienne, et al., 2005). Solutions are therefore not only based on purely technical problem-solving criteria. They also result from compromises between designers: solutions are negotiated. The initial specifications are never complete or without ambiguity: initial problem specifications are not sufficient to define the goal, meaning that the solution and the progressive definition of new constraints are necessary, since the constraints can be explicit or implicit in the viewpoints expressed by the designers. During this process, alternative solutions are proposed and the choice of one solution among the set of proposed solutions is based on assessment via multiple constraints. Designers develop and assess design solutions partly due to their own specific constraints (which reflect their own specific viewpoint) in relation with the specificity of the tasks they perform and their personal backgrounds. Furthermore, the selection and the weighting of constraints evolve through the participants' interactions. In 90% of the cases observed, Detienne et al. (Détienne, et al., 2005) showed that the different assessment modes are used in the following order: Step 1: analytical assessment mode of the current solution; Step 2: if step 1 has not led to a consensus, then a comparative or/and analogical assessment is involved; Step 3: if step 2 has not led to a consensus, then one (or several) authoritative argument(s) of authority is (are) used.

As the conceptual design is the phase of the project where the most ideas must be explored,notably during the concept exploration step, there are two ways of obtaining substantial creativity from the designers: first, by increasing the duration of this phase, or by increasing the number of collaborators. As Starey et al. (Starkey, et al., 2016) showed, creativity is greatest at the very beginning of a project, in accordance with Wang et al. (Wang, et al., 2002) who claimed that the more the design progresses, the less decisions influence the final design, thus increasing the number of collaborators must be the better of the two paths for generating the maximum number of ideas.

The issue of collaboration (working with others with shared goals for which the team attempts to find solutions that are satisfying to all concerned (Kvan, 2000)) in the conceptual design is becoming crucial. New organizations, based on concurrent engineering principles, after many years of experimentation within various companies and industrial domains, still suffer from a lack of efficiency. An increasingly high number of participants from various fields are now clearly involved in the conceptual design process, each of them having their own disciplinary worlds made of representations, languages and tools (Boujut & Laureillard, 2002).

In parallel, according to Wang et al. (Wang, et al., 2002), the conceptual design is the most crucial task in an engineered product development cycle, and it requires Computer-Aided-Design (CAD). However, few applications specific to conceptual design are available to facilitate this complex task. It is often difficult to capture, visualize or communicate between multidisciplinary design teams in an MBSE approach, probably due to imprecise and incomplete knowledge of design requirements and constraints at the early phase of design. This is especially the case when this interdisciplinary task relies on the knowledge and expertise of geographically dispersed people (customers, designers and engineers). As the demands for shorter time-to-market and designing a product right-the-first-time are increasing to keep companies competitive in the customer-centric market, conceptual design must adopt a more pragmatic approach based on collaboration and information technologies.

This situation can explain why the mode of collaboration employed in industry is a loose-coupled design process, as defined by Kvan (Kvan, 2000), who explained that communication between different actors can be performed in two ways: a tightly coupled design process, in which participants work together closely and intensely "to achieve a holistic creative result", by observing and understanding each other's operations; or a loose-coupled design process, in which each participant contributes what they can in a different domain of expertise at moments when they have the knowledge appropriate to the situation.

Moreover, as mentioned by de Micheli (De Micheli, 1993) and Buchenrieder (Buchenrieder, 1993), as a system is a physical unit that can deliver a service (for example, a computer workstation or a manufacturing robot), system level design requires solving mechanical, electrical, and software problems, and then performing codesign activities in a unified approach, notably for system-level specification and simulation environments, soft-prototyping techniques, formal design and verification methods, and high-level synthesis and framework technology.

Finally, in view to reducing their duration, Austin et al. analyzed which collaborative steps of the conceptual design take the longest time, and showed that the search for solution principles and the evaluation and choice of alternatives take up 20% and 10% respectively of conceptual design time (Austin, et al., 2001).

4. Interactions between conceptual design actors

a. Description of the roles of the actors

- System Architects :

System Architects are systems engineers. Whereas systems engineer activities focus on the whole system product, leading and working with many diverse technical team members, following the systems engineering development cycle, conducting studies of alternatives, and managing the system interfaces (Kossiakoff, et al., 2011), the main task of the system architects is to develop the system architecture. They have to design the architecture of the system and validate the architecture on the basis of the simulations carried out. To do this, the system architect must understand not only the needs and expectations of the customer but also those of the suppliers, employees (subcontractors) and other stakeholders. The System Architects have a variety of activities (Figure 8), mainly the collection, filtering and processing of data, but also more formal activities such as meetings, visits, etc. Most of their time is used for communication between the different members of the project. They are in contact with several interlocutors such as the sales manager, the project manager, the marketing manager, the technical manager, the engineers, and the designers. Finally, the System Architects validate that the system meets the customer's requirements. They must have detailed knowledge but also know how to switch back in order to move quickly from specific detailed views to more abstract views at a higher level (Muller, 2011).

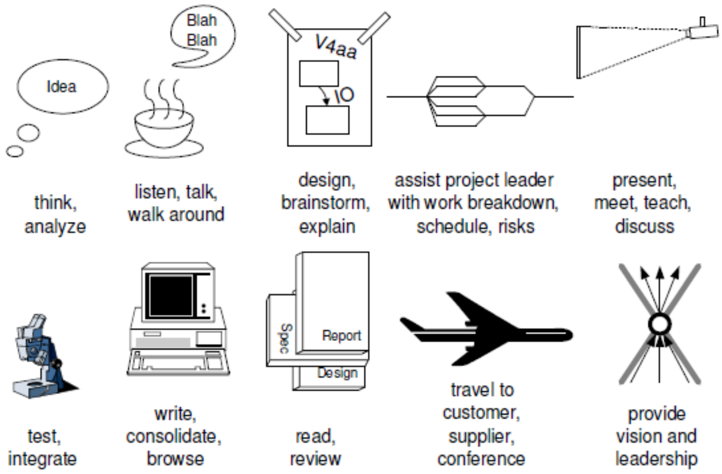


Figure 8: System Architect tasks (Muller, 2011)

They must also understand the issues of the product and the architectural framework they have set up, so that they can study the various design choices and technical solutions and make the right decisions according to various criteria (performances, maturity, price, risks, profits, etc.). They have a vision and understanding of the system and its context. The system architects should allow members of specialized teams (who have a limited field of vision) to make local design decisions by providing them with overall design information.

Thus, they first define the concept, set performance objectives, perform functional analysis, and finally design the architectures in collaboration with others designers. It is a creative process and many possible variations of architectures may meet the requirements expressed. The system architects therefore need methods and tools to analyze these architectures and choose that which best meets the high-level design goals set. Breaking down systems into subsystems, subsystems into modules, etc. is a major responsibility of the system architects. Indeed, this break down must make it possible to understand complex systems and facilitate their physical integration. In addition, the system architects must ensure constantly consistent design throughout the preliminary design phase (from the system to the physical level) (Muller, 2011).

This need to compromise between design objectives and requirements can sometimes be contradictory and constitutes one of the main activities of the preliminary design phase. The system architects guarantee balance and compromise with regard, for example, between cost and added value in terms of functionality and system performance. They are responsible for the integrity of the system specifications throughout the design. When a parameter changes, the system architects must verify and ensure that the other design parameters still meet the specifications and maintain performance.

The architect's work is critical during the preliminary design, because once the architecture has been chosen according to the "high-level" objectives it will form the basis of the work done by all the design / development teams (Walden, et al., 2015).

- **Simulation Architects:**

The simulation architect is the actor at the interface between the system architect and the simulation teams. They have to specify the simulation scenarios to the simulation teams, which include alternative physical architectures and the system's requirements (performances, physical constraints) provided by the System Architects, as well as several geometrical constraints (bounding box, projected locations of components). They have to assist the system architects in their choice of the most adapted concept architecture and they have to provide simulation teams with a simulation architecture, in other words suitably formalize a description of the simulation to be performed, in order to ensure that the simulation is adapted to the requirements expressed. Thus the System Architects may or may not validate a concept architecture obtained from the simulation results provided by the Simulation teams and validated by the Simulation Architect.

- **Simulation teams:**

Simulation Teams are composed of experts in different disciplines (electronics, mechanics, control, hardware/software, etc.) in charge of verifying that the behavior of the physical architecture meets the performance and spatial requirements through physical simulations.

- **3D architects:**

These actors of the conceptual design allocate space consumption and the initial 3D architecture of the components based on the physical architecture and the 3D design. This allocation relies on the 3D architects' expertise and discipline. They consider the constraints of the teams of other disciplines, and space allocation is decided during meetings. The largest space is often assigned to the most convincing teams rather than on the basis of scientific reasoning.

b. Interactions between the actors

When examining the interactions between the System Architects and the systems and the design specialists (i.e. multidisciplinary teams) (Figure 9), it is assumed that the systems engineers will be required to work in a multidisciplinary environment and grasp the essentials of the related disciplines (Kossiakoff, et al., 2011).

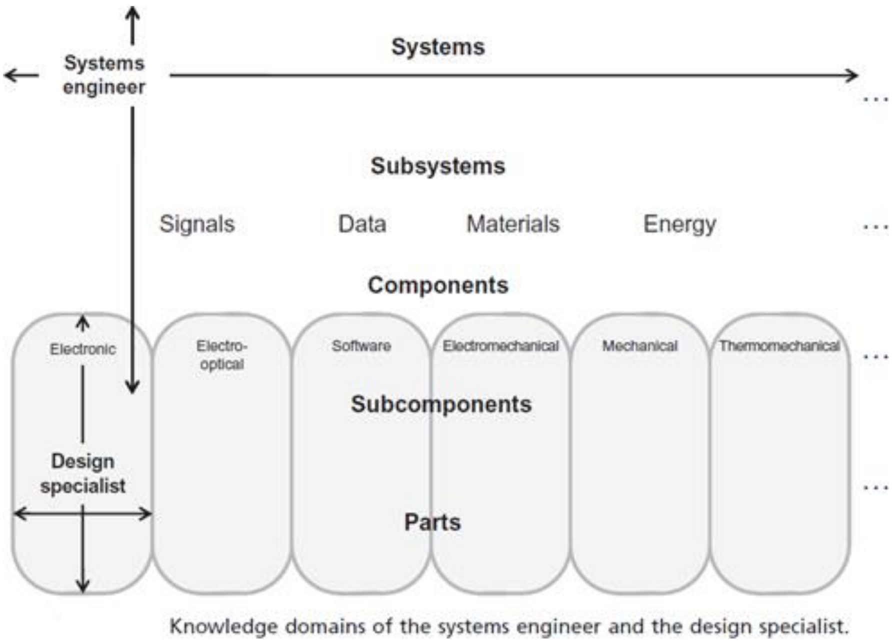


Figure 9: description of systems engineering domains (Kossiakoff, et al., 2011)

Figure 10 presents a view extracted from the global SIM project presented in Appendix 1 (Figure 154), which describes the interactions between the design actors and which will be considered in our research topic. The System Architects define the functions and their allocation in the components on the basis of the requirements. They must evaluate the performances of several alternatives of possible architectures and until they progress to an optimal version while respecting the constraints of risks, costs, delays and quality. Then, starting from a question posed by the System Architect, the Simulation Architects have to formalize a simulation scenario and identify the necessary simulation models and means. Finally, the Simulation Teams implement the simulations and provide their expertise and business know-how.

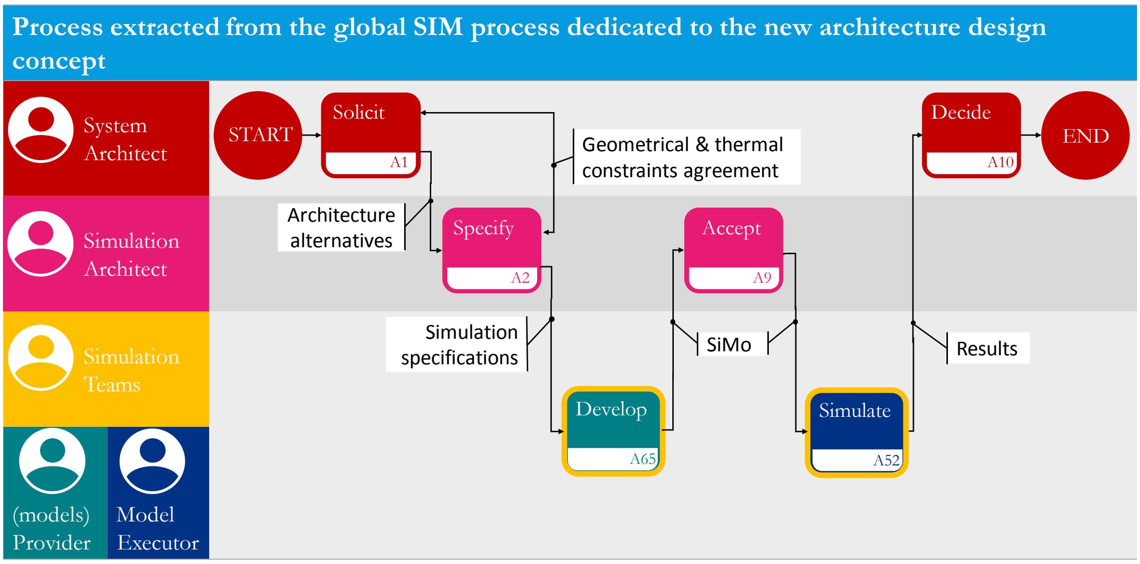


Figure 10: Extracted activities from SIM process

Compared to the global SIM process, we have considered that the (model) provider and the model executor address the same expertise and merge them into what we call the "Simulation teams" in charge of building the physical models and simulating them after validation by the system architects.

The interactions between these actors will greatly depend on the "interaction environments" in which they evolve.

Kvan (Kvan, 2000) proposed two paths of collaboration: a close coupled design process; and a loosely coupled design process. The latter can be split into two collaboration models: the supervised process and the unsupervised one.

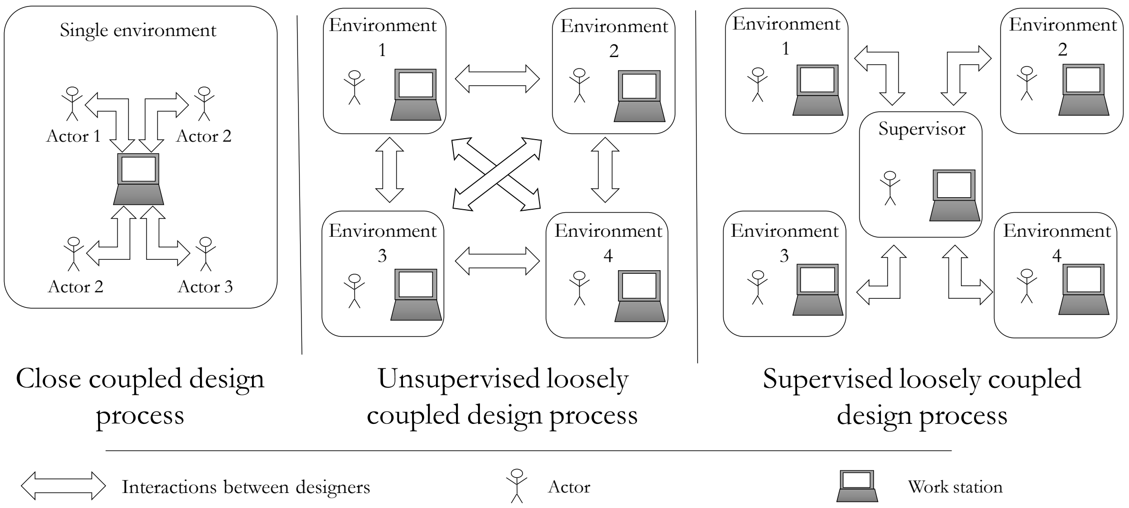


Figure 11: The different kinds of interaction environments

- Close coupled design process

Kvan (Kvan, 2000) described the close coupled collaboration process as participants working closely together to produce a design. The participants work intensely with each other, while observing and understanding each of the modifications made by their counterparts. This old collaboration model has been updated thanks to virtual reality tools (Anthes, et al., 2004) (Wolff, et al., 2007). Different participants can design the product together in a virtual collaborative environment (as shown in Figure 12).

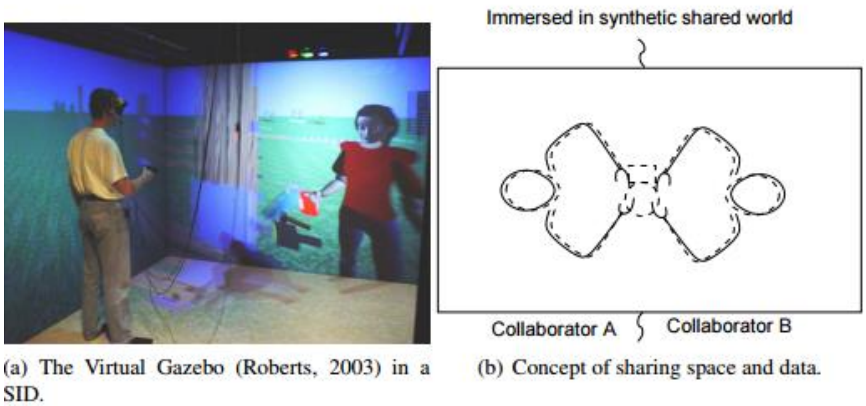


Figure 12: Close coupled collaboration used in a virtual environment (Wolff, et al., 2007)

In a close coupled design process, the fact that the design actors work together on the same model means that interoperability is not an issue, thereby facilitating their collaboration. For example, when a 3D architect positions a motor near a fuel tank, the thermal expert can immediately react by explaining the risk of explosion. Thus this approach minimizes the number of iterations between different designers. However, this approach has two main drawbacks. The first is that there is no traceability concerning the choice of architecture, since all the information is exchanged orally and there no trace of this information is left behind. Moreover, Steiner suggested (Steiner, 2007) that four participants for collaborative close design is the optimum number as more participants can generate inefficient design. Thus this approach is more adapted for a small company.

- Unsupervised loosely coupled design process

The unsupervised loosely coupled design process is the collaboration model used in most companies today. The different actors of the design cycle work in different environments with their own machines and their own models, without any supervisor to formalize the data exchange.

Pinelle et al. (Pinelle & Gutwin, 2005) defines the loosely coupled design process by low interdependence, high differentiation, and low integration. According to Orton and Weick (Orton & Weick, 1990), every single industrial activity is to some extent interdependent with a number of other activities, whose coupling can be defined in various ways depending on the number and the strength of their interdependencies. Kvan (Kvan, 2000) underlined "that much design is in fact loose-coupled, with each participant contributing what they can in different domains of expertise at moments when they have the knowledge appropriate to the situation. The participants are committed to working together because each has a particular expertise that they can contribute to the solution process." For these reasons, the loosely coupled design process focuses on multi-faceted exchanges between the different designers. According to Hagel et al. (Hagel, et al., 2002) "More loosely coupled designs employ a modular approach where the focus is on defining standardized interfaces across modules of activity. In this way, modules of activity can be inserted or removed to tailor the process and activities within a particular model and can be easily modified to accommodate changing business needs." Moreover, models can be transformed in order to provide the interface between the different actors.

As the data exchange is unsupervised, each design actor has to select the data to be sent to another designer and thus the data exchanged may not be adapted to that particular actor's needs. Nonetheless, the advantage of this approach is that it guarantees some freedom for the designers.

– Supervised loosely coupled design process

The supervised loosely coupled collaboration mode is the same as that described previously, except that the actors do not freely exchange data with other actors. The data are exchanged via a supervisor.

Indeed, Harvey (Harvey, 2001) proposed viewing the collaboration as a network. For him, the exchange between different actors (nodes of the network) should be managed by an actor network. This collaborative model is based on the sociology theory of the Actor–Network Theory. This theory shows the complexity of the increase of links according to the number of actors (Figure 12).

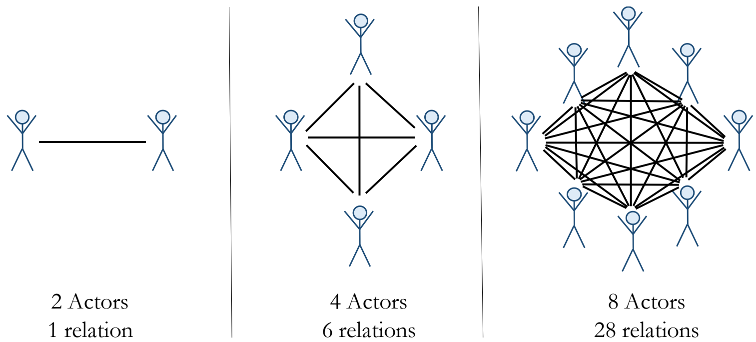


Figure 13: Illustration of complexity in the Actor Network Theory

The number of relations in the Actor Network Theory increases according to a polynomial trend with the number of actors. This large number of interactions implies a high risk of mistakes and misunderstandings between the different actors. Therefore the collaboration process for a large company requires a “supervisor” actor to avoid these problems.

This supervisor will identify and then connect together the common parameters contained in the different models, and update them for each new model (which can be very time-consuming and thus costly). Therefore, this actor must have good knowledge of the needs of the different designers, in order to provide the right information to the right designer.

Lee and Shepherdson (Lee & Shepherdson, 2004) proposed a method based on multi-agents, using a mediator (that could be assimilated with a supervisor) in order to exchange data and information between different actors. This mediator must break down the action of an actor into a set of top-down actions and contact other actors to give them the corresponding action. Then, the mediator brings together the data resulting from these actions to give them to the requesting actor, and identifies other actors who are involved by this result.

The close coupled design process is not widely used in industry and is mostly found in small companies (Kvan, 2000), while the loosely coupled design process appears to be more adapted for large companies, since more than four actors working together becomes less productive (Steiner, 2007). Concerning the supervisors, they could be the simulation architects that SIM projects tried to introduce. Supervisor actors are increasingly considered by the company partners of the SIM project. However, the unsupervised loosely coupled design process is currently that most used by companies and notably those that are partners of the SIM project.

Whatever the case, all these approaches address the issues of data interaction and raise the question of how to technically ensure the exchange of the corresponding data requested.

5. Design data interactions

To ensure data interoperability during the conceptual design, it is necessary to implement an MBSE approach in a multidisciplinary context. The three previous approaches have been studied according to their impact on data exchange.

Indeed, the objective of each structure proposed is to also improve the collaboration between different tools. The first consists of the creation of a single tool including different views. The second corresponds to the integration of a database which exchanges different parameters between different tools and actors. Finally, the third concerns the storage of all the design data in a common database. These data are transmitted directly from the different tools (Figure 14).

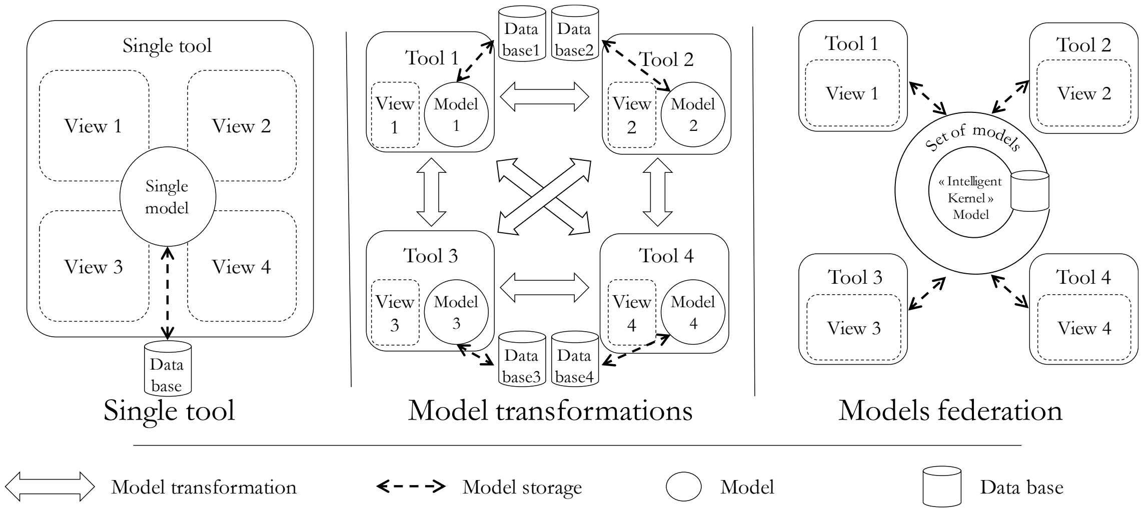


Figure 14: Description of the different structures of data exchange technical solutions

For each previous collaboration model, a technical solution is proposed in order to address the issue of data interoperability (Table 1).

Table 1: Proposed data exchange technical solution for the corresponding collaboration models

Collaboration model	Data exchange technical solution
Close coupled design process	Single tool
Unsupervised loosely coupled design process	Model transformations
Supervised loosely coupled design process	Model federation

a. The single tool

The unification of models can be done using a single tool. This tool can provide different views depending on the actor who performs the design. The main advantage concerns the continuity between different levels of modeling (functional, logical and physical) which is fundamental to decrease data losses, to reduce omissions, errors and redundancies, and then to guarantee compliance with the specifications.

For example, Ng et al. (Ng, et al., 2015) proposed to enrich the virtual reality environment with the FBS framework. The structure elements correspond to 3D components using virtual reality. The classical FBS approach is performed in a single tool "ARCADE", including a behavior analysis of the structure analysis performed by a simplified simulation: the 3D structure generated is animated thanks to the behavior simulation results.

An example of a commercial application which proposes a single tool with multiple views is CATIA V6 (Kleiner & Kramer, 2013). The RFLP process described by Sven et al. corresponds to the different steps of the conceptual design (Requirement, Functional, Logical, Physical), when considering the 3D Logic (simplified geometry for allocated component volume) for the 3D representation. Using the RFLP approach, each element in the RFLP model can be linked to another and traced to the requirements, while ensuring the consistency of the parameters between the different views by automatic updating. However, this approach could be automatized further: for example, when the behavior of a physical architecture is calculated using the Dymola tool (in the Logical view), the corresponding requirements that have to be validated are not explicitly visible. Indeed, as requirements are only textual in the Requirements view, there is no traceability possible between the requirements and the behavior results. When comparing the RFLP module from Catia V6 (Dassault systemes, 2017)(Dassault Systems) and another integrated tool for the conceptual design: TDC software (Software, 2017) (Knowllence) based on the APTE method, Segonds et al. concluded that the TDC suite appears to be more adapted for a system architect because it is semantically riche., However it does not provide a 3D architecture environment, which is important when communicating with different designers, as mentioned previously.

Papa et al. (Papa, et al., 2015) proposed an interesting approach based on the work of Plateaux et al. (Plateaux, et al., 2009), where the simulation parameters are not driven by a 3D architecture, but the 3D architecture is included directly in the simulation model. They proposed to couple the relative positioning constraint parameters with thermal parameters, in order to place components according to their thermal behavior in a Modelica environment. The role of geometrical constraints from TTRS model is to update the geometrical conditions between devices in a 3D space during the thermal simulations. In this step, this goal is accomplished by using the same parameters and objects related both to 13 constraints and to the simulation of the physical interactions. This implementation constraint requires that it is the same actor who carries out the thermal and the 3D analysis, and this actor has to hardcode the coupled model in Modelica language, which is not particularly user-friendly. However, few current simulation experts possess this double knowledge, and even less that of TTRS theory.

Finally, the single tool framework consists in providing an architecture with different views, which has the advantage of unifying the design architecture. However, this environment requires that designers must be adaptive in order to design their model with this single tool, contrary to current industrial practices which point to a diversity of expert tools for design.

b. Model transformation

Model transformation is now a very popular approach for managing data interoperability between the different actors of the design cycle. The emergence of many recent languages specialized in model transformation, like ATL (Jouault, et al., 2008), AGG (Taentzer, 2003), QVT (Object Management Group, 2016), highlights the interest of such an approach, notably for research studies. There are two kinds of transformation: from a model to a model (M2M), and from a model to a text (M2T). While the M2M transformation is rather a means for modeling semantic interoperability, M2T transformation is used more for automatically generating text documents from models to improve collaborations through an MBSE process rather than document-based one. M2M transformations thus enable all the actors of the design process to easily exchange the content of a model without necessarily knowing the modeling language or tool of the source model. This kind of transformation speeds up the design process, since the designers avoid wasting time with the model understanding step, made unnecessary, to adapt the source model to the desired language or tool. Furthermore, Model-to-Model (M2M) transformation includes two kinds of model: the Platform-Independent Model (PIM), which is independent of the specific technological platform used to implement it, and the Platform-Specific Model (PSM), which is linked to a specific technological platform (e.g. a specific programming language, operating system, document file format or database) and is required for the actual implementation of a model transformation.

A common approach to transforming System models implemented in SysML (System Modeling Language) or UML (Unified Modeling Language), into another language or tool, is to enrich UML and SysML by using UML profiles or SysML extensions, which add the required modeling elements from the Simulation Teams to the System model. Some examples of model transformations starting from SysML are cited below. The first SysML extension, based on standards, supports bond graph analysis (Turki & Soriano, 2005), and enables the transformation of the corresponding system models into Modelica, Simulink, or other 0D solvers. The second profile, proposed by Cao et al., addresses the model transformation from a System model in extended SysML into Simulink (Cao, et al., 2013), but requires the preliminary modification of the whole SysML model. Another model transformation between an extended SysML model and the TRNSYS commercial software enables evaluating the performance of thermal and electrical energy systems (Kim, 2014). The model transformation between SysML and DEVS (Discrete Event System Specification) formalism (Kapos, et al., 2014) describes state transitions and differential equations of the system, but does not permit simulation.

To satisfy this industrial need, the model transformation process can transfer data from one software application to another. Focusing on our research perimeter, Gross et al. (Gross, et al., 2012) extended UML to specify geometrical and thermal data, and the transformation rules are used to display (B-rep) geometry (positioned with coordinates) in CATIA or OpenCascade and to perform finite element modeling-based thermal simulation using the ESTAN-TMS language. The model transformation is provided by the FORTRAN language. Although this work does not address conceptual design (in which finite element modeling is not relevant for non-detailed geometry), it represents a major effort to provide a seamless process between the system model and the simulation teams. The drawback of this approach is that the geometry in UML is proposed only through a library. The model transformation is thus limited to the library and does not allow the creation of a new geometry in UML without modifying the transformation rules. Similarly, for thermal modeling, Gross et al. explained that this analysis is specifically adapted for the aerospace domain without any GUI effort, but new development is necessary for other domains. This will require a user-friendly GUI. Finally, this model transformation process does not seem to implement a bilateral process, since no traceability process (from 3D model to UML model) is mentioned.

Another model transformation between certain tools more adapted for the conceptual design was proposed by Schamai et al. (Schamai, et al., 2009). They proposed an innovative UML profile named ModelicaML, whose goal is to provide a seamless link between two important languages in conceptual design: SysML and Modelica. This UML profile gives the necessary formalism to manage the model transformation between UML and Modelica, but without covering the geometry (Schamai, 2009). We could imagine coupling Schamai's approach and that of Plateaux (Plateaux, et al., 2009) (which provides a geometrical library: components and constraints in Modelica) to introduce geometry in UML in a way compatible with Modelica, except that it does not deal with the need for a 3D visualization environment.

Model transformation does away with the need to rely on a single tool on which each designer is dependent. But the major drawback of this environment is technical, since for each new tool or release or new design actor, it requires major programming development to ensure each transformation process (Shen, et al., 2008) (a process is required for each pair of tools). Moreover, compared to the first approach, an important issue is that certain model transformations do not guarantee consistency between each model developed. Indeed, a SysML model structure can be transformed into a 3D model, while the modifications made to the 3D CAD model may not be automatically transferred back to the SysML model. This greatly depends on the content of the transformation rules.

c. Model federation

Model federation is a hybrid concept between the first and the second approach. A unique database is built to store all the model data (e.g. a PDM structure) and a partial projection generates the models for each tool.

Gross et al. (Gross & Rudolph, 2016) used the federation between SysML for system architecture, 3D CAD modeling for geometry, Simulink modeling for altitude control, and ESTAN modeling to support thermal simulation, for an artificial satellite design. This approach, although close to our research topic, used the finite element approach which slows down the simulation step and thus is more adapted to the embodiment design stage.

Thramboulidis (Thramboulidis, 2013) proposed to implement the federation approach by using the powerful and rich semantics of the SysML Language. A block can thus contain information on the respective domains of different disciplines: electronics, mechanics, and software. Although each domain model can have a local optimum according to the requirements, the federation approach does not ensure that the combination of these different models can succeed in finding the optimal for the system. Moreover, the data model provided does not include the exchange of data types, for example, a geometrical dimension such as length is not automatically given in a thermal model. The connection must be specified manually for each model.

Federation is an interesting approach, since the single data base is always synchronized, and thus there is no inconsistency between the different models. However, although the first paper illustrating this approach (Heimbigner & McLeod, 1985) was published in 1985, the federation model is not widely used in industry. Indeed, the development of such technology implies that each actor, including service providers, should have access to a single data base, which can raise security issues concerning the product under development (risk of hacking the all the database contents). Moreover, a database which contains information on each version of each component of a complex system must be capable of managing vast quantities of data.

III. Conclusions

Model Based System Engineering (MBSE) proposes methods for implementing a model-driven design process, something that is increasingly sought in industrial practices. The usual SysML language associated with a suitable MBSE method can easily support conceptual design while providing a multi-view description of the system. As the objective of this design phase is to select a concept architecture, before moving on to pre-design during the embodiment design, it seems interesting to assess various architectures regarding physical (and notably thermal) constraints. As thermal behavior is based on geometrical considerations, the works related to the integration of geometry and physical modeling during the conceptual design have been explored. Although many research works over the last fifteen years have addressed the integration of geometrical data for conceptual design by focusing on its importance for tolerancing, (Mao, et al., 2008) (Dantan, et al., 2003) process planning (Feng, et al., 1999) (Hassan, et al., 2010) and assembly (Sodhi & Turner, 1994) (Hsu & Woon, 1998) , few research studies have dealt with its implementation in the context of system engineering, especially in the case of evaluating design concept architecture. Concerning the assessment of 3D architecture in conceptual design, two main methods have been described: metrics and simulation. Simulation-based architecture assessment now seems that which is best adapted regarding multi-physical requirements. Finally, considering interaction management, three collaborative structures have been described with three technical implementation solutions to manage collaborations and design data exchanges between different actors: single tool, model transformation and model federation.

Now that we have examined the existing approaches that could be used to address our research issue, in the next chapter we will define the detailed requirements that will be used as criteria to assess these approaches and select the most suitable one.

Chapter 2 - Approach selection process

I. Expression of needs 31

II. Analysis of existing collaborative structures 36

III. A proposal of alternative approaches 39

A. First approach for data exchange automation of current industrial practices 40

B. Second Approach: geometrical and physical enrichments for semantic interoperability and traceability 41

C. Third approach: physical and geometrical specifications and traceability in a single 3D physical platform 43

IV. Approach selection 44

V. Demonstrator implementation environment definition . 46

A. Validation of the SysML Language for the System modeling .. 46

B. Selection of the model transformation process 49

VI. Conclusions 52

The previous chapter underlined that architecture selection is becoming a crucial step, due to the vital need to integrate new technologies. These new IT-based technologies have to be integrated in classical mono-domain components and in certain mechatronic systems. The design of such multi-domain systems is a delicate and complex process, notably due to the increasing number of components that have to be included in a compact volume usually involving multi-physical couplings between them.

Furthermore, the complexity of these multi-domain systems requires heterogeneous resources, whether human or technical. Indeed, the participation at the beginning of the design process of several multidisciplinary teams using different modeling tools that do not always ensure consistency between the various modeling levels makes the design of such multi-domain systems very difficult. To address this need for multidisciplinary collaboration, new approaches have emerged, notably methods based on Model-Based Systems Engineering (MBSE), which is aimed at ensuring the consistency of the modeling data and the traceability process for verification, while improving communication between teams.

In parallel, as the design of such systems also requires multi-domain simulations to evaluate the performances and multi-physical couplings of candidate physical architectures, various simulation environments like Dymola/CatiaV6, Comsol, Ansys Simplorer, and Simulink can be used. However, these tools are typically used once one physical architecture of a concept has been chosen, without first ensuring that a corresponding 3D architecture of this concept can satisfy the multi-physics constraints usually generated in such systems. This shortcoming leads to late changes during the embodiment and detailed design phases, and then to a dramatic increase of the corresponding costs and time.

Based on this observation, this chapter first expresses the needs related to our works, in accordance with the MBSE approach. These needs have been represented in SysML and established based on the research issue, the objective of a possible future industrial implementation and the PhD constraints. They aim at providing the criteria required to underpin the selection of the most suitable collaborative organization and structure among those established in the previous chapter. Based on the structure selected, the collaboration between system architects, 3D architects, and simulation teams has been studied through three approaches for the system architecture assessment during the conceptual design phase. Finally, after selecting the best adapted approach, the global structure of the framework is described through the definition of its corresponding demonstrator implementation environment.

I. Expression of needs

Thus it is now necessary to modify the "classical" concept architecture process. One research issue is how to select the best physical architecture of such complex systems during the conceptual design phase, in order to limit multi-physical couplings in further stages. This results in the need to evaluate the 3D architecture of concepts in the conceptual phase, in order to verify their multi-physical constraint specifications upstream (Figure 15). Based on the previous state of the art (Chapter 1 II.), we saw that there are no currently available methods capable of performing physical concept verification based on the positioning of 3D components. We propose to contribute to these research issues, by providing a framework to assess 3D architectures under physical constraints, by focusing on thermal constraints. In this section, we first define the specifications of such a framework, before proposing different structures that satisfy these specifications, even if only partially.

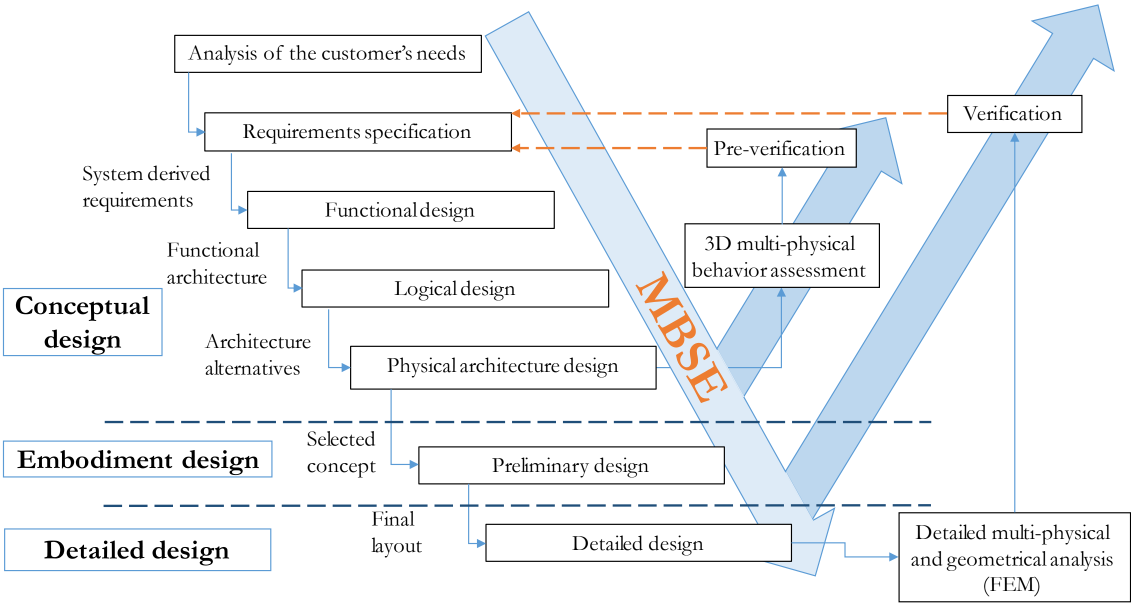


Figure 15: Current design cycle and its proposed evolution.

As the conceptual design is a phase in which many ideas emerge (chapter 1), it is important to provide a means of assessing various alternative architectures to select that which will most reduce the risk of long iterative loops in further design phases (embodiment and detailed design). Figure 16 presents a state machine diagram (Appendix 2 II.E) that describes where the proposed framework could be integrated within the current industrial design cycle, to take into account thermal interactions in the conceptual design phase.

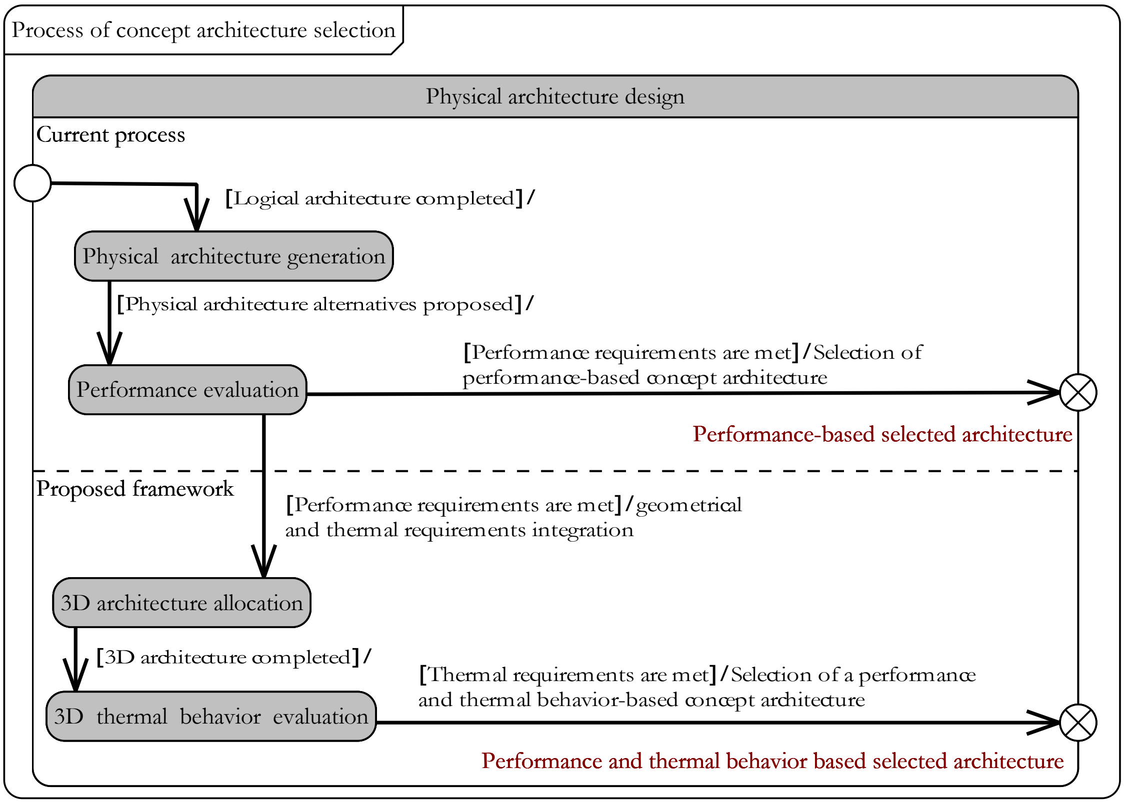


Figure 16: Position of the proposed framework in current industrial design cycle.

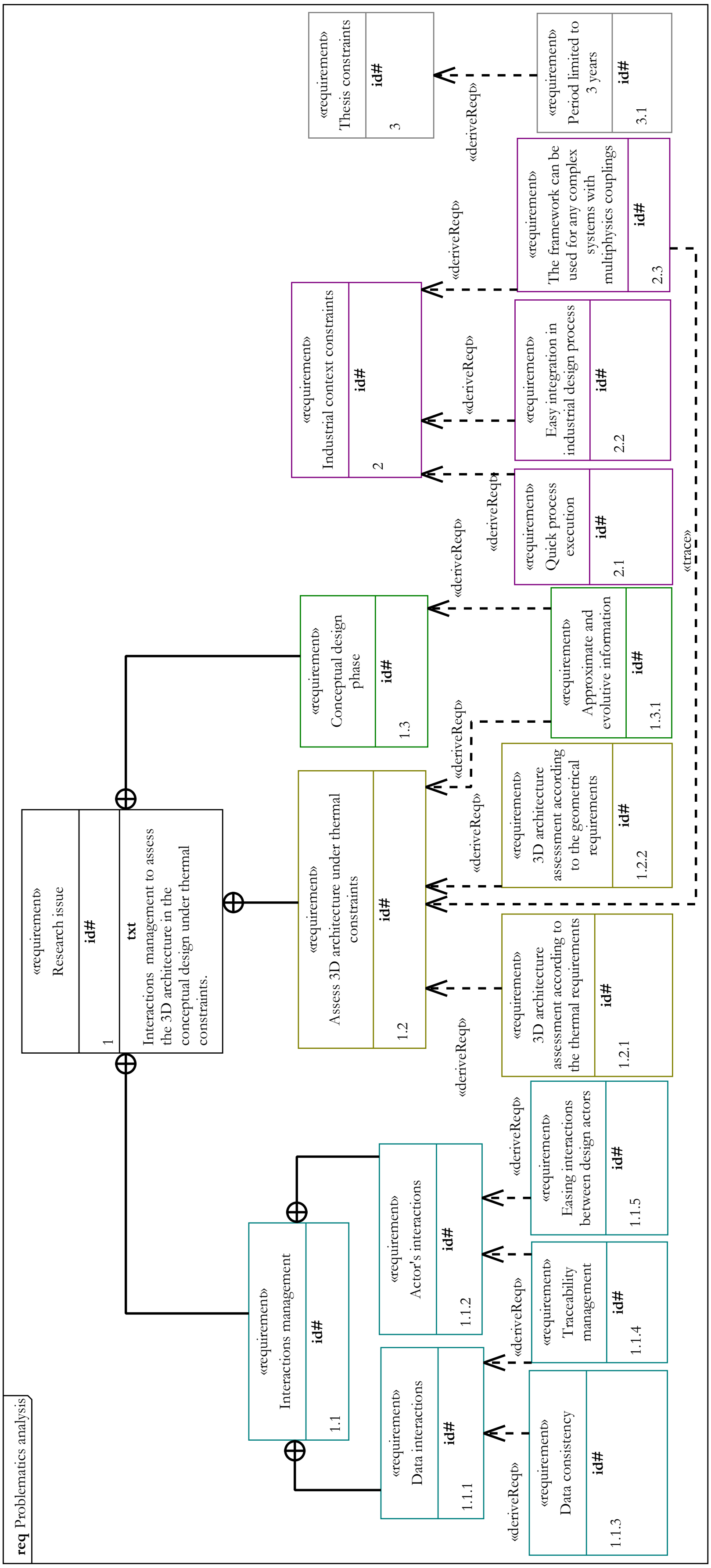


Figure 17: Research issue analysis related derived requirements

The derived requirements of this new architecture evaluation process are presented in a SysML requirements diagram (Req diagram, Appendix 2 II.A)) in Figure 17. To provide these derived requirements diagrams of our main initial objectives (i.e. to manage design interactions for the 3D architecture assessment under thermal constraints in the conceptual design phase), we follow the black box analysis of the MBSE methodology proposed by Mhenni et al. (Mhenni, et al., 2014), based on the SysML language.

The detailed description of these requirements is provided in Table 2.

Table 2: Requirements specification

Id#	Name	Txt
1	Research issue	Interaction management to assess the 3D architecture in the conceptual design under thermal constraints.
1.1	Interaction management	The interactions to be tackled concern both the exchange between model data and the human interactions between design actors.
1.1.1	Data interactions	Data interaction management addresses the interoperability between data from various models with different languages and tools.
1.1.2	Actor interactions	The human interaction management requirement describes the need for collaboration between the different conceptual design actors.
1.1.3	Data consistency	Data consistency facilitates efficient interactions between actors, avoids errors, and thus reduces design time.
1.1.4	Traceability management	Traceability simultaneously addresses data exchange for the verification and validation steps and human interactions, since the traceability of a valid or invalid architecture is very important for capitalization in future developments. Indeed, traceability management allows not only tracing the requirements but also the simulation results of a selected/discarded 3D architecture for the analysis and decision-making process.
1.1.5	Easing interactions between design actors	Design actors, e.g. the System architects, the simulation teams or the 3D architects, usually work in their own environment, with their specific tools. The framework has to propose a collaborative environment, so that all the actors can continue to work with their usual model and tools.
1.2	Assess 3D architecture under thermal constraints	Evaluating physical architecture according to thermal requirements usually implies that a 3D architecture must have been built previously. The thermal simulation results will depend on the thermal behavior of the components positioned relatively.
1.2.1	3D architecture assessment according to the thermal requirements	The framework must include the specification and the fulfillment of thermal requirements, based on the positions of the 3D components.
1.2.2	3D architecture assessment according to the geometrical	As the thermal behavior depends on the components' positions (3D architecture), the framework must take into account several geometrical requirements. For

	requirements	example, expert knowledge and certain business rules may specify a minimal distance between two heating components.
1.3	Conceptual design phase	As the conceptual phase is aimed at providing the suitable architecture of a concept, the framework proposed has to address this objective, taking into account the inputs and outputs of this design phase and the knowledge level of the system, which can be low.
1.3.1	Approximate and evolutive information	As the design phase addressed is the conceptual design, the data concerning the architecture components can be very poor. Thus the designers aim at obtaining approximate information to assess alternative architectures that can evolve towards higher levels of detail as the conceptual design progresses**.**
2	Industrial context constraints	Although this requirement is not directly linked to scientific problematics, it implies that the framework proposed should be adapted to the industrial processes and tackle the economic challenges involved. Thus these industrial needs will be gathered through interviews with the design actors involved in the partner companies in the SIM project (Airbus Group and Renault).
2.1	Quick process execution	Architecture selection will be fast to meet the objective of reducing design time.
2.2	Easy integration in industrial design process	The framework proposed will be seamlessly integrated in current industrial design processes. Easy integration will be ensured provided that the new framework involves few changes in the current industrial processes of the partner companies in the SIM project (Airbus Group and Renault), assuming many companies operate in a similar way. It also includes the conceptual design actors' habits (design activities, work environment and tools, etc.) and the common language and tools used in an industrial context.
2.3	The framework can be used for any complex systems with multi-physics couplings	The emergence of increasingly complex IT-based systems makes it necessary to take into account non-desired multi-physical couplings as soon as possible in the design cycle.
3	Thesis constraints	This work must fulfill common thesis requirements.
3.1	Period limited to 3-years	This work must be performed within the limited duration of a PhD thesis (3-years).

II. Analysis of existing collaborative structures

The last paragraph of the previous chapter proposed a comparison of three different implementation methods for a collaborative approach we have called “collaborative structures”: single tool, model transformation and model federation. To select the structure that will fulfill our requirements, certain new requirements have been added for use as selection criteria (indicated in red contours in the figure). They are described in Figure 18 and detailed in Table 3.

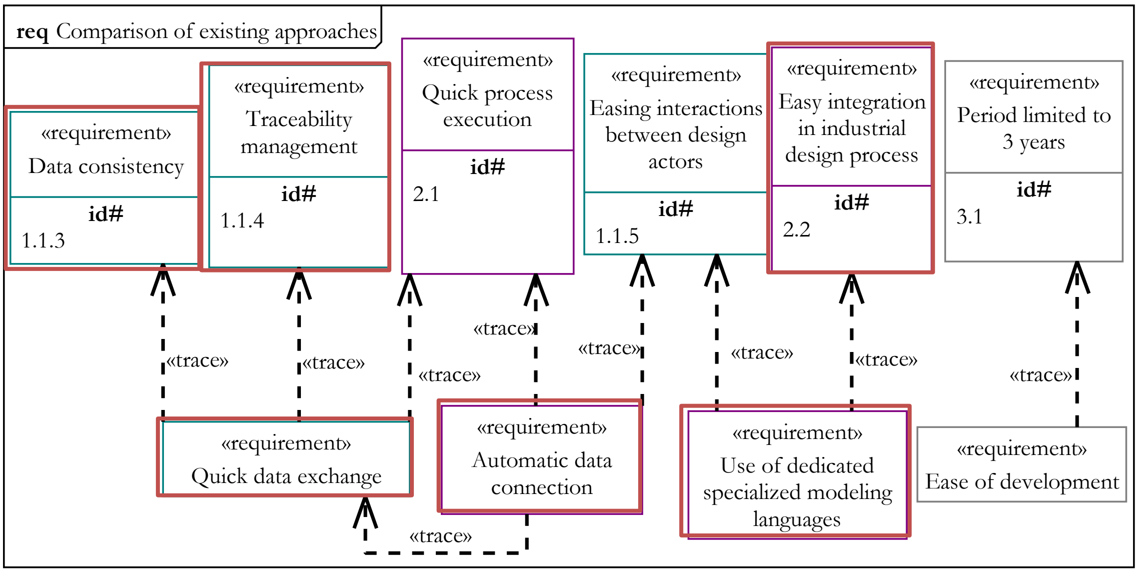


Figure 18: Selection criteria related requirements for choosing the collaborative structure to be implemented.

Table 3: Description of additional selection criteria related requirements

Id#	Name	Txt
1.1.10	Quick data exchange	The exchange between different models and tools must be performed quickly.
2.5	Automatic data connection	An efficient way to reduce design time, data inconsistencies and to ease collaboration between design actors, consists in automating the data connection. This means that users will not need to specify the parameters and data to be linked together for each model.
2.6	Use of dedicated specialized modeling languages	To guarantee interaction between different designers, it is important that they can use their own specific modeling languages and tools.
3.2	Ease of development	Due to the limited scope of the PhD, ease of development mainly addresses the production of the demonstrator: it has to be easy to implement by the PhD student involved, so it can be completed within three years.

For each collaborative structure, the requirements mentioned previously have been assessed (Table 4).

Table 4: Comparison of each collaborative structure according to the selection criteria related requirements

Requirements	Single tool	Model transformation	Model federation
Traceability management	✓	✓ For bidirectional model transformation	✓
Model consistency	✓	✓ Only if designers trace data back at each modification	✓
Quick data exchange	✓	✓	✓
Automatic data connection	✓	✓	✓
Use of dedicated specialized modeling languages	✗	✓	✓
Ease of development	✓	✓ Depending on the numbers of tools/languages involved	✗
Easy integration in the industrial design process	✗	✓	✗

The previous table shows that no approach meets the whole set of requirements. Each approach has its shortcomings.

First, concerning the model consistency requirement, the model transformation approach will meet this requirement provided that the designers return their model to all the other actors every time they modify it.

Ease of development is not systematically ensured with model transformation, when a large number of model transformations must be developed and updated in proportion to the number of languages/tools involved and their corresponding releases (Chapter 1 II.C.5.b). In the SAMOS framework, only 3 tools will be used, with no release addressed in the 3-year period.

Concerning the model federation approach(Chapter 1 II.C.5.c), its development leads to many arduous problems, notably for data security and big data issues, due to the single data storage location linked to the single main model. These difficulties are solved neither by the "Ease of development" requirement nor that of "Easy integration in the industrial design process". Indeed, data security and big data do not fall within the scope and field of knowledge of the PhD thesis concerned. In addition, the corresponding industrial constraints were described in the Introduction I.B.

Thus, the model transformation approach seems to be the best compromise according to our requirements. Nevertheless, when choosing this approach, particular attention must be paid to the designer's procedures of propagating their updated model to the other designers involved, in order to ensure data consistency.

Specific requirements related to the model transformation process are described in Figure 19 and Table 5.

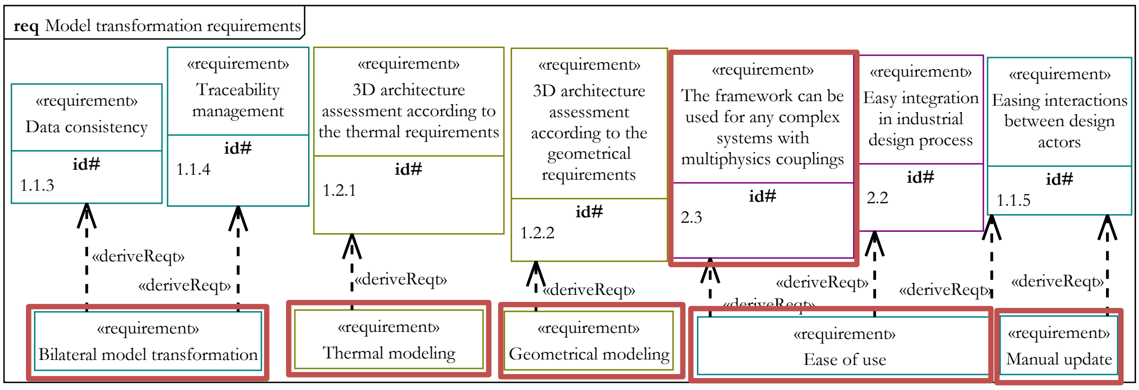


Figure 19: Requirement diagram for the analysis of the existing model transformation process.

Table 5: Description of the additional requirements related to the model transformation process

#Id	Name	Txt
1.1.6	Bilateral Model transformation	The bilateral model transformation approach meets both the requirements regarding data consistency and traceability management, as described in chapter 1.
1.1.11	Manual update	Manual updating is required for both transformation directions to ensure data consistency.
1.1.8	Ease of use	This requirement results from the need for the demonstrator to be tested by industrial companies, and especially by many actors from different disciplines. Moreover, this requirement satisfies easy integration in current industrial process.
1.2.3	Thermal modeling	The thermal assessment of 3D architecture requires that thermal modeling is performed for the simulation. This modeling must then be considered for the MTM transformation from and towards the system model for the thermal information specification and traceability processes.
1.2.7	Geometrical modeling	The thermal assessment of 3D architecture requires considering a geometrical modeling approach for the 3D positioning of 3D components, and ensuring the MTM transformation from and towards the system model for the geometrical information specification and traceability processes.

The state of the art mentions two model transformation approaches that integrate both thermal and geometrical analysis. The first is the approach proposed by Gross et al. (Gross, et al., 2012) and the second is the ModelicaML (Schamai, et al., 2009) profile potentially coupled with the approach of Plateaux et al. (Plateaux, et al., 2009). The previous specific requirements related to the model transformation process have been evaluated according to these approaches in Table 6.

Table 6: Comparison of existing model transformation approaches regarding the previous specific requirements.

Requirement name	Gross et al. approach	ModelicaML and Plateaux et al. coupled approaches
Thermal modeling	Finite element analysis (✓)	Thermal resistance (✓)
Geometrical modeling	B-Rep (✓)	TTRS (✓)
Bilateral Model transformation	✓	✗
Manual update	Required (✗)	Required (✗)
The framework can be used for any complex systems with multi-physics couplings	✗	✓
Ease of use	✓	✗

While the bilateral model transformation requirement is met by the approach of Gross et al. , the ModelicaML development is not bilateral, since only the Modelica simulation results are traced back to SysML, and the modification of the architecture is not propagated in the SysML model. Manual updating is required for both approaches to ensure data consistency. The approach of Gross et al. is only adapted to the domain of satellite applications, whereas ModelicaML can be used for any complex multi-domain/multi-physics systems. The approach of Gross et al. is easy to use in contrast to ModelicaML, since the geometry constraints are not easy to model in Modelica without a CAD model, as described in the Chapter 1 II.C.5.b . Finally, none of these approaches fully meets our requirements.

III. A proposal of alternative approaches

Since the traditional methods of system architecture assessment during the conceptual design phase do not satisfy the industrial MBSE expectations described in Introduction I.B, we propose three new approaches which will be described successively in this section. The following approaches aim at ensuring the suitability of the models exchanged during the different conceptual design stages through several successive model transformations between the models of the System Architects (SAs), the 3D Architects (3DAs) and the Simulation Teams (STs), respectively. Indeed, in order to make the architecture assessment more efficient and reduce the downstream iterative process; multi-physical simulations, usually performed during the embodiment and detailed design phases, are therefore brought forward in the conceptual design phase.

Finally, this section deals with three alternative MBSE approaches, based on model transformations between system models, physical behavior simulation models and 3D models. These approaches aim at evaluating physical architectures during the conceptual design phase, by relying on preliminary physical simulations under geometrical constraints.

A. First approach for data exchange automation of current industrial practices

The first approach suggested is illustrated in Figure 20 and presents the automation of the common industrial practices presented in Introduction I.B., for the conceptual design. This approach proposes to provide a "physical 3D architecture framework" aimed at facilitating data exchange management between the different actors (System architects, 3D architects and Simulation Teams).

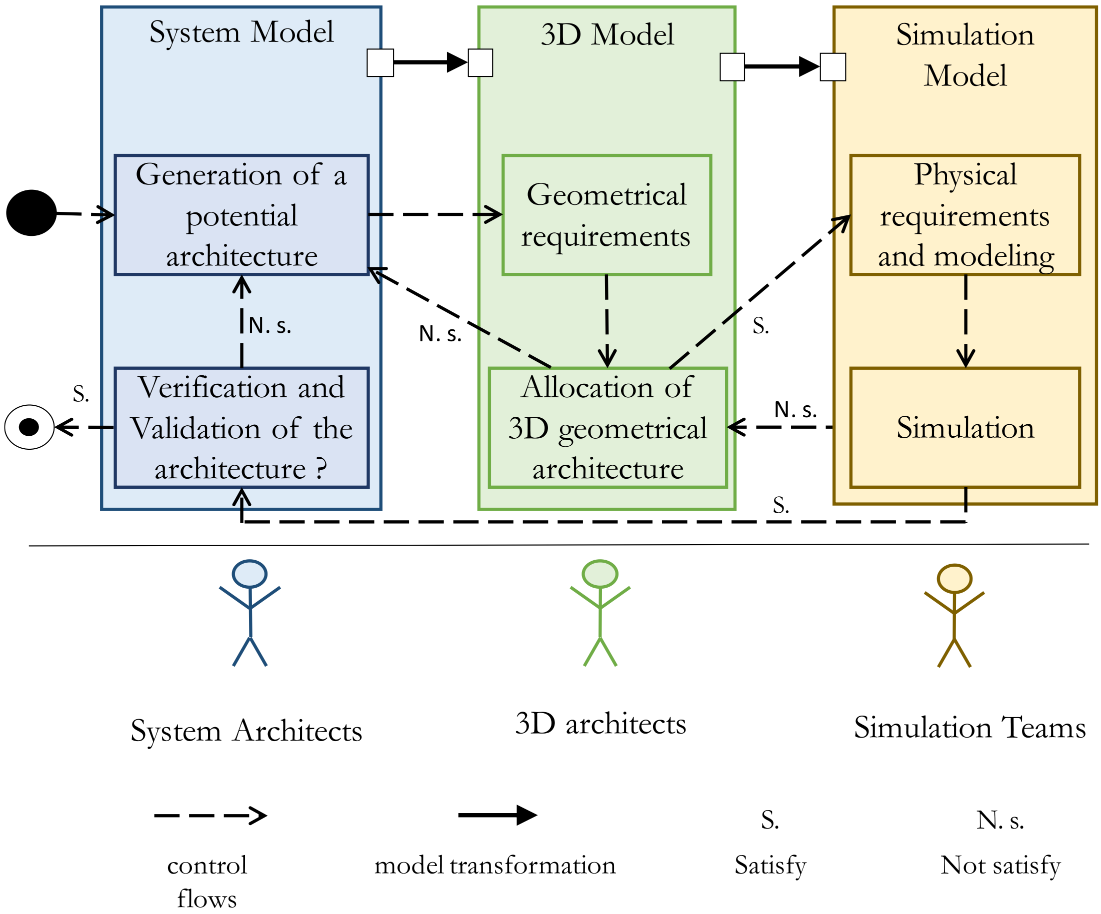


Figure 20: Principles of the first physical 3D architecture framework

Firstly, the physical 3D architecture framework will allow system architects to automatically provide 3D architects with the physical architecture to be assessed, via the transformation from the System model into a 3D model, so that the 3D architects can specify the allocation of component volumes in the 3D environment. This System-3D model transformation will generate the structure of the components, with an empty 3D modeling element for each component. In the new 3D model generated, the 3D architects will first add geometrical requirements, typically assign the geometry (shape and volume) and an initial position for each component of the whole system. Secondly, the framework will provide the model transformation from the predefined 3D model (completed by the 3D architects) into a simulation model. This 3D model-simulation model transformation takes into account the architecture components with their corresponding geometrical data. Then the simulation teams will add the physical requirements before performing simulations in order to verify whether the 3D architecture designed meets these requirements and the spatial allocation resulting from the 3D architects. Usually, at the beginning of this process, the first 3D architecture proposed by the 3D architects does not meet all the physical performance requirements provided by the System architects, thus the Simulation teams ask the 3D architects to make modifications to their spatial allocation or to give some degree of freedom regarding the geometry of the components or the relative position constraints. Consequently, many iterations may be performed until a convenient architecture is found, and for each iteration the Simulation teams have to perform a new physical model of the most recent 3D spatial architecture proposed. If the problem persists, in the case where the simulation teams do not succeed in satisfying the specifications for physical performance with any of the spatial allocations proposed by the 3D architects, the potential physical architecture is invalidated by the System architects who then have to propose another physical architecture. After that, the previous iterative steps are restarted and the different actors are confronted with the same costly and long process.

Concerning the advantages of this first approach, it ensures a slight improvement in data exchange between the different actors, by automatically transferring the specifications necessary for the assessment process as a function of the successive model transformations, instead of this being done through oral or document exchanges. This framework gives the design actors a dynamic basis by gathering all the design information and facilitating the automatic data exchange of the different models in real time during the architecture assessment process. Moreover, it ensures data consistency between the different models implemented. Another advantage of this approach is that it results in slight modifications of the current industrial design process.

However, this approach does not reduce the risk of a high number of iterations before finding a suitable architecture. It can thus prove very costly for companies and generate time losses. Moreover, the traceability in the system model of geometrical and physical data resulting from the assessment process cannot be ensured, because the System model does not usually contain the corresponding semantics.

Therefore, it is now essential to propose a faster architecture assessment framework which would be more efficient and thus feature numerous cost and time benefits while providing modeling data traceability.

B. Second Approach: geometrical and physical enrichments for semantic interoperability and traceability

The second approach suggested, illustrated in Figure 21, proposes to enrich the System model with geometrical and physical semantics through bidirectional model transformations, in order to add the traceability process required by MBSE principles.

The process of this approach is slightly different from the first one, since the System architects can generate architectures with geometrical and physical requirements, thanks to the enrichments of the System models' semantics. Then, a model transformation takes place and provides, for each architecture component, the corresponding 3D modeling element, integrating geometrical requirements (for example, shape and volume) previously specified by the System architects in the System model. In accordance with other geometrical requirements (for example, positioning constraints), the 3D architects complete the volume allocation and position of the remaining components in their 3D environment tool. Once this complete 3D architecture has been built, the reverse model transformation from the 3D model into the System model allows enriching the System model with geometrical information from this 3D architecture. These geometry-enriched physical architectures and physical requirements are then provided to the Simulation teams through another model transformation to be assessed by quick physical simulations. If these simulation results fulfill the System architects' requirements, this physically-enriched 3D architecture is traced back to the System model by a third model transformation to be validated by the System architects.

On the contrary, if the simulation results do not satisfy the initial System architects' requirements, they are still traced back to the System model for capitalization, and the System architects have to decide if they prefer to ask the 3D architects for another alternative 3D architecture or to propose another physical architecture before restarting the assessment process. After a number of iterations, the process converges to an enriched system architecture, satisfying the different geometrical and physical requirements.

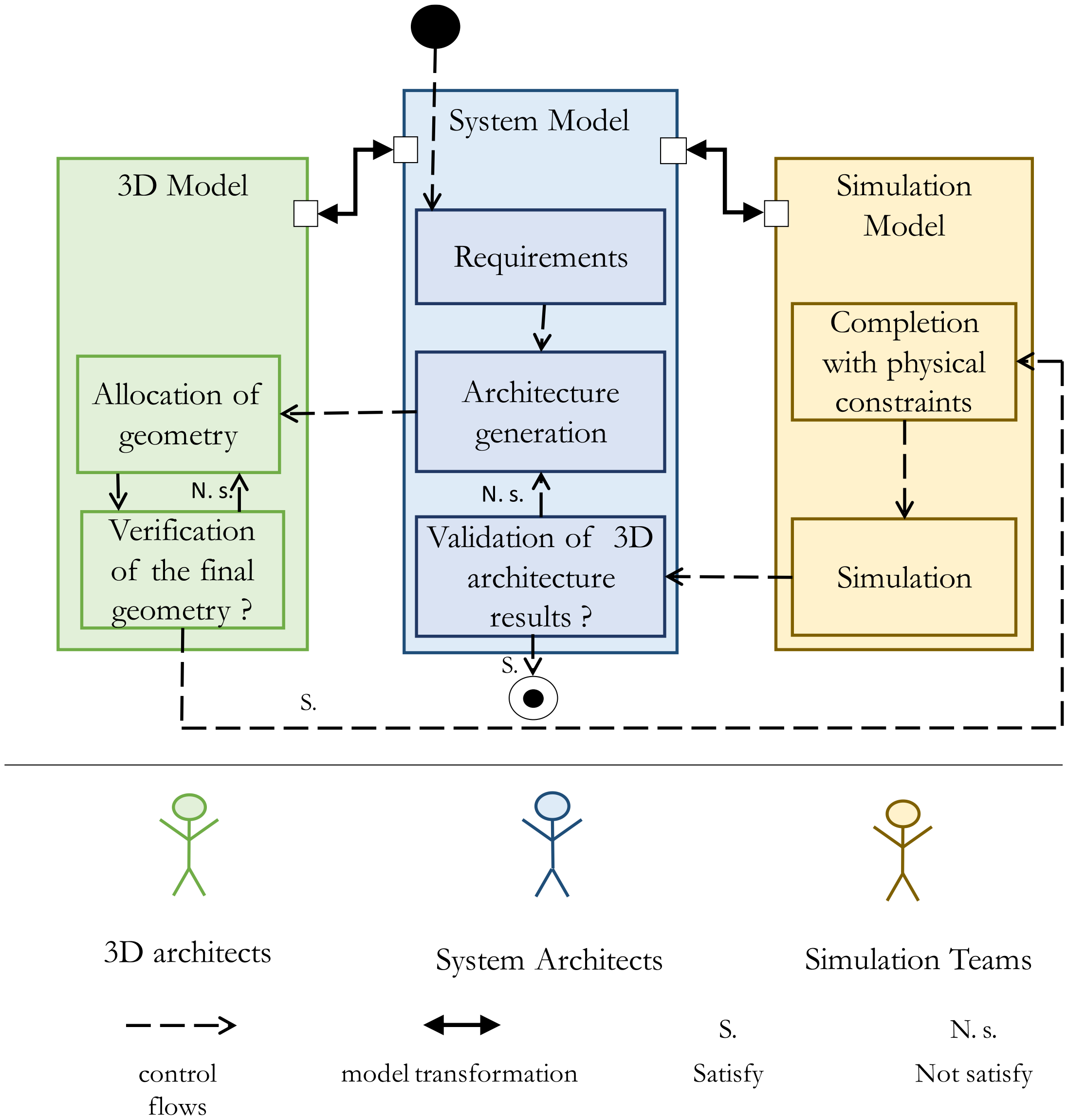


Figure 21: Principles of the second physical 3D architecture framework.

The first advantage of this approach is to reduce the number of design iterations. Indeed, both the 3D architects and the Simulation teams can themselves verify if the enriched 3D architecture proposed meets the geometrical and physical requirements, respectively, which were initially given to them by the System architects, before tracing back the results of their work in the System model to be validated by the System architects. The semantic geometrical and physical enrichments of the System model then allow tracing and capitalizing the geometrical and physical information of the 3D architecture assessed in the System model, so that the System architects have all the necessary information in a single model to validate the proposed 3D architecture or not. Finally, bidirectional model transformations prove to be quick processes for ensuring the consistency and traceability of model specifications and results. All these advantages help to reduce the design time and costs of the concept assessment process.

However, this approach is not efficient enough, since for each architecture modification, the process proposed requires intervention from the System architects. Indeed, as this framework provides access to all the design information only through the System model, it does not allow the 3D architects and Simulation teams to easily interact to quickly find a compromise that fulfills the System architects' requirements.

C. Third approach: physical and geometrical specifications and traceability in a single 3D physical platform

To tackle this problem, a third approach presented in Figure 22, proposes to integrate the tasks of the 3D architects and Simulation teams within a single physical 3D modeler environment.

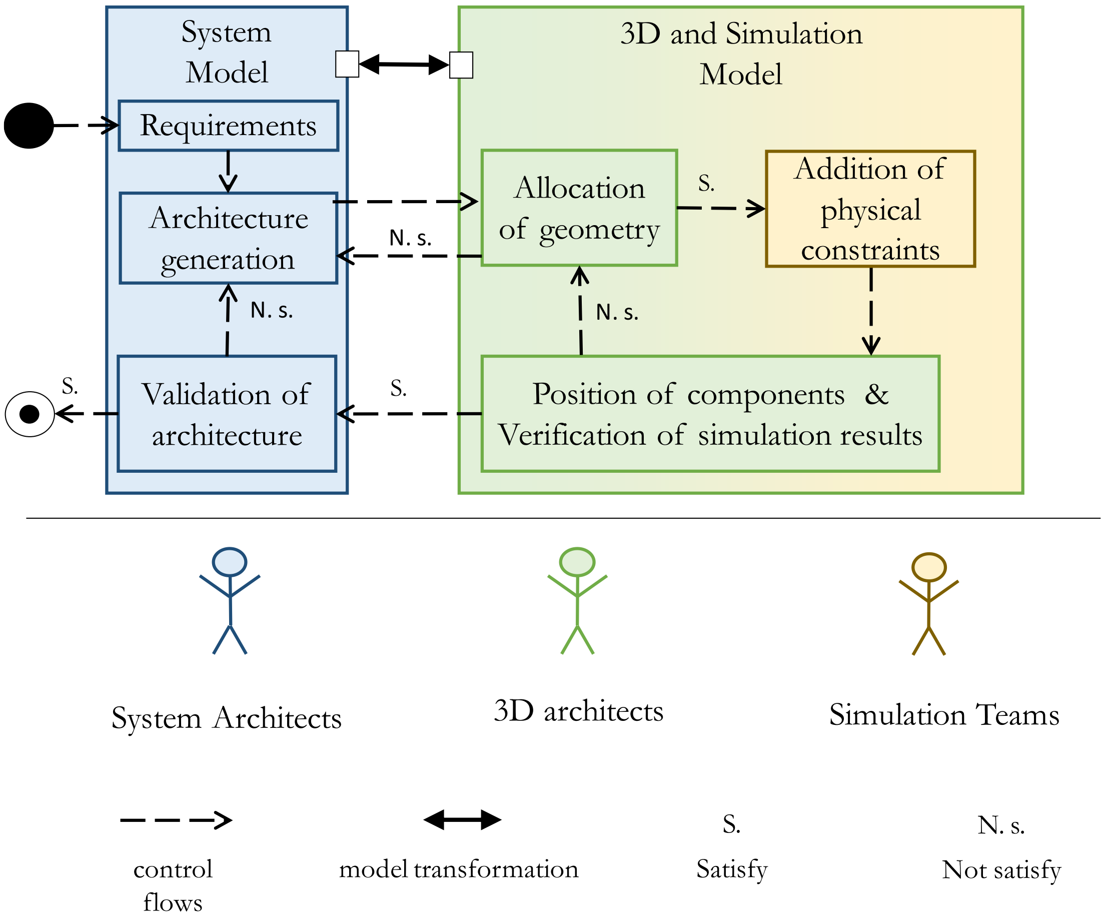


Figure 22: Principles of the third physical 3D architecture framework.

This process begins with the generation, by the System architects, of the physical architecture and the geometrical and physical requirements that are automatically transferred into a 3D and simulation model through the model transformation process. Then, the 3D architects allocate the volume of the initial components to meet the System architects' geometrical requirements stated previously in the 3D physical modeler environment. If a 3D architecture is satisfactory, the Simulation teams add the System architects' requirements, which include the different physics requirements, to carry out the physical modeling of the 3D architecture proposed, taking into consideration, for example, the thermal radiation, conduction, convection areas and acceptable range of physical values of the components, etc. The Simulation teams can then perform simulations to analyze, for example, the effects of the physical constraints for the given 3D architecture. Based on these results, if satisfactory, the corresponding 3D architecture is traced back to the System architects in the System model through the reverse model transformation; if they are not satisfactory the 3D architects propose a new 3D architecture, taking into account the unsatisfactory physical constraints, in collaboration with the Simulation teams. If the 3D architects fail to find a 3D architecture that meets the System architects' requirements and physical constraints, a request for modification is then transmitted to the System architects who are asked to change their proposed physical architecture.

The main advantage of this approach, compared with the others, is the unique physical 3D modeler environment that allows reducing both time and cost, by facilitating the quick evaluation of the 3D physical architecture via direct collaboration between the 3D architects and the Simulation teams, without requiring the intervention of the System architects. It also helps to reduce the time taken by the iterations and thus find a 3D physical architecture validated according to its physical behavior and which meets the System architects' requirements.

Finally, the single bidirectional model transformation between the System model and the 3D and Simulation model ensures the consistency and traceability of the design models exchanged in the different steps of the conceptual design, from the specifications until the validation of the successful architecture by the System architects.

Finally, although this approach leads to several changes in the current industrial process, notably regarding the work usually done by the design actors (the 3D architects have to work together with the simulation teams to define a suitable 3D physical architecture), the resulting reduced design time is a larger benefit for all of them, since this approach reduces long, difficult iterations. Thus we can consider that requirement 2.2, Easy integration in the industrial design process, has been validated.

IV. Approach selection

These previous approaches, presented in the context of the multi-physical analysis will be studied in this section though focused on thermal analysis.

An analysis of these three approaches is provided in Table 7 with respect to the various derived requirements, defined previously in section I.

The following requirements: Data consistency (1.1.3), 3D architecture assessment according to the thermal and geometrical requirements (1.2.1 and 1.2.2), Easy integration in the industrial design process (2.2), and limited 3-year period (3.1) are satisfied by all the approaches. However, the difference between these approaches concerns the three other requirements.

The first approach meets neither the requirement of traceability nor easy interaction between the different actors. As this approach consists only of the automation of the current industrial process, it is natural that it conserves the same related issues. The requirement of easy interaction between the different actors is not fulfilled, since the actors work on their own, with oral or document exchanges, thus maintaining a long, costly iterative process to obtain the optimal concept architecture.

Table 7: Analysis of the three different approaches.

Name	ID	First approach	Second approach	Third approach
Data consistency	1.1.3	✓	✓	✓
Traceability management	1.1.4	✗	✓	✓
Easing interactions between design actors	1.1.5	✗	✓	✓
3D architecture assessment according to the thermal requirements	1.2.1	✓	✓	✓
3D architecture assessment according to the geometrical requirements	1.2.2	✓	✓	✓
Quick process execution	2.1	✗	✗	✓
Easy integration in the industrial design process	2.2	✓	✓	✓
limited 3-year period	3.1	✓	✓	✓

Neither the first approach nor the second approach fulfill the quick solving time requirement, compared to the third framework, in which thermal 3D architectures can be assessed in less time to converge towards a fitted concept architecture. Indeed, as the 3D architects and simulation teams work with the same platform on a single model in the third approach, they do not need to request any change from the systems architects. They can assess the 3D architecture under thermal constraints and check its compliance with the system architect's requirements themselves. Moreover, whereas with the first and second approach, a new 3D architecture implies the modeling of thermal constraints every time a simulation is launched, the third approach makes it possible to specify all the thermal constraints only once, provided that the same thermal phenomena are considered, to automatically launch simulation whatever the 3D architecture.

For these reasons, the third approach (Figure 23), which we have baptized Spatial Architecture, based on Multi-physics and Organization of Systems (SAMOS), has been chosen to address our research issue focused on thermal analysis.

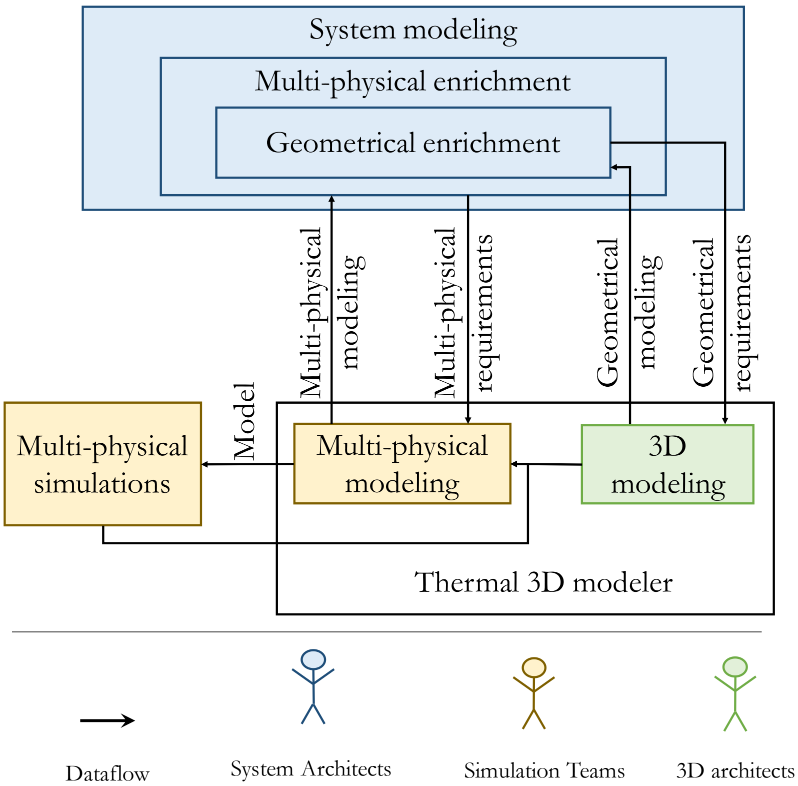


Figure 23: Overview of the SAMOS framework applied to thermal analysis.

V. Demonstrator implementation environment definition

The SAMOS process presented in Figure 23 above has been analyzed through the main implementation requirements.

This section will only partially detail the definition of the demonstrator's implementation environment, since a preliminary requirements analysis will be performed for each aspect covered in the following chapters, underlining the corresponding related induced requirements for the choices made for implementation:

- Chapter 3 will provide the implementation choices related to the geometrical modeling;
- Chapter 4 will underline the implementation choices linked to the thermal modeling;
- Chapter 5 will detail the choice of tools and the Thermal 3D Sketcher (which contain geometrical modelling, thermal modelling, and the Thermal 3D modeler) developments regarding all the implementation requirements.

Thus the next paragraphs of this section will describe the implementation requirements related to the choice of the System model language and to the implementation of the model transformation process.

A. Validation of the SysML Language for the System modeling

Concerning the choice of the System modeling language, the main derived requirements are shown in Figure 24.

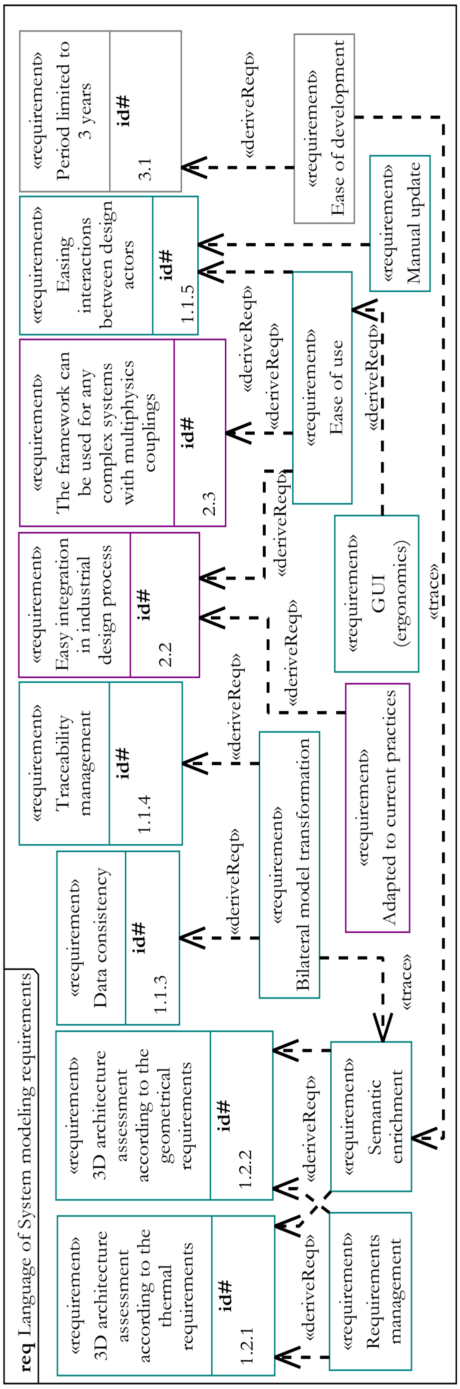


Figure 24: Flow chart of system modeling language requirements.

Table 8: Description of the additional requirements related to the System modeling language.

#Id	Name	Txt
1.2.12	Semantic enrichment	The System model has to integrate information related to the geometrical and thermal fields.
1.2.13	Requirements management	The System model has to integrate requirements management (specification and traceability processes).
2.6	Adapted to current practices	The System modeling language and the 3D modeling building must be adapted to current practices mainly based on commercial tools.

We then analyzed the relevance of the SysML language (Appendix 2) regarding these derived requirements in Table 9.

Table 9: Analysis of the system modeling requirements regarding the SysML language.

Name	ID	SysML
Data consistency	1.1.3	SysML ensures the unicity of the data whatever the diagram concerned.
Traceability management	1.1.4	Each SysML modeling element can be allocated to another or traced to several requirements
Bilateral Model transformation	1.1.6	Available in most tools integrating the SysML.
Semantic enrichment	1.2.12	Satisfied by SysML's extension (profile mechanism) building ability.
Requirements management	1.2.13	Thanks to its requirements diagram, SysML allows define and describing all the requirements and the corresponding traceability of other model elements.
GUI (ergonomics)	1.1.9	Ensured by a large variety of graphical diagrams.
Adapted to current practices	2.6	SysML is the language developed specifically for Systems Engineering applications, to support MBSE approaches and is therefore well-known by most System Architects.
Ease of development	3.2	The defininga new semantic field is easy thanks to new stereotyped elements (profile development ability). The PhD student already knows this language..

In the state of the art, Chandrasegaran et al (Chandrasegaran, et al., 2013) rise one main shortcoming of SysML is that it can lead to inconsistencies between different diagrams without a suitable methodology. However, many researchers have since proposed several MBSE methodologies to mitigate this drawback (Mhenni, et al., 2014), which can also be a benefit, allowing easy industrial process customization. Finally, SysML meets the previous requirements and is thus validated as the modeling language to support the System modeling.

Regarding the "Semantic enrichment" requirement, SysML addresses it through the development of specific SysML extensions for geometrical and thermal enrichments, respectively: the thermal extension will be based on the geometrical one, since all physical behaviors rely on specific geometries (of components and of physical phenomena). Nonetheless, the transformation of the System model could be processed by using SysML language without any extension, although this would make the transformation process more complex. Indeed, the stereotypes and tags developed in the extensions will ease the modeling processing, since the extensions define the available syntax elements necessary to generate a viable architecture for users so they can build a System model that functions in the model transformation process. SysML extensions require that System model designers (System architects) build a model with a structure that agrees with the specific meta-model defined, to ensure successful model transformation. However, as it is also important that the Simulation teams and 3D architects can themselves also specify the physical and geometrical requirements and the reverse model transformation (from the thermal 3D architecture model into the SysML model). The semantics of the SysML must be enriched to ensure that all the information can be traced back in a common description.

Moreover, these extensions will efficiently support the modification of the metamodel needed during the model transformation process to ensure data exchange between the tools.

B. Selection of the model transformation process

As the model transformation process involved will address two specific software tools, the model transformation implemented is linked to a PSM-PSM transformation (Kellner, et al., 2015). Thus the following state of the art of existing model transformation processes is proposed for the PSM-PSM transformation. Furthermore, we address exogenous transformations (Mens & Van Gorp, 2006) as the metamodels involved in the SAMOS model transformations relate to different fields (System engineering, 3D architecture and thermal simulation).

Regarding the existing corresponding (PSM-PSM and exogenous) model transformation approaches, Czarnecki and Helsen (Czarnecki & Helsen, 2003) cited the VIATRA, ATOM, GreAT, UMLX and BOTL approaches. The most well-known VIATRA (VIsual Automated model TRAnsformation system) process was described by Varró at al. (Varró, et al., 2002) in 8 steps:

- Description of target and sources metamodel.
- Standardization of the representation of the metamodel described using the XML based language
- Description of the model transformation rules. This step is supported by a graph transformation description. These rules can be described in UML and exported in XML-based language.
- Provision of a proof that these model transformations are correct and complete.
- Generation of the transformation rules.
- Implementation of the transformation engine using a low abstraction level language (i.e. C++ or Java).
- Test of the model transformation using several scenario models.
- Analysis of the results and return to the third step if need be.

Kappel et al. (Kappel, et al., 2012) described this model transformation using two main phases:

- Modeling: in this step, the user creates a model with concrete syntax, named the initial model.
- Configuration & Generation: in this phase, an initial version of the transformation is inferred by both analyzing the initial and revised models.

The process proposed by Kappel et al. (Kappel, et al., 2012) is given in Figure 8

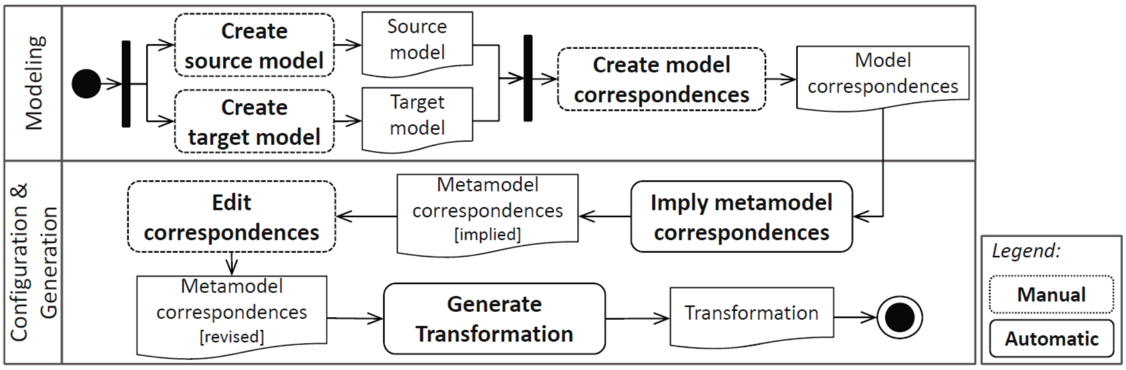


Figure 8: Exogenous model transformation proposed by Kappel et al. (Kappel, et al., 2012)

A comparison between these existing exogenous model transformation methods will be presented, after having detailed the additional requirements needed to select the model transformation implementation method (Figure 25). The requirements that will be used as criteria for the evaluation of the existing methods are outlined in red.

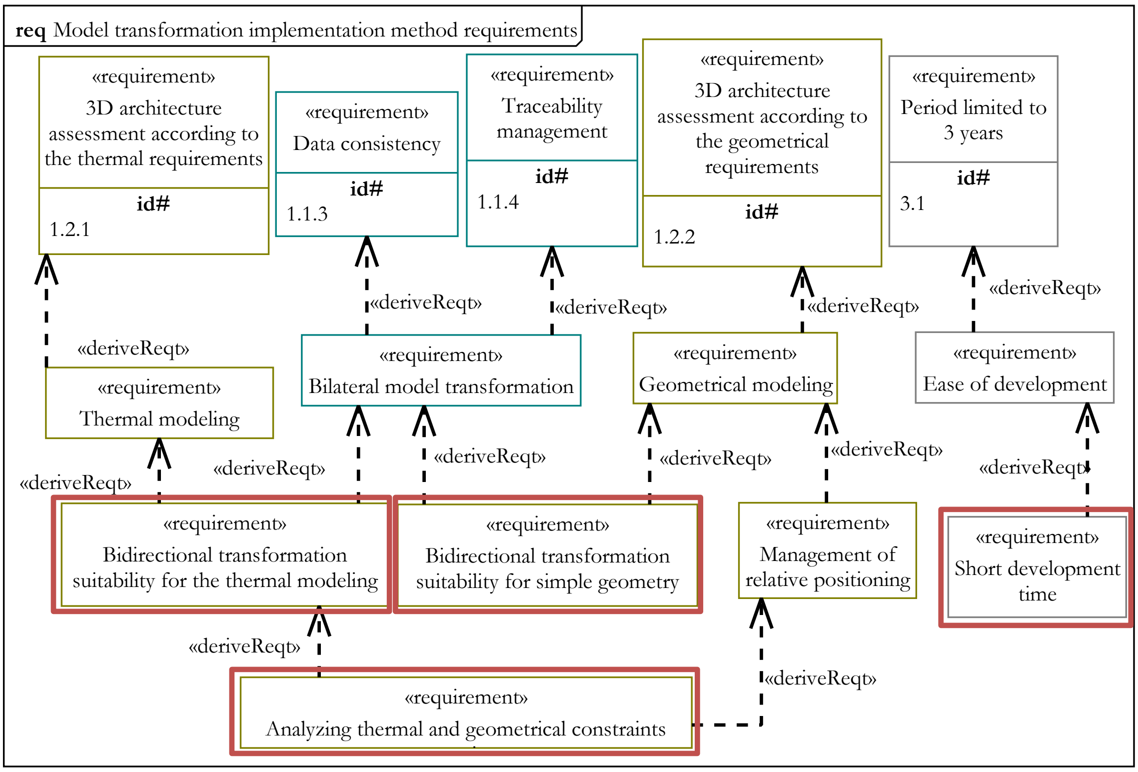


Figure 25: Model transformation implementation method requirements

The additional requirements are described in the following table (Table 10).

Table 10: Additional requirements related to the model transformation process.

#Id	Name	Txt
3.1.3	Short development time	As described previously, the model transformation has to be developed within a short period, since the development must be performed during the period devoted to the PhD thesis.
1.2.8	Management of relative positioning	Management of relative positioning constraints will be ensured to facilitate the definition of the 3D architecture.
1.2.9	Bidirectional transformation suitability for simplified geometry	The model transformation process will be adapted to an exogenous bidirectional M2M transformation for simplified geometry. A simplified geometry is a primitive geometry (cylinder, sphere, cone, and elements composed by planes, etc.). The spline model will be considered as a complex geometry.
1.2.10	Bidirectional transformation suitability for thermal modeling	The model transformation process will be adapted to an exogenous bidirectional M2M transformation also for thermal modeling, which implies solving second order partial differential equations.
1.2.11	Analyzing thermal and geometrical constraint equations	The model transformation will be able to analyze thermal and geometrical constraint equations.

A comparison between these two existing exogenous model transformation methods regarding the corresponding specific requirements (described in Figure 25 and Table 10), is presented in Table 11.

Table 11: Evaluation of existing exogenous methods regarding the requirements of the model transformation implementation method.

Criteria \ Exogenous existing methods	Kappel et al. (Kappel, et al., 2012)	VIATRA (Varró, et al., 2002)
Short development time	✓	✗
Suitable bidirectional transformation for simplified geometry	✓	✓
Suitable bidirectional transformation for thermal modeling	✓	✓
Analysis of thermal and geometrical constraints equations	✓	✓

Although the VIATRA framework proposes an interesting process, as it ensures that the model transformation is applicable for any source model, this method requires a mathematical demonstration that the model transformation is injective. Varro et al. explained that “any formal proof of correctness and completeness of these transformation scripts is almost impossible”, thus this demonstration can take a very long time. Thus the approach of Kappel et al. (Kappel, et al., 2012) is finally that which is best-adapted to our requirements, notably according to the short development requirement, and it will be used to implement the SAMOS demonstrator.

VI. Conclusions

According to the requirements derived from the expression of needs provided at the beginning of the chapter, we chose the collaborative model transformation structure to manage the data and human exchanges between the various actors involved in the conceptual design. Then, we compared three alternatives to ensure data exchange between the different actors for the system architecture assessment during the conceptual design phase, in the context of a multi-physical analysis. The third approach, which we call Spatial Architecture based on Multi-physics and Organization of Systems (SAMOS) was chosen to solve our research issue focused on thermal analysis. The analysis of the main implementation requirements of SAMOS allowed us to validate the SysML language for the system model and select the model transformation process proposed by Kappel et al (Kappel, et al., 2012). The additional requirements for each aspect covered in the following chapters and the corresponding options for implementation will be described in each of these chapters respectively.

Since 3D architecture is the focal point of the SAMOS approach, we will describe the geometry modeling in the next chapter.

Chapter 3 – Geometrical modeling

I. Expression of needs 55

II. Existing geometrical modeling approaches.................... 56

A. Geometry modeling...57

1. Surface modeling...57

2. Solid modeling..58

B. Topology modeling...61

III. Analysis for the choice of geometrical modeling in SAMOS... 64

A. Choice of geometry and topology modeling for geometrical modeling...64

1. Geometry modeling...64

2. Topology modeling 65

B. Integration of face modeling for SAMOS.................................66

1. Face construction process..66

2. Related geometrical views...66

IV. The GERTRUDe SysML extension............................... 67

A. Definition of the GERTRUDe metamodel 67

B. Example of a description of a finite cylinder geometrical........68

V. Geometrical information model transformation strategy 70

A. Different M2M transformation strategies.................................70

B. Analysis of the "Neutral format usage" strategy applied to SAMOS 71

C. Analysis of the "Direct translation" strategy applied to SAMOS 73

D. Selection of the model transformation strategy for SAMOS 73

VI. Conclusions 74

After having described the initial requirements of the SAMOS framework to be developed in chapter 2, this chapter will focus on geometry modeling. Geometry plays an important role in the development of the SAMOS Platform, since it will provide the basis for the whole thermal modeling and the 3D architecture.

The first section provides the derived requirements related to the geometrical modeling, including the geometry and the topology modeling. Then, a state of the art is given presenting, on the one hand, the main existing geometry modeling methods by distinguishing surface and solid modeling, and on the other hand, topology modeling which describes the connectivity and associativity of the geometrical entities. Afterwards, these approaches are analyzed regarding their ability to meet the geometry modeling and topology requirements for SAMOS, before presenting the construction process of shape modeling based on faces. The fourth section presents the metamodel of the Geometrical Extension Related to the TTRS Reference for a Unified Design (GERTRUDe) which will be implemented in the SysML environment. Finally, several strategies for managing the transformation of the geometrical model information from GERTRUDe to the 3D CAD tool are compared, in order to choose that which will be applied in SAMOS.

I. Expression of needs

As 3D CAD tools are based on a given geometry model, the choice of the geometry modeling theory to be implemented in the demonstrator must also meet the specific requirements described in Figure 26. Then the fulfillment of these requirements (identified with a red contour: solid line when related to the geometry and dotted line when related to the topology) by certain existing geometry models will be analyzed as selection criteria for choosing the geometry model best adapted to the objective.

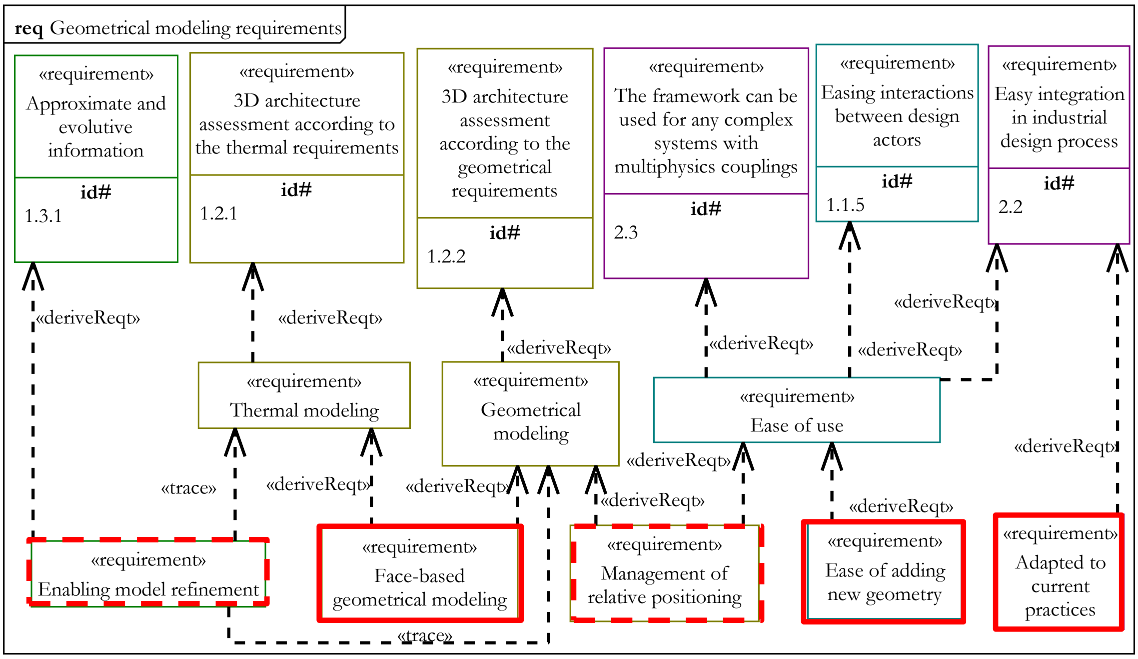


Figure 26: Geometry modeling requirements.

This diagram shows that three new requirements related to the geometry modeling theory to be chosen have been added (Table 12).

Table 12: Additional requirements for geometrical modeling

#Id	Name	Description
1.2.10	Face-based geometrical modeling	As the thermal modeling is based on faces (this aspect will be detailed in chapter 4), a geometry model based on faces will be useful for adding thermal boundary conditions.
1.1.11	Ease of adding new geometry	The geometrical model must be easy to use and allow adding new geometry.
1.3.2	Enabling model refinement	The thermal model must be refined when the quality of the results needs to be improved.

Finally, these requirements will be used as criteria to support the choice of the geometrical modeling approach (III.A).

II. Existing geometrical modeling approaches

In this section, we will give an overview of existing geometry modeling.

A complete representation of a geometrical object includes both topology and geometry information. Geometry describes the shape and dimensions of the object, whereas topology describes the connectivity and associativity of the object entities (Figure 27). Topology thus determines the relations between the object entities.

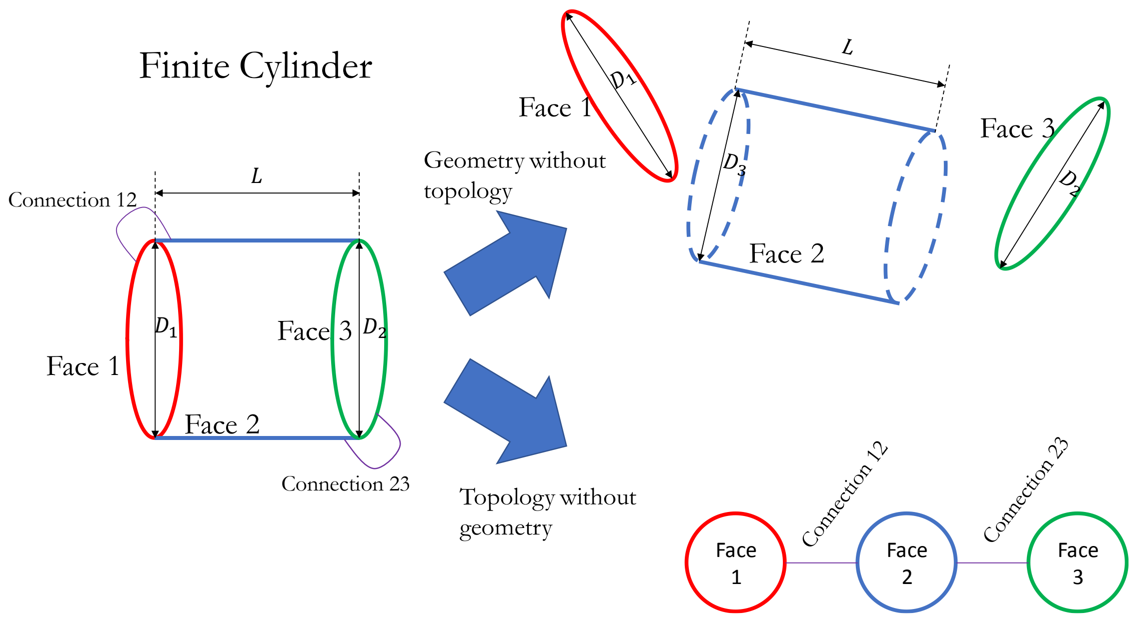


Figure 27: Example of geometry and topology modeling for a finite cylinder.

In the rest of this chapter we consider that a face is a surface with a contour (i.e. a finite surface), whereas a surface is infinite.

A. Geometry modeling

Usually, geometry modeling can be performed in two ways. The first consists in modeling the geometry based on surface modeling, whereas the second, called solid modeling (Weiler, 1986), represents geometry based on the generation of volume.

1. Surface modeling

a. Wireframe modeling

Wireframe modeling (Weiler, 1986) consists in modeling objects by edges (Figure 28). The edges can be arcs, circles, conics or other curves. In a 3D wireframe model, an object is not modeled as a solid. The vertices that delimit the edges are defined as a set of points. Therefore, face identification can be ambiguous (Figure 29).

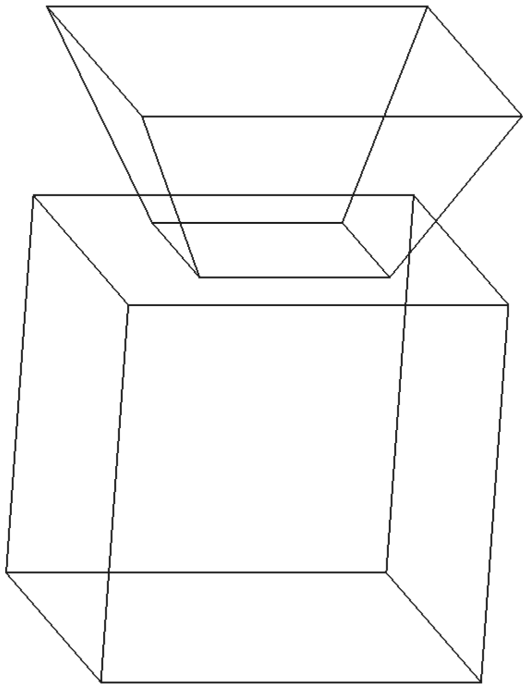


Figure 28: Example of wireframe modeling

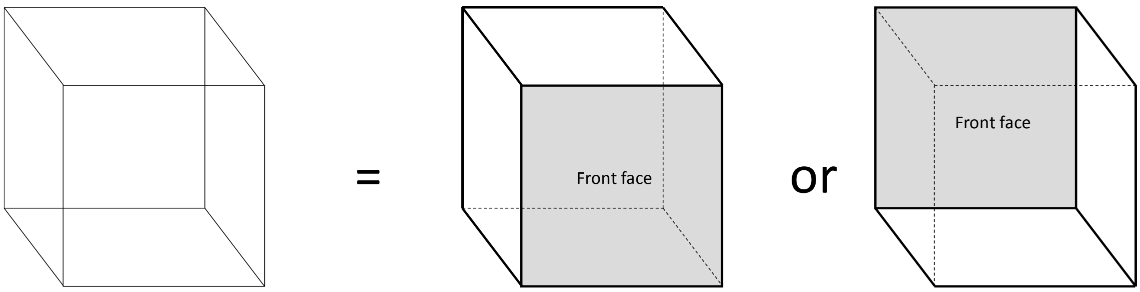


Figure 29: Face identification, the ambiguity of wireframe modeling.

b. Surface modeling

Since this difficulty may lead to ambiguous understanding of a 3D geometrical object, it is necessary to carry out surface modeling developments. Surface modeling (Weiler, 1986) is based on input points or curved net interpolation using wireframe modeling to draw the contour of the surfaces on the one hand, and the set of corresponding faces on the other hand. These surfaces can be modeled in different ways; they can be determined analytically or based on Free-form, curved or sculptured surfaces, etc.

2. Solid modeling

Surface modeling does not include the definition of the object volume, i.e. the object is defined by the volume space contained within the defined boundary of the object. The boundary of the model separates the interior and exterior of the modeled object. In this case, the half space concept can be used to transform a surface model into a solid model.

a. Half space concept

This method is used as the basis of solid modeling. It consists of unbounded geometric entities that divide the representation space into infinite portions, one filled with material and the other empty. Then, the surfaces can be considered as half-space (Kada, 2006) boundaries and half spaces can be considered as directed surfaces (Figure 30.). An object is defined by the volume space contained within the defined boundary of the object. Indeed, each surface is oriented using a director vector, which is defined as outgoing from the solid. Introducing the direction into the modeling thus enables the topological information to be stored in the geometrical model.

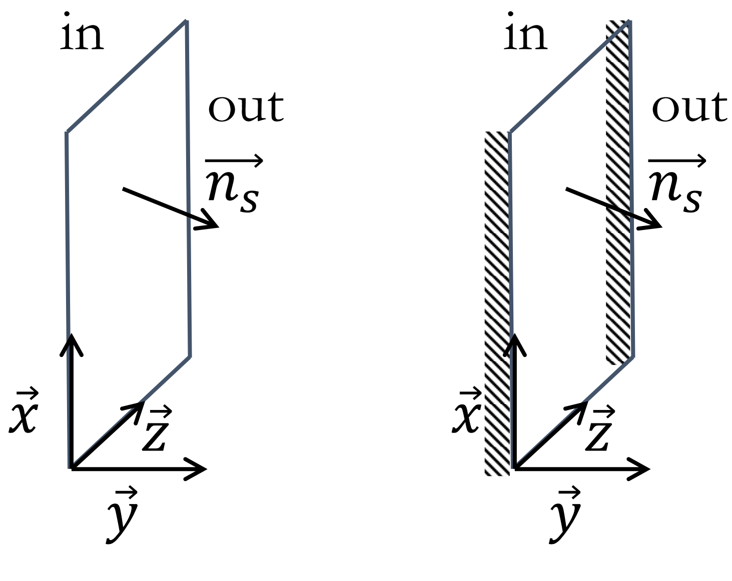


Figure 30: Half space based on surface modeling

By specifying different boundary surfaces, we can obtain any half-spaces that can be combined using Boolean operations in building block fashion, in order to build various solids. This construction is described in Figure 31. The most commonly used half-spaces are planar, cylindrical, spherical, conical, and toroidal.

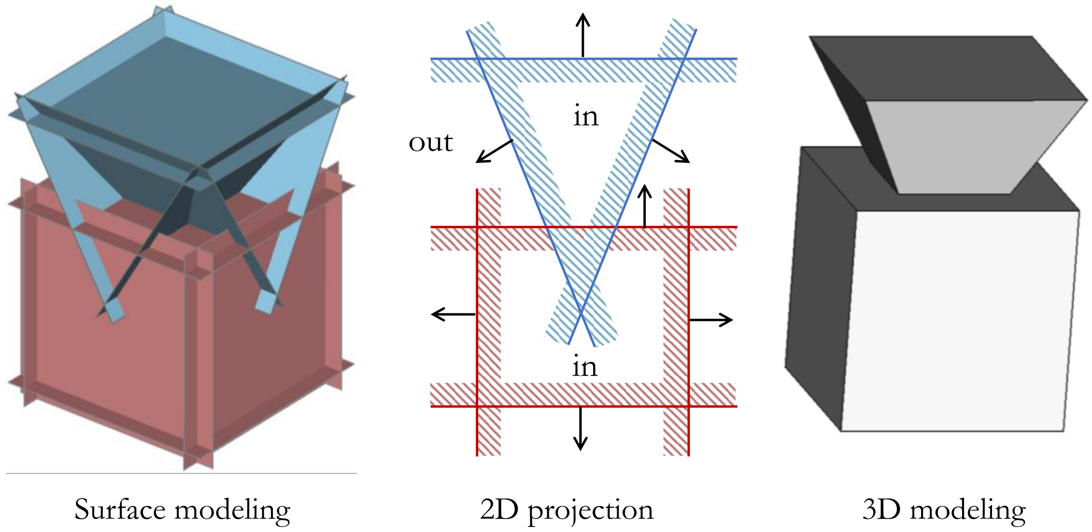


Figure 31 Example of the half space concept.

Among the various existing solid modeling approaches based on the half space concept, here we detail the two-main well-known 3D CAD modeling approaches used to build geometrical elements: **Constructive Solid Geometry** (CSG) and **Boundary Representation** (B-Rep).

b. CSG

The Constructive Solid Geometry (Hughes, et al., 2014) is the simpler approach for geometry modeling. A set of primitive geometries is defined first: cuboids, cylinders, prisms, pyramids, spheres, cones. These primitive geometries are described by a shape and the corresponding parameters. Then, all these simple geometrical elements can be combined using Boolean operations, like union, intersection and difference operations (Figure 32).

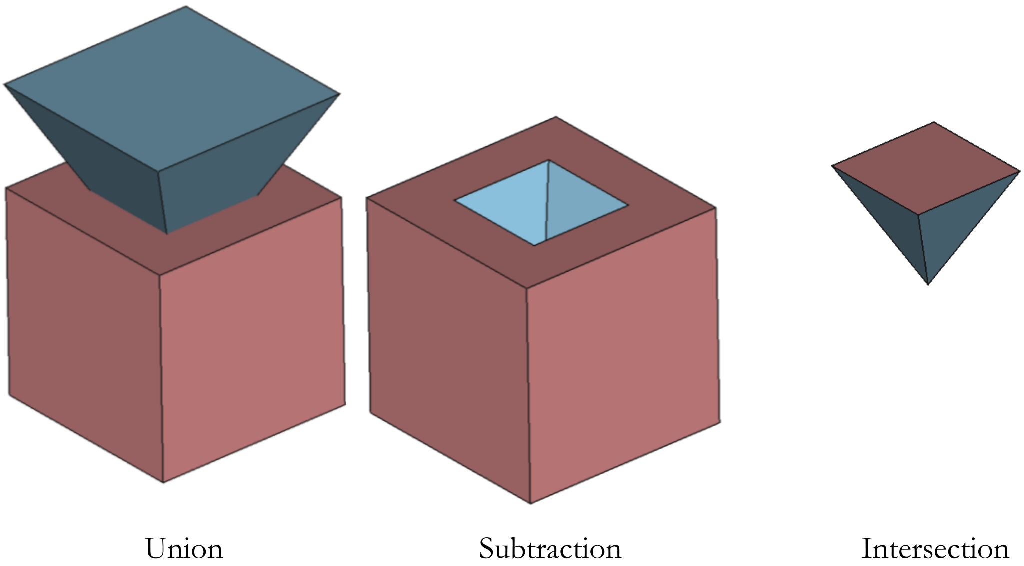


Figure 32: Example of Constructive Solid Geometry operations

These associated elements can in turn be combined with any other geometrical element. This allows generating geometries built with simple geometry. Nevertheless, this representation remains limited in the case of building complex geometry. The successive steps of geometry construction are then stored by the corresponding Boolean tree. Each tree leaf corresponds to a primitive geometry. A vertex corresponds to a geometry composed of different geometries.

As tree representation is adapted to the implementation of classes (since a class can contain many different classes), CSG modeling could be well-adapted to representation using SysML language. Moreover, the definition of the initial simple geometries of the CSG approach is easy to describe in a library (like any object language) in the SysML language. Warniez et al. (Warniez, et al., 2014) proposed to define a library of simple geometries with predefined parameters in SysML. Although Boolean operations have not yet been implemented, a combination of such geometries including these operations could be used as CSG Modeling.

c. B-Rep

Boundary Representation provides another representation of geometry which has the advantage of combining both geometrical and topological definitions (thanks to faces, vertexes, and edges). With this type of geometrical modeling, a component is represented using a set of faces called Shell, the faces are delimited by edges and the edges are defined by vertices (International Standard Organization, 2000). Finally, volume is built by the union of closed faces (Figure 33).

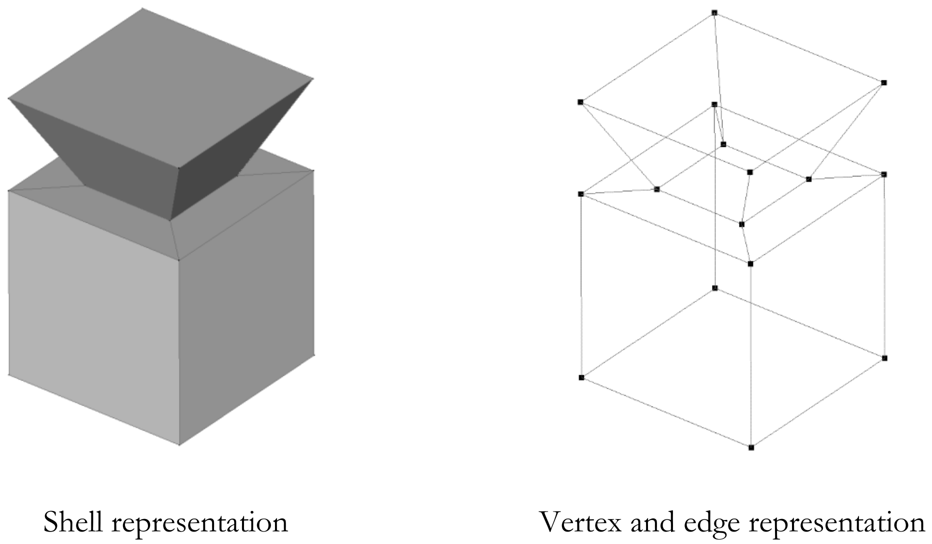


Figure 33: Example of B-rep modeling

B-Rep representation is frequently used, since it can address all kinds of solids, and notably complex geometry. The B-Rep representation model has already been implemented in a UML profile, using Airplane Design Language (Bohnke, et al., 2009). However, B-Rep can be very complex when used to represent geometry in SysML, as each vertex modeled as a class is represented as a three-coordinate object. Thus a simple square, which has 8 vertices, 12 edges, and 6 faces, must be modeled in SysML by at least 26 classes. Such an approach will provide a considerable amount of data to manage even for only one component.

Finally, CSG representation allows quickly building certain geometries through the composition of simple geometric elements, but it cannot address complex geometry. B-rep representation addresses any kind of geometry but it requires the management of a vast amount of data.

B. *Topology modeling*

In this paragraph, we compare several different methods for relative positioning management.

The CSG and B-rep representations described above do not implement specific positioning constraint management. Thus 3D CAD tools that implement these approaches have to be coupled with other existing relative positioning constraints.

For example, Kim et al. (Kim, 2014) proposed ontologies in order to describe CAD modeling based on six types of spatial relationships supporting geometrical constraints. The constraints between CAD components are given by a metamodel. Kim et al. used the constraint modeling approach proposed by Ambler and Popplestone. They considered 6 constraints available to assemble one element relative to another element. These constraints depend on geometry; a degree of freedom is associated for each constraint in order to solve the position and the orientations of the components. These constraints do not depend on the geometry. For example, there is no difference between a contact between two spheres, and a contact between a cylinder and a plane. Thus it is not possible to create all the possibilities of assembly.

In order to solve this issue, CAO et al. (Cao, et al., 2013) proposed to model an assembly with 6 constraints. The difference between the constraints proposed by Kim et al. is that the contact relation is broken down into three cases. According to the type of contact, for example, a contact between two faces and a contact between a cylinder and a face are differentiated. However, some constraints are still missing, for example, the angle between a cylinder and a plane. Angle constraints are only defined between two planes.

The TTRS concept, proposed by Rivière and Clément, and then Serré (Clement, et al., 1998), classifies surfaces into seven different subsets or classes, according to their kinematic motion invariances (Figure 34). These classes are: spherical, plane, cylindrical, helical, revolute, prismatic, and complex. For each class, it is possible to associate one Minimal Geometric Reference Element (MRGE), which is the minimal combination of the following simpler geometric objects, named Reduced Geometric Element (RGE): plane, line and point. This reduced geometrical representation allows easier object positioning in Euclidian space.

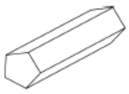
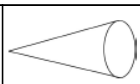
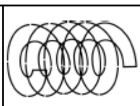
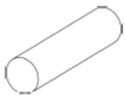
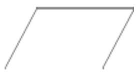
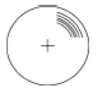
TTRS classes	 Complex	 Prismatic	 Revolute	 Helicoid	 Cylindrical	 Plane	 Spherical
Invariance degree	0 (identity)	1 translation	1 rotation	1 rotation & 1 translation combined	1 rotation & 1 translation	1 rotation & 2 translations	3 rotations
MRGE	Point		Point				Point
	Line	Line	Line	Helix	Line		
	Plane	Plane				Plane	

Figure 34: TTRS and MRGE definition

Like the tree-based building in the CSG approach, the TTRS theory allows the composition of other TTRS. An example of this composition process is given for a finite cylinder whose TTRS belongs to the revolute class, as it has a rotation invariance. The finite cylinder is composed of an infinite cylinder TTRS cut by two infinite Plane TTRS (Figure 35).

Each TTRS belongs to a class which is described by an MRGE composed of RGE. Thus, the resulting TTRS of the Finite cylinder, belonging to the Revolute class, has an MRGE {Point-Line} which can be used to determine its relative positioning (Table 13).

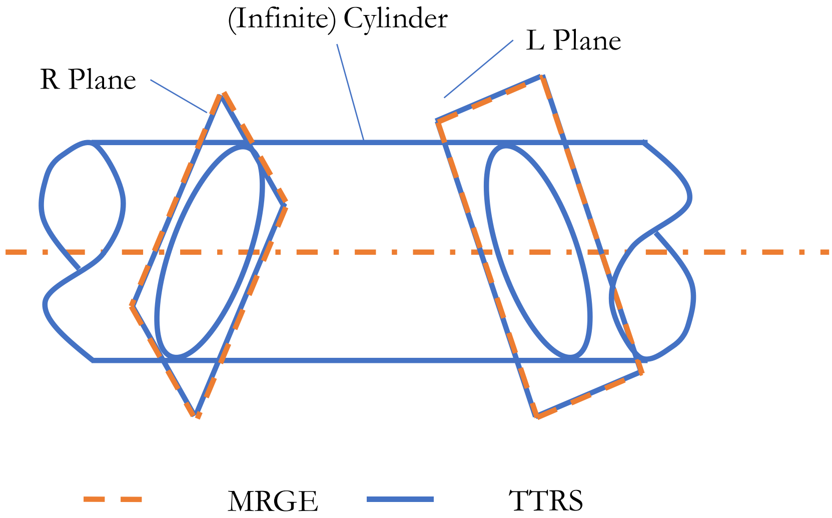


Figure 35: TTRS modeling of a finite cylinder.

TTRS modeling manages relative positioning between two surfaces of one (as a TTRS can be composed of TTRS) or different components, through a set of 13 geometrical constraints (Figure 4) applied to the MRGE addressed. These constraints allow positioning and orienting the MRGE of the TTRS of a component relatively to another TTRS or between two TTRS contained in the components.

Table 13: The 13 TTRS constraints (Clement, et al., 1998)

Reclassing cases of MRGE and induced constraints	Line (Cylindrical)	Plane (Plane)	Point (Spherical)
Line (Cylindrical)	D1=D2 : C11 D1//D2 & D1≠D2: C12 Else: C13	D2⊥P1: C8 D2 // P1: C9 Else: C10	O1∈D2: C4 Else : C5
Plane (Plane)		P1//P2: C6 Else: C7	C3
Point (Spherical)			O1 = O2: C1 Else : C2

An example of a TTRS constraint is given with the previous example of the finite cylinder: in this example, a C06 (two plane parallelism) constraint is applied between two plane TTRS (Figure 36)

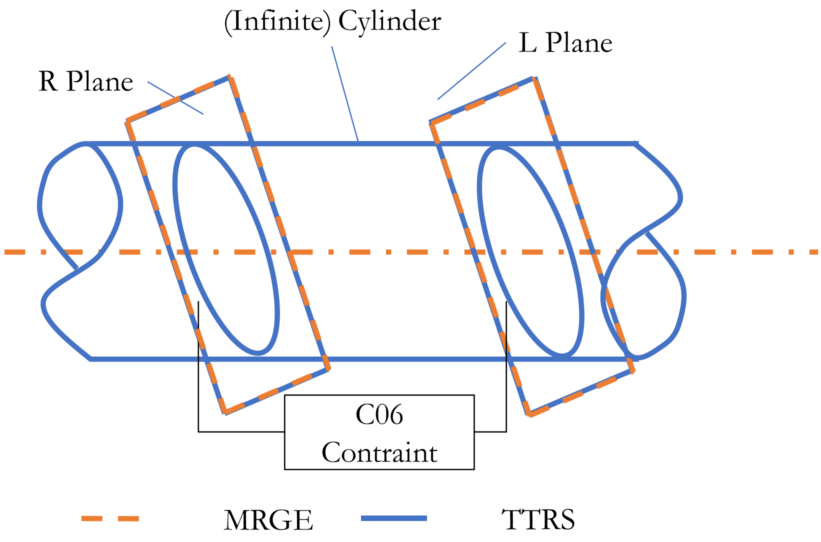


Figure 36: Example of the C06 constraint between the MRGE of the two plane TTRS.

The TTRS composition is similar to a binary tree based on two TTRSs and the associated constraint between them. As the TTRS theory allows the automatic determination of the TTRS class of the resulting root, which is called the TTRS reclassification, it allows building any geometrical entity based on TTRS, and identifying its final resulting TTRS. However, as the reclassification between two surfaces can be carried out between any pair of TTRS, there is no uniqueness of the binary decomposition tree. For example, to model a Finite Cylinder based on the TTRS, we could consider these two associations:

- either $\left((Right\ plane\ U\ Left\ plane)\ U\ Cylinder\right)$
- or $\left((Right\ plane\ U\ Cylinder\ )\ U\ Left\ plane\ \right)$

Furthermore, as TTRS is defined as an infinite surface, we must use the half space concept, in order to take into consideration the finite surfaces (faces) defining the volume of the architecture components that will be positioned in the 3D space and thermally interact with each other.

Finally, to address the "Enabling model refinement" requirement related to the geometrical modeling of the components (made of faces) while taking into account the previous "non-uniqueness of the TTRS-based composition", the TTRS-based geometrical construction of surfaces will be implemented in arbitrary order.

These different decomposition levels are represented in Figure 37.

The TTRS constraints were initially developed for the tolerance modeling of a component (International Standard Organization, 2005) and then extended to pseudo-TTRS (Clement, et al., 1996) to support the tolerance modeling of parts assembly. Pseudo-TTRS represent contacts (with or without clearance) between two surfaces of two different parts. Indeed, for tolerance modeling, the topological change of the number of surfaces to be taken into consideration is crucial (e.g. assembly leading to the clearance of a surface between two parts in contact).

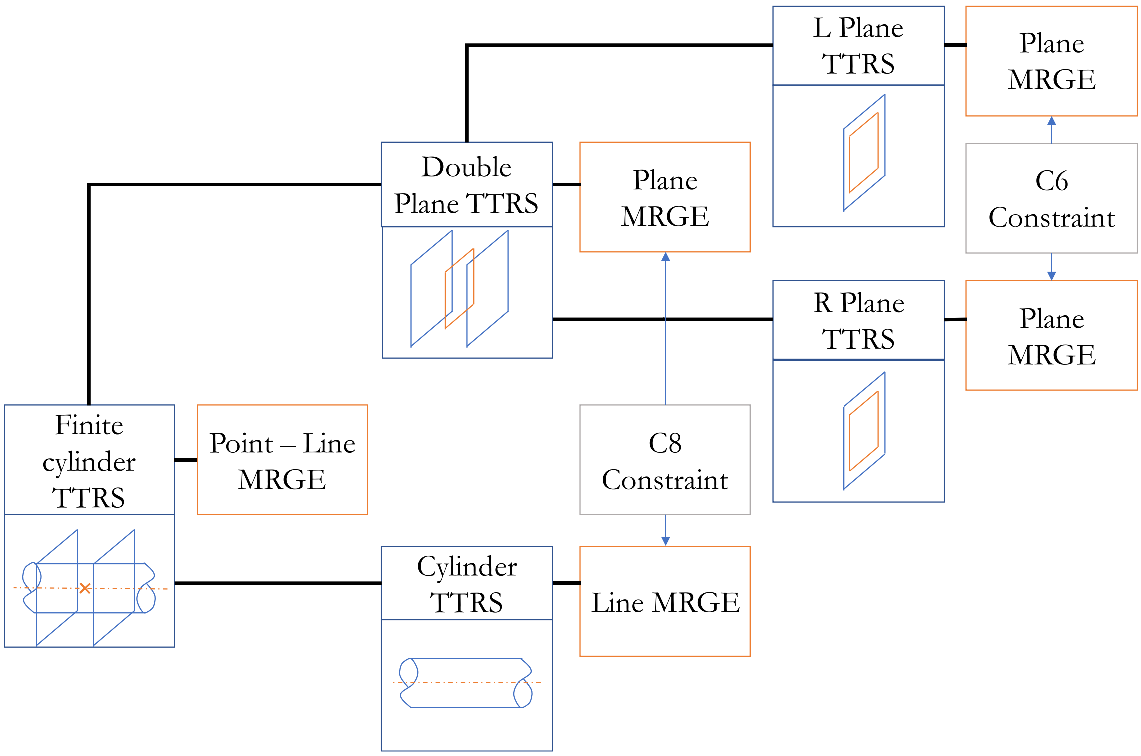


Figure 37: Multi-scale TTRS decomposition for an hollow cylinder

III. Analysis for the choice of geometrical modeling in SAMOS

A. Choice of geometry and topology modeling for geometrical modeling

1. Geometry modeling

The previous state of the art was provided to address the "geometry modeling" requirement of SAMOS. The analysis of these different approaches regarding the SAMOS requirements previously defined (in section I) will support the selection of the most suitable geometrical modeling approach.

Three main requirements will be assessed to compare the previous existing geometry modeling approaches:

(i) the geometry modeling approach will be based on face modeling, in order to support thermal modeling;
(ii) the addition of new geometry should be easy to perform;
(iii) the geometry modeling should be adapted to current practices.

Table 14 summarizes the evaluation results of the three methods presented previously according to these requirements.

According to this table, none of the existing geometry models meets all the requirements. The CSG and wireframe methods are not face-based. Considering the wireframe, surface modeling and B-rep methods, the addition of new geometry is not easy to perform, since the generation of each face implies the use of vertices and edges, which significantly increases the complexity of the design.

Table 14: Evaluation of geometry modeling approaches according to the requirement specifications.

Requirements \ Geometry modeling	Wireframe	Surface modeling	CSG	B-Rep
Face-based geometry modeling	✘	✓	✘	✓
Easy to add new geometry	✘	✘	✓	✘
Adapted to current practices	✘	✓	✓	✓

2. Topology modeling

The requirements related to the geometrical modeling approach are given in section I. Two main requirements will be assessed to compare the previous existing topology modeling approaches presented in section II.B:

(i) it has to manage relative positioning,
(ii) and it should enable model refinement.

Table 15 summarizes the corresponding evaluation results.

Table 15: Evaluation of topology modeling approaches according to SAMOS requirement specifications.

Requirements \ Topology modeling	Kim et al	Liu et al.	TTRS
Relative positioning	✓	✓	✓
Model refinement	✘	✘	✓

Regarding the topology related requirements, the modeling approaches proposed by Kim et al. and Lui et al. are based only on constraints and do not support the refinement model, contrary to the TTRS concept, in which each TTRS can be composed of other TTRS. In addition, TTRS offers the most complete constraints modeling approach with the least parameters and it allows positioning any geometry thanks to MRGE (which ensures that few data are integrated in the System model in SysML). Consequently, we choose the TTRS modeling approach for the topology.

TTRS is adapted to topology modeling, but it does not include any geometry data although the GPS standard (International Standard Organization, 2005) introduces several "intrinsic parameters". Then, each TTRS can be completed with intrinsic parameters, for example, the radius for a sphere or a cylinder. These parameters add a geometrical (dimensional) description to this topological modeling approach.

Nonetheless, as TTRS is based on (infinite) surface modeling and as thermal modeling needs faces with finite areas as described in chapter 4, we require a suitable face modeling approach.

B. *Integration of face modeling for SAMOS*

1. Face construction process

The process employed to generate faces is described in Figure 41. It allows constructing faces from a set of TTRS enriched with their intrinsic parameters, implemented previously implemented in a SysML extension (§ IV). Based on the half space concept, the infinite surfaces (i.e. TTRS) topologically associated via several constraints are then transformed into faces in the 3D modeler environment that can construct faces from the intersections of surfaces. The geometrical parameters of the faces (area and dimensions) are then returned in SysML, in order to support thermal analysis traceability.

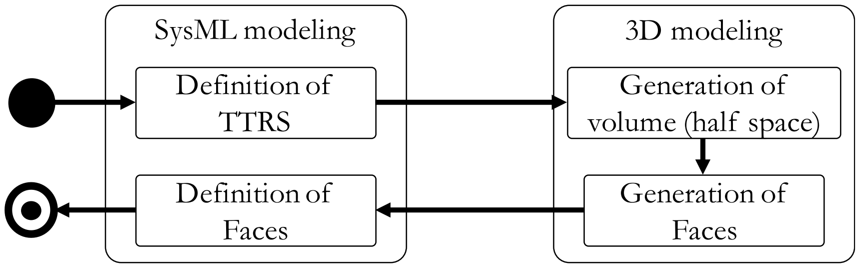


Figure 38: Face generation process.

2. Related geometrical views

Thermal modeling requires face modeling, since thermal boundary conditions are added to the faces. The model refinement requirement for improving the accuracy of the thermal calculations can be met by introducing specific views. The first addresses the shell surface view of the whole component, whereas the second view consists in decomposing the faces into different independent faces. In addition, each face is associated with a TTRS that supports its construction, based on the construction process described previously (Figure 42).

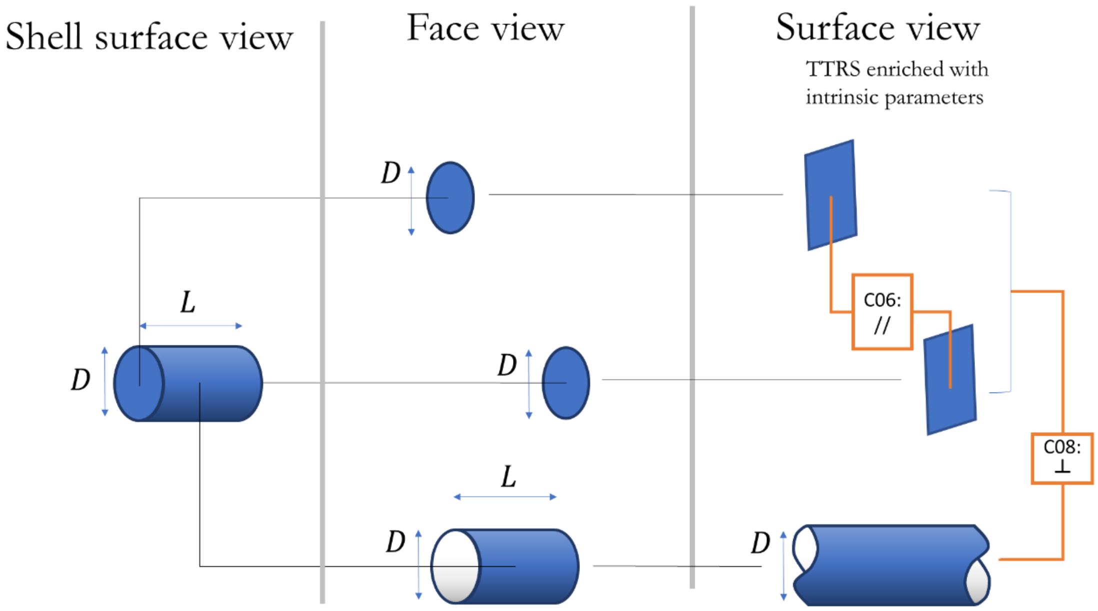


Figure 39: Different geometrical views.

IV. The GERTRUDe SysML extension

As the geometrical model transformation has to manage a model transformation between a SysML model into a 3D CAD model, it is necessary to develop a SysML extension to support it. This SysML extension is called GERTRUDe (**G**eometrical **E**xtension **R**elated to a **T**TRS **R**eference for a **U**nified **De**sign).

A. Definition of the GERTRUDe metamodel

The GERTRUDe SysML extension has to support the definition of both the TTRS with intrinsic parameters and the face definition. Prior to that, it is important to describe the corresponding requirements and hypotheses:

a. GERTRUDe can be used before or after the SysML modeling step. The system architects should not only be able to create their architecture directly from the geometrical elements, but also to integrate the geometry after modeling the architecture in SysML. This implies compatibility between the GERTRUDe stereotypes and certain existing SysML models.
b. GERTRUDe HMI should be easy to use, since system architects are not experts in geometry, but they may have to specify certain geometrical requirements. Thus a useful HMI has to be developed.
c. The enriched model must be compatible to export or import geometrical data to/from a 3D CAD modeler for the future developments of our work.

The integration of the TTRS theory-based surface representation in a SysML model then requires the integration of the TTRS metamodel in the new SysML extension. Based on the TTRS modeling approach described by Clement & al. (Clement, et al., 1996), we have considered intrinsic and situation features resulting from the GPS standard (International Standard Organization, 2005), simply as generic "dimensional parameters" in the TTRS Block attributes (Block property values). The explanation of this simplification is that these dimensional parameters, whether originating from a TTRS itself (intrinsic feature), for example, radius for the cylinder, or resulting from a TTRS constraint, applied between the MRGE of two TTRS (situation feature), and length considered as the distance between both parallel planes of a cylinder, are required to define a finite volume.

The metamodel of GERTRUDe is given in Figure 40.

The GERTRUDe metamodel is first described with the stereotype "*Component*". This stereotype addresses all the physical elements that have a geometry. This stereotype can be added to each UML class, including the already stereotyped UML class (such as the SysML "Block"). This ensures that a SysML model initially built without GERTRUDe can thereafter be enriched, for example, by stereotyping its existing blocks with the "Component" stereotype to transform them into 3D objects.

A component is composed of only one 3D object. The "*3D object*" stereotype is composed of only one TTRS (the resulting reclassified one).

The "*TTRS*" stereotype is composed of zero or more TTRS (zero for example for a plane) and belongs to a "*TTRS Class*" class, according to its kinematic invariance degree. TTRS can also be composed of constraints to position and orientate the TTRS composing them.

An "*MRGE*" belongs to a "*TTRS Class*" and is composed of one, two or three Relative Geometrical Elements (*RGE*) which can be Plane, Line or Point.

The "*3D object*" can contain zero or one shell: zero shell at the first step of TTRS definition (Figure 38), and one shell is added during the face construction process after solving in a 3D CAD environment.

The "*Shell*" describes the whole surface geometry of the component. The shell is composed of "Face(s)".

The "*Face*" has different parameters, such as the area and dimensions of the contour (for 1D convection), which will be used for the thermal analysis in chapter 4. Each "*Face*" is linked to its corresponding related TTRS which represents their topology.

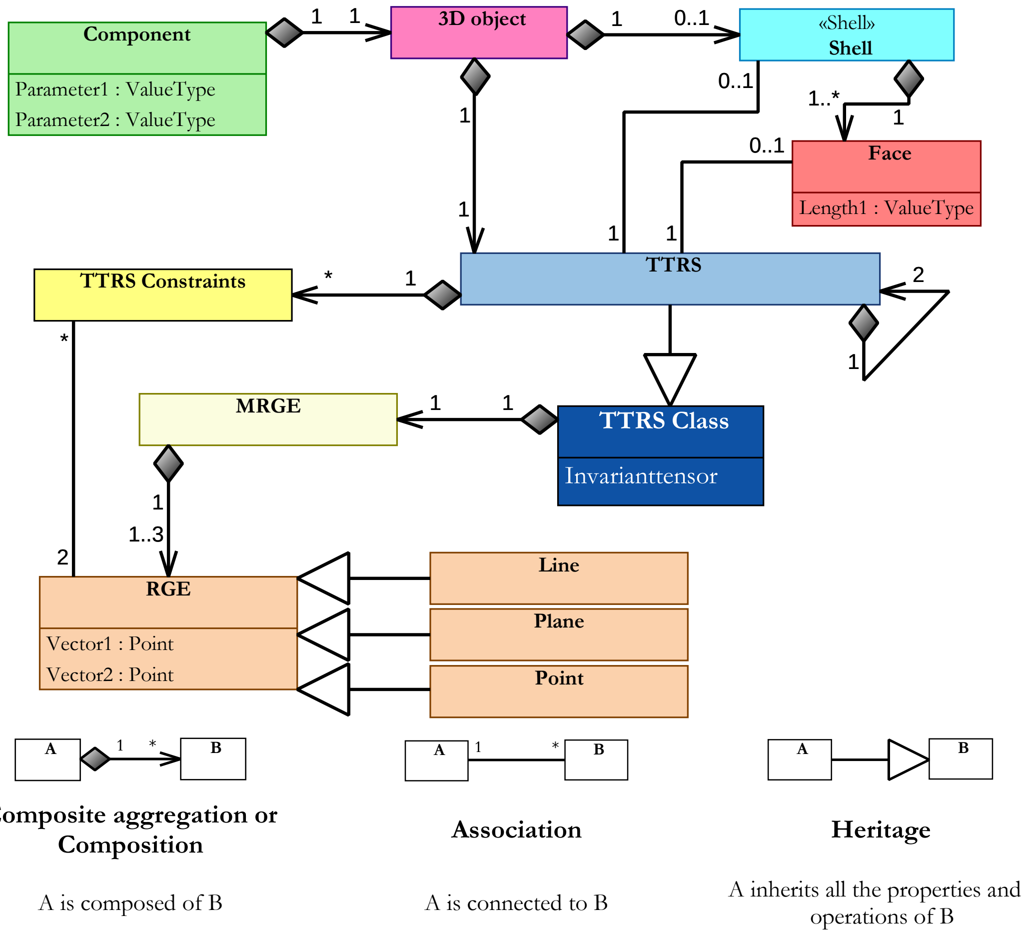


Figure 40: GERTRUDe metamodel used to support geometrical information in SysML.

Although TTRS modeling may appear difficult for a system architect, we propose a library of geometrical objects to facilitate its usage. Then, when the system architects want to add a geometry to a component, they can select the geometry in a library and simply fill in the intrinsic and constraints parameters (e.g. radius and length for a finite cylinder).

B. Example of a description of a finite cylinder geometrical

To explain our data model, we describe the example of a component part with a cylindrical geometry. This example will be represented in a stereotype assembly diagram. This custom diagram, created from the UML "class diagram" allows viewing the relations between different TTRS elements. The component includes a finite cylinder 3D object, composed of one "resulting Finite cylinder TTRS". This element is composed in turn of 3 TTRS: the (infinite) Cylinder TTRS and two Plane TTRSs, as described in Figure 41. Let us point out that the finite cylinder built is itself a revolute TTRS. Each TTRS contains its corresponding MRGE (e.g. line for the (infinite) Cylinder TTRS). The constraints defined between the

MRGE of two TTRSs are linked to the parent geometrical object (e.g. the C6 perpendicularity constraint between a plane and a line belongs to the finite cylinder block). Dimensional parameters are declared as Block property values in the cylinder TTRS for the radius, independently from any TTRS constraint, and in the Finite Cylinder for length L, which results from the C8 TTRS constraint. The 3D parameters correspond to the parameters requested for the representation of the volume objects in the 3D Euclidean space: coordinates (named X, Y and Z) for point RGE or for vectors (direction vector for the line RGE or normal vector for the plane RGE).

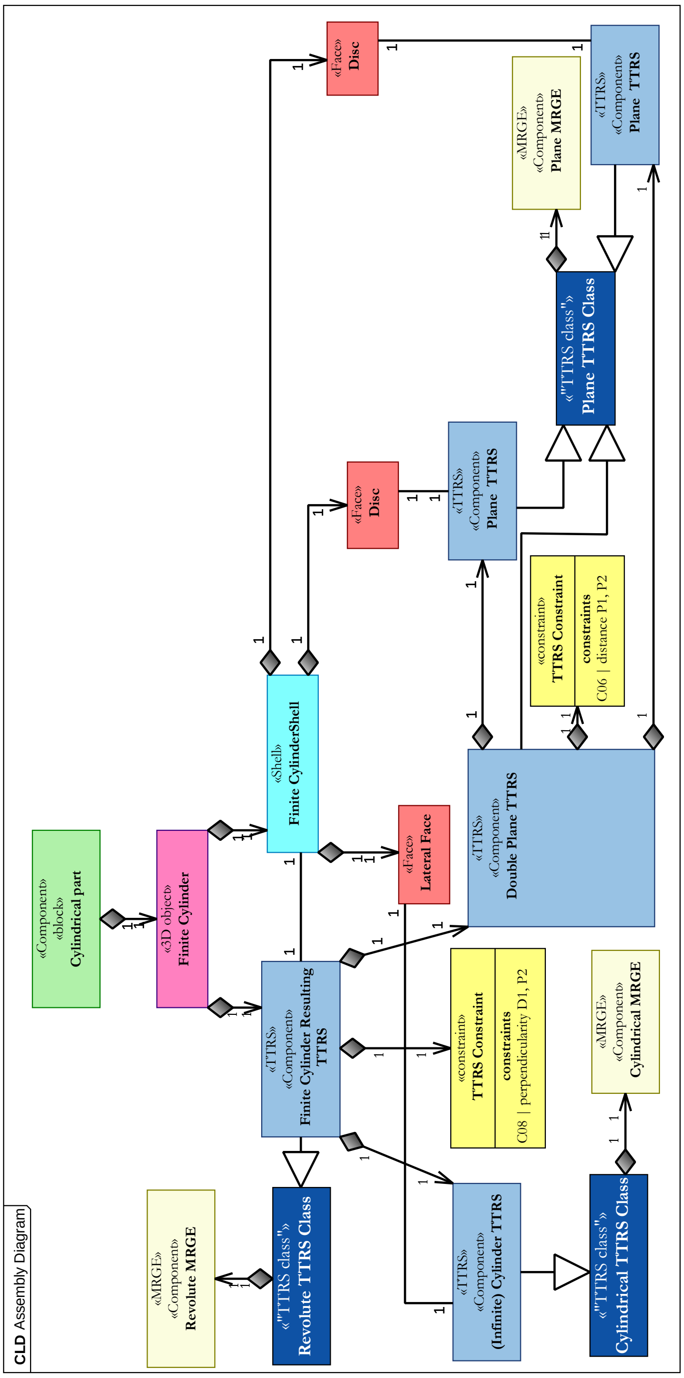


Figure 41: Example of a cylindrical component modeling with GERTRUDe

The positioning constraints between the TTRS of the "resulting Finite Cylinder TTRS" are detailed in an STRD composite structure diagram (Figure 42). Some constraints have been added: the C03 and C11 constraints position and orientate the Revolute Class MRGE.

The “Length equality” constraint block ensures equality between the length parameter of the cylinder and the distance between the two planes. Finally, the “Radius equality” constraint ensures equality between the Radius parameter of the finite cylinder, and the Radius parameter of the Infinite Cylinder.

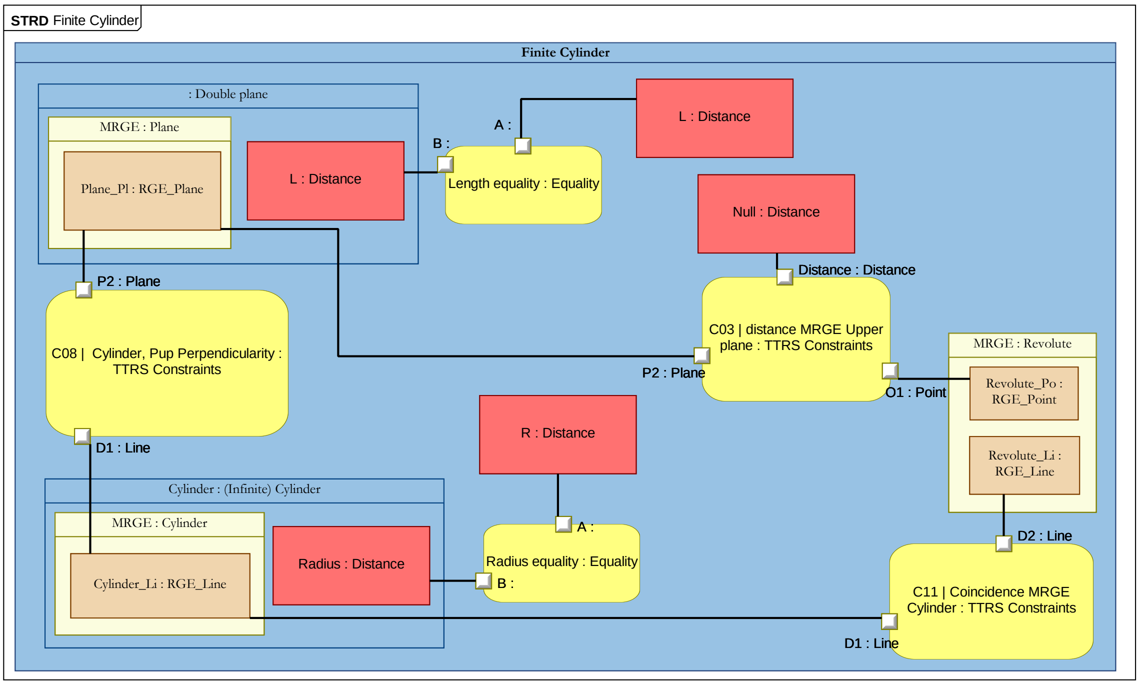


Figure 42: TTRS modeling of a Finite cylinder with GERTRUDe

All models created with GERTRUDe can be exchanged with the 3D CAD tool. This process is performed through model transformation.

V. Geometrical information model transformation strategy

A. Different M2M transformation strategies

The challenge in this paragraph is to describe how to consistently integrate geometrical and design data from various modeling tools used by different design actors during the concept architecture evaluation process.

Indeed, a model transformation procedure has to be developed to improve design data consistency between a system model and a 3D simulated model during the conceptual design,. As a Model-to-Model (M2M) transformation involving a system model (in UML or SysML languages) usually requires building a specific profile beforehand, we developed GERTRUDe, which adds a 3D geometrical structure to a SysML model.

This model transformation has to ensure that the creation of a new geometry in SysML automatically generates the same action in a 3D CAD model, and vice versa, without any additional development. The final 3D architecture is traced back in the system model (in SysML) with the corresponding physical simulation results.

Model transformation operators often work at the M2 metamodel level, by manipulating instances of metamodel constructs (e.g. GERTRUDe and 3D CAD tool metamodels, or between 3D CAD metamodels). A metamodel (M2) is a model that defines the language which expresses a model (M1) of the real world (M0). A metamodel (M2) conforms to a language whose abstract syntax is represented by a reflexive (that conforms to itself) metametamodel (M3), whose OMG standard is the Meta Object Facility or MOF. Two strategies can be considered: direct translation or neutral format. The advantages and drawbacks of each strategy was described by Fowler et al. (Fowler, 1995) and is detailed in Table 16.

Table 16: Pros and cons of using direct translations and neutral formats usage (Fowler, 1995)

Direct Translators	**Neutral Formats**
Software designed for specific translation need	Combine two "half-links" from potentially different suppliers to achieve translation
Includes necessary conversions of data as well as translation	May require "flavoring" of data to achieve best results
Expensive to maintain: they have to be updated every time one system changes	Published formats are stable
Require $n\times(n-1)$ translators to communicate between *n* systems	Require $2 \times n$ half-links to communicate between *n* systems

These two strategies will be studied in the following paragraphs, in view to selecting that which will be used for the M2M transformation of the Thermal 3D Sketcher implementation.

B. Analysis of the "Neutral format usage" strategy applied to SAMOS

In order to test the feasibility of such an approach, this strategy applied to the SAMOS framework has been implemented through an activity diagram, as presented in Figure 43. Dotted arrows indicate control flows, i.e. the temporal sequence of the activities, while complete arrows are object flows that present physical (data) flows. The prefix *Pl_* indicates a flow belonging to the Transformation Platform.

In this approach, the System Architects first create the system model through the "Modeling System" activity in the SysML modeling process. Then, when they have several geometrical specifications, they enrich the model and especially the different architectures, by integrating the geometry through GERTRUDe. The geometrical specifications of the enriched SysML model can then be exported to an XMI standard format file. A specifically developed tool will import and transform this XMI file into a STEP format file, readable by a 3D CAD tool, after having solved, if necessary, certain geometrical constraints through the creation of a geometrical Modelica (.mo format) file. The digital processing of these geometrical constraints will be addressed by a Modelica language solver using a library previously developed by Plateaux et al. (Plateaux, et al., 2009). Then, the 3D architects or other 3D designers will import this STEP file into a 3D CAD tool, to automatically generate the component geometries specified by the System Architects. The 3D architects can then improve this specified architecture, by adding other constraints on the defined components, in order to propose a complete 3D candidate architecture. For example, they can add/modify certain geometric parameters of certain components (shape or parameters such as length, radius, and others), and add relative positioning constraints between the components in order to place them in 3D relative to each other. Once the 3D architecture has been finalized, this 3D CAD file will be transformed and exported into a STEP file, after which, if necessary, certain geometrical constraints will be solved using a geometrical Modelica (.mo format) file, before being transformed again and exported into an XMI file that will be imported into the SysML model. This will ensure the traceability (expected by the System Architects) of the 3D spatial architecture validated by the 3D architects, thanks to an update of the GERTRUDe-enriched SysML model.

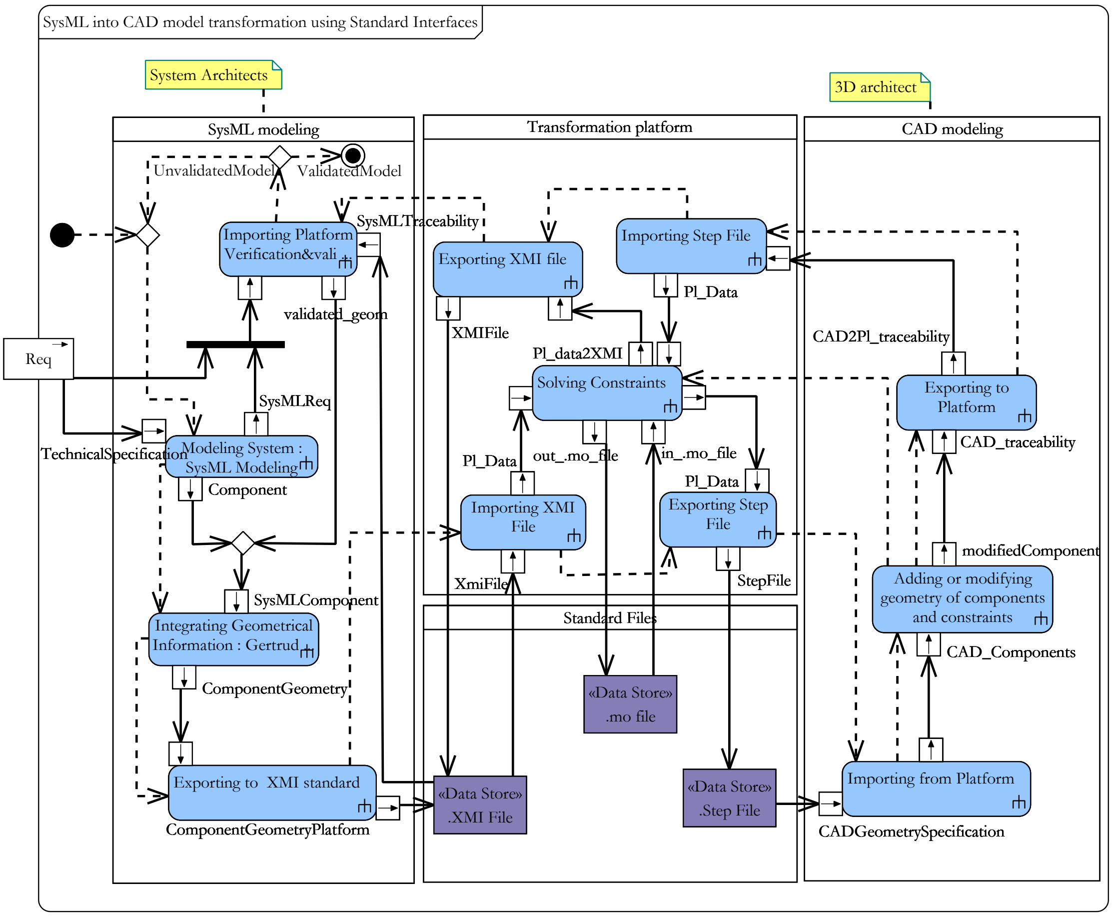


Figure 43: Model transformation formalization using neutral formats (Standards) files.

However, whereas this approach includes standard (neutral) format transformation, it does not guarantee comprehensive geometrical interoperability between all the SysML and 3D tools. Although STEP and XMI are standard formats, software providers do not usually implement them in the same way, so that only a limited number of commercial tools can exchange STEP or XMI format files with each other. In reality, "data flavoring" is required to make this exchange possible. Software editors prefer to develop their own file formats, in order to force customers to buy the same software if they need to exchange their models (notably with their subcontractors), rather than closely adhere to a universal tool interoperability.

C. Analysis of the "Direct translation" strategy applied to SAMOS

The second approach corresponding to the development of a transformation platform that will directly transform a geometrical enriched-SysML model into a 3D CAD tool model, and reciprocally, is described in Figure 44.

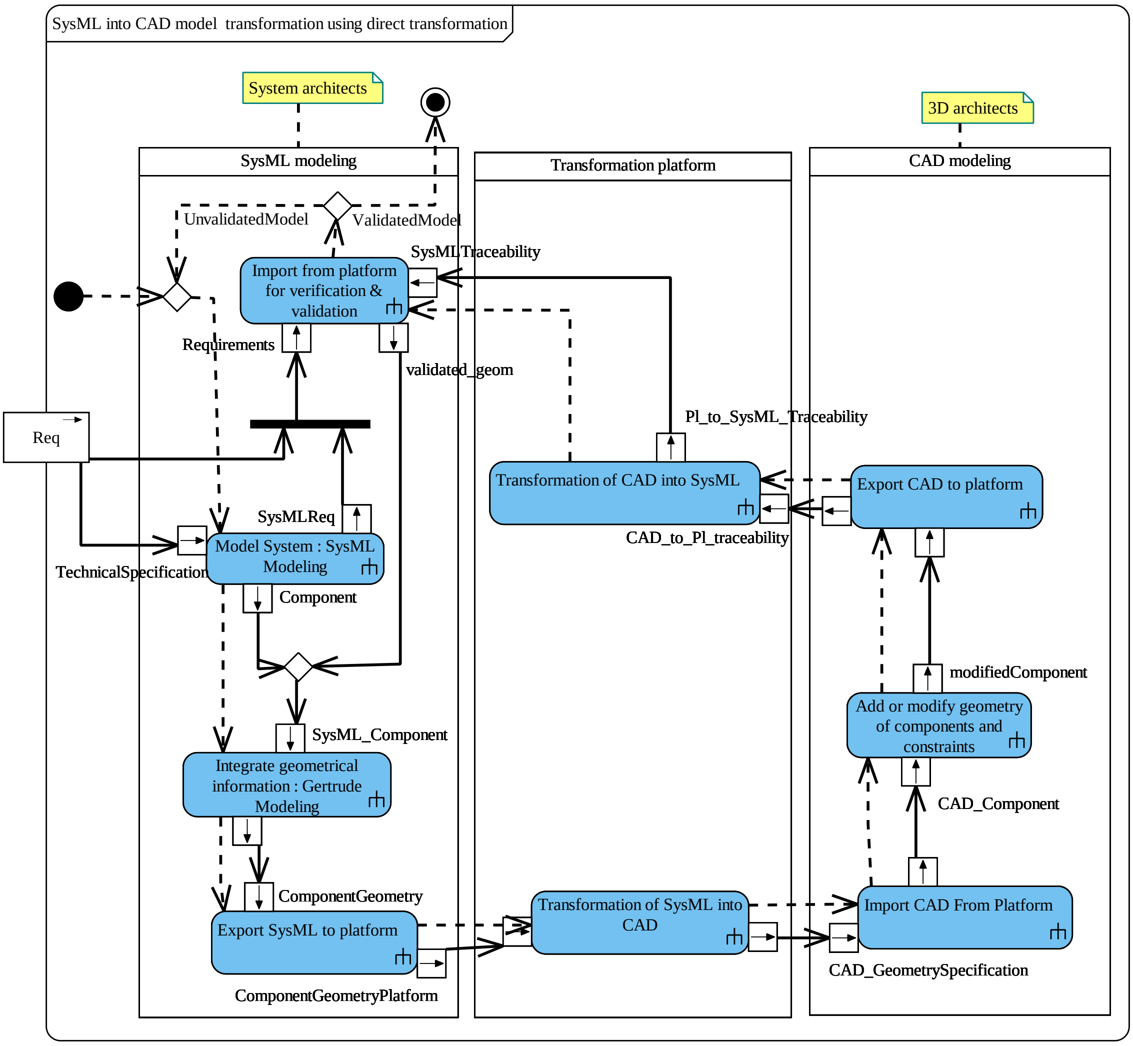


Figure 44: Model transformation formalization using direct translation.

This transformation model presents only two main transformation activities, ensuring the specification and the traceability between a SysML model and a 3D model, respectively.

D. Selection of the model transformation strategy for SAMOS

Two strategies of M2M transformation have been studied: the first approach "Neutral formats usage", which ensures the connection between different software applications using standard files, and the "Direct translation" which directly links the metamodel of the different software applications.

Although the first approach to developing a theoretical two-way parser XMI-STEP seemed more advisable in theory, the fact that it does not guarantee comprehensive geometry interoperability between all SysML and 3D commercial tools makes it less attractive. Indeed, although (Object Management Group, 2016) the works of the Model Interchange Working GROUP (MIWG) have contributed to substantial progress in improving the interoperability of MOF/XMI-based tools, recent publications (Cutting & Noppen, 2015) (Selim, et al., 2015) have underlined that, in spite of the increasing introduction of the XMI standard in

UML tools compared to what existed five years ago (Eichelberger, et al., 2009), these tools do not use the same metamodel to implement UML, thereby limiting the compatibility of their models. Besides, a benchmark established by the AFNeT association (AFNet, 2015) revealed that despite the effort made to manage interoperability between 3D CAD tools, through the STEP file format, interoperability is not yet perfect. Indeed, assembly representation is sometimes problematic even if the geometry is well formatted. This aspect is an important requisite functionality for us. Finally, as not all software providers implement these standards in a fully compatible way, in reality "flavoring of data" (Fowler, 1995) is required to facilitate this data exchange. All the more so as some software vendors prefer to develop their own file formats, in order to encourage customers to buy the same software if they need to exchange models (notably with their subcontractors), rather than closely adhere to universal tool interoperability. Thus, more tools allow model import than allow model export. This is because when supporting the import function, tool vendors can widen the scope of their tool by replacing the imported model generation tool (Kern, 2014).

Thus, due to the additional difficulty of implementation and the considerable development effort required, compared to its expected use, combined with the "3-year deadline" related to the demonstrator's implementation, the direct translation strategy has been chosen to demonstrate the feasibility of the SAMOS framework.

VI. Conclusions

This chapter dealt with the requirements analysis and the resulting choices of approaches related to geometrical modeling for the SAMOS framework. Our choice of geometrical modeling integrates both face modeling (for the 3D representation and to support the thermal analysis) and the TTRS theory (based on infinite surfaces) to support the topology and the relative positioning in particular. The corresponding metamodel was then implemented in the SysML environment, thanks to the development of the GERTRUDe SysML extension. Finally, after comparing the direct translation and the neutral format model transformation strategies, the direct translation was chosen, notably due to its ease of implementation in the SAMOS framework.

As the main objective of the SAMOS framework is to assess 3D architecture based on thermal analysis, including the management of thermal requirements, the next chapter will focus on the integration of thermal modeling in the SAMOS framework.

Chapter 4– Thermal modeling

I. Expression of needs 78

II. Preliminary thermal hypotheses and reminders............. 79

III. Description of the main existing thermal modeling approaches 81

A. Analytic calculation...81

B. Thermal resistance modeling 82

C. Finite element analysis /Finite volume method analysis (FEA/FVM) 84

IV. Thermal modeling selection for SAMOS 86

A. Selection of the thermal modeling approach............................86

B. Description of the temperature calculation with the analytic method selected for thermal modeling...86

1. Conduction modeling...87

2. Convection modeling 90

3. Radiation modeling..96

V. Thermal modeling for the SAMOS framework 98

A. Component, Medium, Interacting Face and thermal interaction network concepts...98

1. Component 98

2. Medium 98

3. Interacting Face...98

4. Thermal interaction network...99

B. Thermal modeling according to geometrical views................100

1. Conduction heat transfers 100

2. Convection heat transfers 102

3. Radiation heat transfers.. 103

C. Example of complete thermal modeling with a multiple component architecture 104

VI. Use of the TTRS theory for the analytical thermal modeling 109

A. Conduction modeling..109
1. Simplification of analytical thermal laws solving.................................... 109
2. Thermal volume conduction analysis in complex geometries 111
3. Contact conduction with partial contact between the faces of two solids112
B. Convection modeling 114
C. Radiation modeling.. 118
VII. TheReSE SysML extension 123
VIII. Conclusion 125

Now that the geometrical modeling theory for the SAMOS framework has been selected in the previous chapter (i.e. shell/face modeling completed by the TTRS theory), and implemented using the GERTRUDe (Geometrical Extension Related to TTRS Reference for a Unified Design) SysML extension, we also need to choose and describe the thermal modeling approach that will be used to assess the 3D physical architecture according to the predefined thermal requirements. In order to make this choice, the expression of needs will be performed through a SysML requirement diagram. Then, hypotheses and reminders of thermal analysis will be provided, before describing the main thermal modeling alternatives: analytic analysis, thermal resistance analysis, and the finite elements analysis/volumes method. Each thermal model will be assessed with the criteria defined from the previous requirements. A description of the thermal model chosen will be given, and its application and adaptation to support the SAMOS framework. Developments related to the use of the TTRS theory for thermal modeling will be presented. Finally, the metamodel of the Thermal Related SysML Extension (TheReSE) developed will be described.

I. Expression of needs

As the thermal modeling procedure is based on 3D geometrical information, whether for the shapes of the component or for their relative positions, the thermal modeling related requirements will also be based on geometrical considerations. The specific thermal modeling related requirements are highlighted by a red contour in Figure 45, in order to study their fulfillment by existing thermal modeling approaches (presented in section III) in section IV, and finally to select the best adapted thermal model procedure for the SAMOS approach.

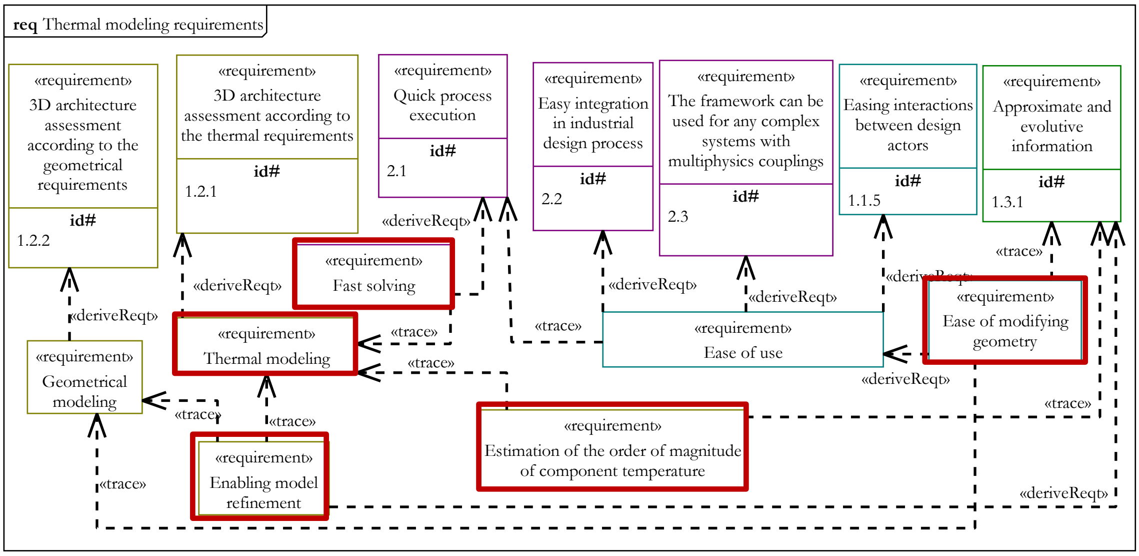


Figure 45: Thermal modeling related requirements.

New additional requirements have been added and are described in Table 17.

Table 17: Description of additional thermal requirements

#Id	Name	Description
1.1.13	Ease of modifying geometry	As we are in the conceptual design phase, the component geometry is not well defined and must be easily modifiable
1.2.12	Enabling model refinement	However, it should be possible to refine the thermal model based on multi-level geometrical elements, as the conceptual design progresses.
2.4	Fast solving	As the phase addressed is the conceptual design, many of ideas are given within a short time, and the results of the assessment must be calculated "instantly" to check whether the architecture evaluated meets the requirements.
1.2.6	Estimation of the order of magnitude of components temperature	As we address the conceptual design, the thermal assessment mainly deals with the estimation of component temperatures in relation to the thermal requirements.

II. Preliminary thermal hypotheses and reminders

Regarding the physical modeling of thermal behavior, three heat transfer modes (and their corresponding couplings) have been considered: conduction, convection and radiation modes.

The following assumptions have been considered:

- Continuum theory (Taine & Petit, 2003) applies for all the physical components studied and their thermal environment. This means that they are all considered mesoscopic and continuum media, with the assumption that each molecular level phenomenon is averaged.
- The components and their surrounding material exhibit homogenous physical properties;
- The local thermodynamic equilibrium is reached at any time.
- The plasma thermodynamics state is not considered in this study.

The following thermal variables have been considered (Taine & Petit, 2003):

- The **heat flux vector** $\overrightarrow{J_{th}}$ (W/m²), which is the flow of heat per unit area normal to the direction of flow.
- – The **heat rate/heat flow** ϕ (W) is calculated by multiplying the heat flux by the total cross-sectional area through which the heat flows for a 1D problem or by integrating it over the area of flow for a multidimensional problem.

$$\phi = \oiint_S \overrightarrow{J_{th}}.\overrightarrow{dS} \quad (1)$$

- – where $\overrightarrow{dS} = \overrightarrow{n_S}.dS$, with $\overrightarrow{n_S}$ is the surface outgoing normal unit vector.
- Temperature T is defined as the variable that quantifies particle agitation in a gas (when there is no particle agitation, T is at its lowest limit, defined as $0\ K$ $(-273.15\ °C)$).

Throughout the rest of this manuscript, we will consider the thermal laws described in Table 18.

Table 18: Main thermal laws considered.

Heat transfer modes	**Law name**	**Equation**	
Conduction	Fourier	$\overrightarrow{Jth} = -\lambda.\overrightarrow{grad}(T)$	(2)
Convection	Newton	$\overrightarrow{Jth} = h\left(T_s - T_f\right)\overrightarrow{n_S}$	(3)
Radiation*	Stephan Boltzmann	$\overrightarrow{Jth} = \epsilon.\sigma.T^4.\overrightarrow{n_S}$	(4)
	Exchanged flows	$\phi_{er} = F_{e\to r}.\phi_e - F_{r\to e}.\phi_r$	(5)

* We will consider that the whole incident heat is absorbed by the components exposed.

Where:

Symbol used (French standard)	**US standard symbol**	**Name**	**Unit (S.I.)**
λ	k	thermal conductivity	$W.m^{-1}.K^{-1}$
$\overrightarrow{Jth}$	$\vec{q}$	heat flux vector	$W.m^{-2}$
T		temperature	K
h		heat transfer coefficient	$W.m^{-2}.K$
T_s		temperature of the solid	K
T_f		temperature of the fluid	K
$\overrightarrow{n_S}$		unitary vector normal to the surface	Dimensionless
ϵ		emissivity	Dimensionless
σ		Stephan-Boltzmann constant	$W.m^{-2}.K^{-4}$
ϕ_{er}	Q_{er}	exchanged rate of heat flow	W
ϕ_e	Q_e	emitted rate of heat flow	W
ϕ_r	Q_r	received rate of heat flow	W
$F_{e\to r}$		view factor from the emitting to the receiving component	Dimensionless
$F_{r\to e}$		view factor from the receiving to the emitting component	Dimensionless

When considering a complex system, the heat exchanged between the different components often results from a coupling between different heat transfer modes. Thus, these equations will be solved to calculate the resulting temperature of the components and to verify that it meets the corresponding component requirements.

Existing research and current industrial practices mainly propose three methods to solve these heat transfer equations: analytic calculation, the thermal resistance approach, and the finite element analysis/volume method.

III. Description of the main existing thermal modeling approaches

A. Analytic calculation

The analytic calculation consists in manipulating a set of equations, including the boundary conditions, in order to solve them. Analytic calculation has the advantage of providing exact results.

Concerning conduction inside a solid, the temperature solution is projected on the eigenvectors that depend on the component geometry and the boundary conditions (cf. §IVB1). Using this method Haji-Sheikh et al. proposed to manage conduction through the contact between different rectangular parallelepiped components (Haji-Sheikh, et al., 2003). A year later, De Monte improved this method by enabling the automatic calculation of the eigenvalues α_n required to determine the temperature (cf. Table 21) for a rectangular parallelepiped component (De Monte, 2004). Finally, Hahn and Ozisik enhanced this method by considering different simple geometries, such as cylinder, sphere, half-sphere, and the contact conduction management between these different geometries (Hahn & Özişik, 2012).

Concerning the analysis of convection between a fluid and a solid, research has mainly focused on the calculation of the heat transfer coefficient. Although this coefficient is often approximated as a constant, it in fact depends on various parameters, such as the velocity and the material of the fluid, and the geometry and size of the solid component considered (Taine & Petit, 2003) .The main method for calculating this coefficient is the **Nusselt correlation** (appendix 3). This correlation is calculated empirically for each component geometry. For example, Licht proposed a correlation for a square tube containing a high pressure fluid (Licht, et al., 2008). Although the main Nusselt correlations are available for simple geometries, such as plane, cylinder and sphere (Bergman & Incropera, 2011), the main issue concerning the convection analysis lies in the calculation of the fluid velocity. Indeed, this calculation is based on the Navier-Stokes equation (Temam, 1984). This equation has not yet been fully resolved: it is one of the seven millennium prize problems[1]. Nevertheless, some studies have proposed solving this equation under specific conditions (Turkyilmazoglu, 2013) (Guerrero-Martinez, et al., 2017). A classical simplification is to consider the fluid as incompressible and to use Bernoulli's principle (Taine & Petit, 2003) along a streamline.

Radiation modeling is usually based on the calculation of view factors between the emitting and the receiving components. These view factors depend on the geometry and orientation of the components considered. They are usually calculated for simple geometries. For example, Krishnaprakas proposed a calculation of the view factor between two inclined rectangle surfaces (Krishnaprakas, 1997). This makes it possible to calculate the view factor between two rectangular parallelepipeds, as they consist of a set of rectangles. Aliasghar and

---

[1] http://www.claymath.org/millennium-problems

Felske calculated the view factor between two cylinders (Ameri & Felske, 1982). Finally, Felske also calculated the view factor between two spheres (Felske, 1978).

Analytic calculation is well-adapted for the exact calculations of heat transfers for simple geometries. However, equation solving is not easy to do with complex geometries, since a mere chamfer in a part can significantly change the heat exchange. This was observed when we studied the convection phenomenon in the case study of an Airbus helicopter bay (chapter 6): a small variation of geometry can greatly influence air movement and the temperature of the components. The same problem was observed with the radiation propagation mode (Howell, 2014).

B. *Thermal resistance modeling*

Although analytic calculations are adapted for simple geometries, the calculation can be hard to resolve. For example, the solution of the conduction heat transfer mode for a rectangular parallelepiped is composed of two infinite sums Appendix 3. An approximation by considering the ten first terms can be sufficient, but the calculation may still take a long time. For this reason, a simplification of these equations has been developed by using an electrical analogy: thermal resistance modeling (Taine & Petit, 2003) (Figure 46). The rate of heat flux ϕ acts as the current intensity i and the temperature T acts as the electric potential V. Ohm's law therefore expresses that the electric resistance can be calculated using the following formula:

$$R_{el} = \frac{V_B - V_A}{i} \tag{6}$$

where V_A and V_B are the potential pins of the resistance and i is the intensity flowing through the resistance.

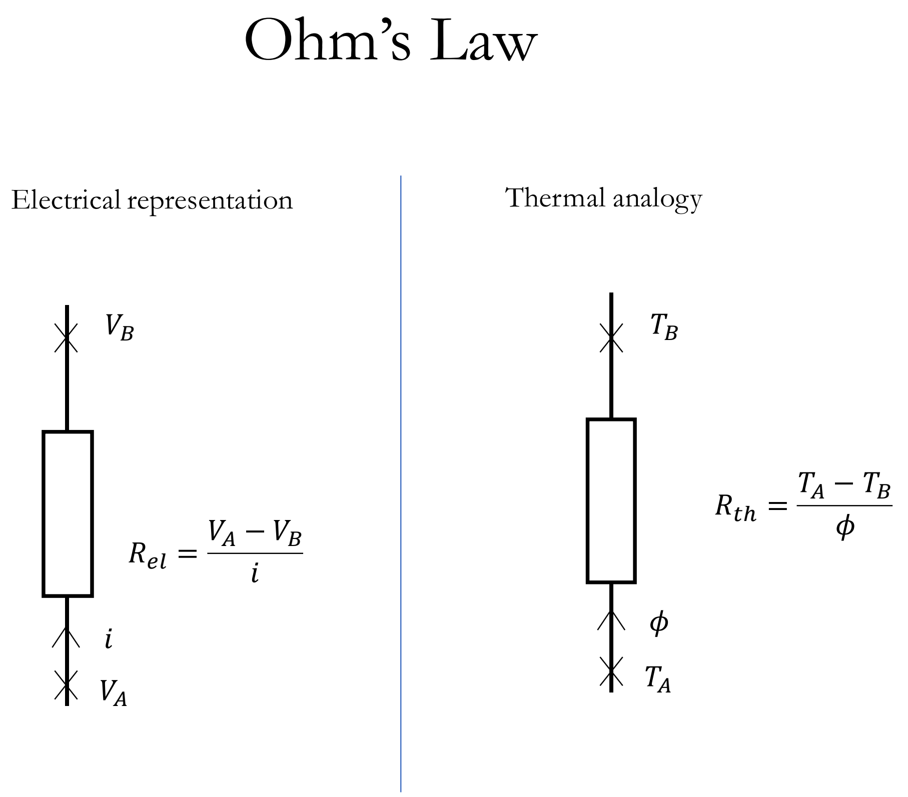


Figure 46: Analogy of the thermal resistance regarding the electrical resistance.

The analogy thus allows calculating the thermal resistance (6) using the temperature difference between A and B $(T_A - T_B)$ and heat flow exchanged ϕ_{th}.

$$R_{th} = \frac{T_A - T_B}{\phi_{th}} \quad (7)$$

The calculation of this thermal resistance in the cases presented in Table 19.

Table 19: Main thermal resistance formula

Heat transfer mode		Thermal resistance
Conduction	Constant section	$R_{Coth} = \frac{e}{\lambda . S}$ (8)
	Cylinder section	$R_{Cyth} = \frac{\ln\left(\frac{R_e}{R_i}\right)}{2.\pi.\lambda.L}$ (9)
	Spherical section	$R_{Cpth} = \frac{1}{4.\pi.\lambda}\left(\frac{1}{R_e} - \frac{1}{R_1}\right)$ (10)
Convection		$R_{Cvth} = \frac{1}{h.S}$ (11)
Radiation		$R_{Rth} = \frac{1}{F_{e\to r}.S_R.\sigma.\epsilon.\left(T_{le}{}^2 + T_{lr}{}^2\right).(T_{le} + T_{lr})}$ (12)

Where:

Symbol used (French standard)	US standard symbol	Name	Unit (S.I.)
e		height of extruded section	m
S	A	constant surface	m^2
R_e		external radius	m
R_i		internal radius	m
L		length of the cylinder	m
S_R	A_R	receiving apparent surface	m^2
T_{le}		approximated temperature of the emitting component	K
T_{lr}		approximated temperature of the receiving component	K

The radiation resistance is particular, as it depends on the (approximate) temperature of the components. Then the radiation resistance is calculated by linearization of Stephan Boltzmann's law and the nearest radiation equivalent resistance is given by iterations: the calculation begins with T_{le} and T_{lr} as the initial temperatures of the components and then the corresponding calculated temperatures replace T_{le} and T_{lr} for the next iteration to get a finer result.

C. *Finite element analysis / Finite volume method analysis* (FEA/FVM)

The finite element analysis (FEA) (Krysl, 2006) is a numerical approach adapted to solve physical partial differential equations. This method can address convection and radiation problems with any kind of geometry.

The finite elements method is performed in three steps:

1. First, the shape is discretized in meshes. These meshes can have two kinds of geometries: tetrahedron or hexahedron. A mesh is composed of different nodes. Different variables, which correspond to the degree of freedom, can be associated to each node. In the thermal analysis case, there are four degrees of freedom per node: one for the temperature T, and three for the coordinates of the heat flux vector (Figure 47).

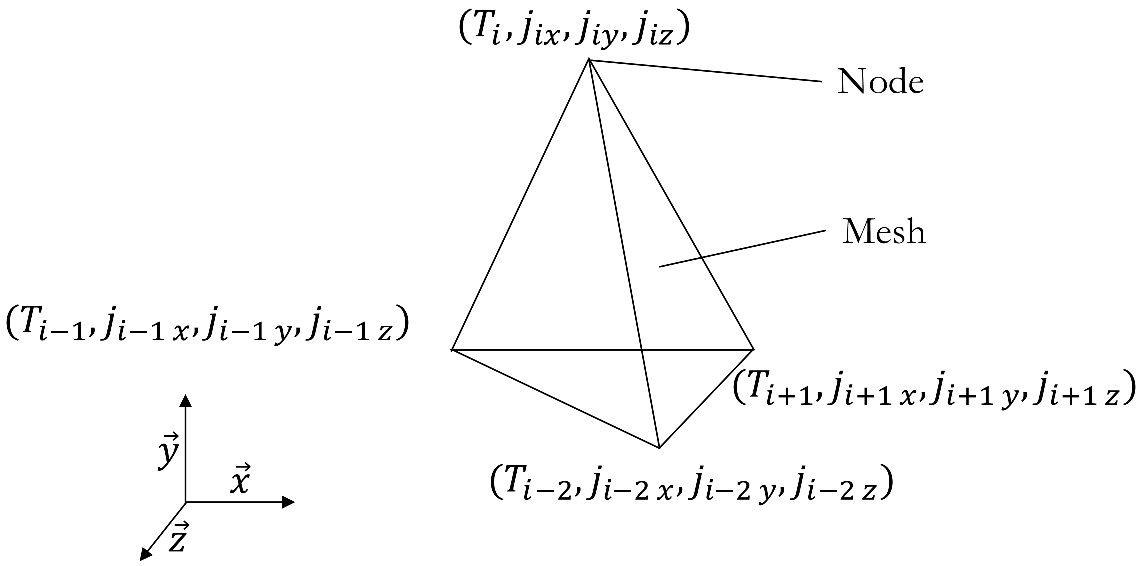


Figure 47 Thermal modeling based on a solid geometrical discretization.

2. Secondly, physical equations are established and then linearized.
3. Finally, the boundary conditions are applied locally for the external nodes.

The finite element analysis is a matrix calculation, where each degree of freedom is considered as a vector component and the matrix that links the boundary conditions to this vector is given by the second step of the finite element analysis.

Whereas the finite element analysis is adapted to the solid elements, the finite volumes method is more adapted for fluid analysis, notably to calculate the fluid displacement.

The FEA and FVM (Krysl, 2006) allow calculating the discretized temperatures and heat flows of any geometrical element. However, as this calculation is based on discretized geometries, the quality of the results strongly depends on the mesh size and may therefore generate long calculation solving times. Current research related to finite element analysis focused on reducing calculation time through model order reduction approaches (like the super-elements condensation method (Botto, et al., 2002)).

IV. Thermal modeling selection for SAMOS

A. Selection of the thermal modeling approach

The three thermal modeling approaches presented in the previous paragraph allow modeling thermal behavior and providing the temperature of different geometrical elements. The requirements related to the choice of the thermal modeling approach, outlined in red in Figure 45, will be used as criteria to evaluate these different thermal modeling approaches according to their suitability for SAMOS. This evaluation is given in Table 20.

Table 20: Evaluation of the different thermal modeling approaches according to the requirements for the SAMOS implementation.

Requirements \ Thermal modeling	Analytic calculation	Thermal resistance modeling	Finite elements/volumes analysis (FEA/FVM)
Enabling model refinement	✓	✗	✓
Fast solving	✓	✓	✗
Ease of modifying geometry	✓	✓	✗
Estimation of the order of magnitude of component temperatures	✓	✓	✓

- The "Enabling model refinement" requirement is not satisfied by thermal resistance modeling, because this approach cannot be refined in a more complex model. However, it is satisfied by both analytic calculation and FEA/FVM. Indeed, the analytic calculation model can be refined by modifying the law considered (1D, 2D, 3D), and increasing the number of terms considered in the sum and then the resulting precision of the calculated temperature. In the same way, FEA/FVM can be refined by increasing the number of meshes in the same component considered.
- The "Fast solving" requirement is not fulfilled by the FEA/FVM, since these simulations can be very long, depending on the number of meshes.
- "Ease of modifying geometry" is also crossed out in the FEA/FVM box, since a modification of the geometry implies the complete modification of the mesh and requires reinitialization.
- The "Estimation of the order of magnitude of component temperatures" can be done by the three models.

The analytic calculation and thermal resistance modeling seem to be the best-adapted according to the previous requirements. As the analytic calculation meets all requirements including model refinement, we describe this approach in the next paragraph.

B. Description of the temperature calculation with the analytic method selected for thermal modeling

The analytic temperature calculation must be defined for each heat transfer mode.

1. Conduction modeling

The conduction modeling approach is detailed here for the 2D analysis. The principle is the same as for the 3D analysis detailed in appendix 3 for the following geometries: cylinder, sphere, and rectangular parallelepiped. Concerning a 2D element, and taking into account the hypotheses set out in §II, the elementary modeling representation (for an infinitely small element) is provided in Figure 48.

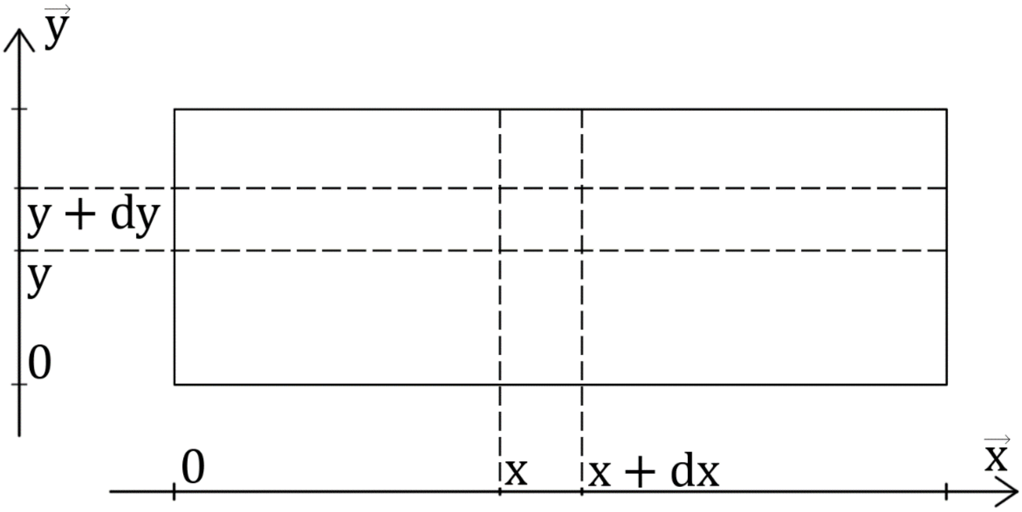


Figure 48: 2D elementary representation for conduction modeling.

As the considered system is a 2D system, we can write: $T(x, y, z) = T(x, y)$ and then:

$$\frac{\partial^2 T}{\partial z^2} = 0 \tag{13}$$

In a Cartesian coordinate system, the 3D Laplacian equation described in appendix 3 becomes:

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0 \tag{14}$$

The expression of the temperature is described as the product of two unidimensional functions thanks to the separation of the variable hypothesis:

$$T(x, y) = f(x).g(y) \tag{15}$$

Then the Laplacian equation in 2D becomes:

$$g.\frac{d^2 f}{dx^2} + f.\frac{d^2 g}{dy^2} = 0 \tag{16}$$

$$\Rightarrow \frac{1}{f}.\frac{d^2 f}{dx^2} = -\frac{1}{g}.\frac{d^2 g}{dy^2} \tag{17}$$

Thus, $\left(\frac{1}{f}.\frac{d^2 f}{dx^2}\right)$ is independent of $\left(-\frac{1}{g}.\frac{d^2 g}{dy^2}\right)$, as $\left(\frac{1}{f}.\frac{d^2 f}{dx^2}\right)$ depends on the x variable and $\left(\frac{1}{g}.\frac{d^2 g}{dy^2}\right)$ depends on the y variable. Then, when introducing the constant $\alpha \in \mathbb{R}^{+*}$, two independent second order equations must then to be solved:

$$\frac{1}{f}.\frac{d^2f}{dx^2} = -\frac{1}{g}.\frac{d^2g}{dy^2} = -\alpha^2 \tag{18}$$

$$\Rightarrow \begin{cases} \dfrac{d^2f}{dx^2} = -\alpha^2 f \\ \dfrac{d^2g}{dy^2} = \alpha^2 g \end{cases} \tag{19}$$

two independent second order equations must then be solved:

$$\Rightarrow \begin{cases} f(x) = K_1.\cos(\alpha.x) + K_2.\sin(\alpha.x) \\ g(y) = L_1.\cosh(\alpha.y) + L_2.\sinh(\alpha.y) \end{cases} \tag{20}$$

α can be calculated using the boundary conditions. As trigonometric functions are periodic, an infinity of solutions are available We must also rename α as α_n where n belongs to the interval $[\![0; +\infty[\![$; α_n are then called the **eigenvalue**.

We deduce that:

$$f_n(x) = K_1.\cos(\alpha_n.x) + K_2.\sin(\alpha_n.x) \tag{21}$$

The function $T(x, y)$ is then projected on the basis formed by the eigenvectors $\{f_n\}_{n\in\mathbb{N}}$, in order to provide the solution function:

$$\Rightarrow T(x,y) = \sum_{n=0}^{+\infty} f_n(x).g_n(y) \tag{22}$$

Where

$$g_n(y) = L_{1n}.\cosh(\alpha_n.y) + L_{2n}.\sinh(\alpha_n.y) \tag{23}$$

L_1 and L_2 are dependent on n, since the temperature is projected on the eigenvector basis, L_{1n} and L_{2n} are calculated by the integration for a period of each term of the sum and the solution of the equation for the given boundary condition. The Fourier basis allows keeping only one term of the sum.

For each case of boundary conditions, the eigenvalues and eigenfunctions are given by (Maillet, 2000) and reported in Table 21. The boundary conditions cited are given in some general cases, but can be easily extended. For example, for the first case, the boundary condition of $T(0, y) = 0$ can be easily changed by $T(0, y) = T_1$ using a change in the variable: $T_1(x, y) = T(x, y) - T_1$, in the same way, this change of variable can be done for the heat flow.

Table 21: Calculation of the eigenvalues and eigenvectors according to different boundary condition cases.

Case	**Boundary condition in x = 0**	**Boundary condition in x = L**	**Eigenvalue α_n**	**Eigenvector $f_n(x)$**
1	$T(0,y)=0$	$T(L,y)=0$	$\frac{n.\pi}{L}$	$\sin(\alpha_n x)$
2	$T(0,y)=0$	$\frac{\partial T(L,y)}{\partial x}=0$	$\frac{\left(n+\frac{1}{2}\right).\pi}{L}$	$\sin(\alpha_n x)$
3	$T(0,y)=0$	$\lambda\frac{\partial T(L,y)}{\partial x}+h.T(0,y)=0$	$\tan(\alpha_n)=-\frac{\lambda\alpha_n}{h}$	$\sin(\alpha_n x)$
4	$\frac{\partial T(0,y)}{\partial x}=0$	$T(L,y)=0$	$\frac{\left(n+\frac{1}{2}\right).\pi}{L}$	$\cos(\alpha_n x)$
5	$\frac{\partial T(0,y)}{\partial x}=0$	$\frac{\partial T(L,y)}{\partial x}=0$	$\frac{n.\pi}{L}$	$\cos(\alpha_n x)$
6	$\frac{\partial T(0,y)}{\partial x}=0$	$\lambda\frac{\partial T(L,y)}{\partial x}+h.T(0,y)=0$	$cotan(\alpha_n)=\frac{\lambda\alpha_n}{h}$	$\cos(\alpha_n x)$
7	$\lambda\frac{\partial T(0,y)}{\partial x}+h.T(0,y)=0$	$T(L,y)=0$	$\tan(\alpha_n)=-\frac{\lambda\alpha_n}{h}$	$\sin(\alpha_n(L-x))$
8	$\lambda\frac{\partial T(0,y)}{\partial x}+h.T(0,y)=0$	$\frac{\partial T(L,y)}{\partial x}=0$	$cotan(\alpha_n)=\frac{\lambda\alpha_n}{h}$	$\cos(\alpha_n(L-x))$
9	$\lambda\frac{\partial T(L,y)}{\partial x}+h_1.T(0,y)=0$	$\lambda\frac{\partial T(L,y)}{\partial x}+h_2.T(0,y)=0$	$\tan(\alpha_n)=-\frac{\lambda\alpha_n(h_1+h_2)}{(\lambda\alpha_n)^2-h_1.h_2}$	$\cos(\alpha_n x)+\frac{h_1}{\lambda.\alpha_n}\sin(\alpha_n x)$

Concerning the 3D elements, we have used the same process to calculate the temperature T (appendix 3).

Furthermore, this analysis can be performed in different coordinate systems. The bases of eigenfunctions corresponding to various coordinate systems (associated with specific geometry) (Hahn, 2010) are reported in Table 22.

Table 22: Eigenfunctions bases associated to specific coordinate systems

Coordinate system	Bases related to the corresponding eigenfunctions
Rectangular	Exponential, circular, hyperbolic
Circular cylinder	Bessel, exponential, circular
Elliptic cylinder	Mathieu, circular
Parabolic cylinder	Weber, circular
Spherical	Legendre, power, circular
Prolate spheroidal	Legendre, circular
Oblate spheroidal	Legendre, circular
Parabolic	Bessel, circular
Conical	Lamé, power
Ellipsoidal	Lamé
Paraboloidal	Baer

2. Convection modeling

Concerning the temperature calculation based on the convection analysis, two aspects must be modeled: the first addresses the exchange between a solid and a fluid, and the second is related to the fluid movement.

a. Heat transfer between a solid and a fluid.

Convection is the transmission of heat by the physical movement of particles. Convection occurs between two elements when the thermodynamic state of one of the materials considered is liquid or gaseous. Thus it must be calculated since the particle movements are linked to the fluid displacement.

Two models of fluid movement have to be considered:

- Forced convection: where the fluid particles are set in motion by an external element (fan, pump, wind, etc.);
- Natural convection: the physical movement of a fluid caused by increasing its temperature. When certain fluid particles are warmer than other fluid particles, the warmer particles rise and are replaced by cold particles. Finally, heat dissipation occurs through the displacement of particles.

Newton's law (3) is applicable for both the forced and the natural convections. The calculation of the heat transfer coefficient h is then performed using the Nusselt correlation. Although the Nusselt correlation is an experimental correlation, the choices of the required parameters for calculating h are defined theoretically by the Buckingham theorem (Buckingham, 1914). This theorem explains that each physical problem with n variables can be decomposed by $(n-k)$ dimensionless variables. (k is the number of fundamental units). Wami et al. (Wami & Ibrahim, s.d.) use this theorem to demonstrate, for the forced convection, that:

$$\frac{h.L}{\lambda} = f\left(\frac{v.L}{\nu}, \frac{c_p\,\mu}{\lambda}\right) \tag{24}$$

where:

Symbol used (French and US standard)	Name	Unit (S.I.)
$\vec{v}$	velocity of the fluid (v designates the norm of the vector)	$m.s^{-1}$
ν	kinematic viscosity	$m^2.s^{-1}$
c_p	specific heat	$J.kg^{-1}.K$
μ	dynamic viscosity of the fluid	$N.s.m^{-2}$
ρ	density of the fluid	$kg.m^{-3}$

Using this theorem with the previous notations, four different dimensionless numbers have been considered and are detailed in Table 23.

Table 23: Definition of the dimensionless numbers used for the Nusselt correlation.

<table>
<tr><th>Name</th><th>Physical meaning</th><th>Formula</th></tr>
<tr><td>Reynolds number</td><td>The Reynolds number is the ratio of the inertial forces to the viscous forces. This number is used in order to predict the regimes of the fluid considered.
• $Re < 2000$: laminar flow
• $2000 < Re < 4000$: transitional flow
• $Re > 4000$: turbulent flow
This number is considered only for forced convection.</td><td>$Re = \frac{v.L}{\nu}$</td></tr>
<tr><td>Nusselt number</td><td>The Nusselt number is the ratio of the total heat transfer to the conductive heat transfer. This number is used to give the order of magnitude of the heat transferred by convection.</td><td>$Nu = \frac{h.L}{\lambda}$</td></tr>
<tr><td>Prandtl number</td><td>The Prandtl number is the ratio of the viscous diffusion rate to the thermal diffusion rate in the fluid. The Prandtl number assesses the speed of the thermal phenomena compared to the hydrodynamic phenomena in the fluid.
Thus a high Prandtl number means that the temperature profile will be strongly dependent on the speed of the fluid. Conversely, a low Prandtl number means that the thermal conduction is so predominant that the fluid speed profile has little effect on the temperature profile.</td><td>$Pr = \frac{c_p.\mu}{\lambda}$</td></tr>
</table>

In accordance with the Buckingham theorem, different experimental tests have been performed in order to establish the relations between the Nusselt, Reynold and Prandlt numbers. These correlations are named **Nusselt correlations** and strongly depend on the solid geometry immersed in a fluid, as in Table 24. Nusselt correlation given in this table are given in (Taine & Petit, 2003) (Bergman & Incropera, 2011)

Table 24: Nusselt correlation for forced convection applied on various geometries of solid.

Geometry of the solid immersed in a fluid	**Nusselt correlations**	**Conditions**
Flat plate	$Nu = 0.332. Re^{\frac{1}{2}}. Pr^{\frac{1}{3}}$	- Laminar - $Pr \geq 0.6$
	$Nu = 0.664. Re^{\frac{1}{2}}. Pr^{\frac{1}{4}}$	- Laminar - $Pr \geq 0.6$
	$Nu = 0.564. Re^{\frac{1}{2}}. Pr^{\frac{1}{2}}$	- Laminar - $Pr \leq 0.05$
	$Nu = 0.0296. Re^{\frac{4}{5}}. Pr^{\frac{1}{3}}$	- Turbulent flow - $Re \leq 10^8$ - $0.6 \leq Pr \leq 60$
	$Nu = \left(0.037. Re^{\frac{4}{5}} - 871\right). Pr^{\frac{1}{3}}$	- Turbulent flow - $Re \leq 10^8$ - $0.6 \leq Pr \leq 60$
Cylinder	$Nu = C_1. Re^{m_1}. Pr^{\frac{1}{3}}$	- $0.4 \leq Re \leq 4.10^5$ - $Pr \geq 0.7$
	$Nu = 0.3 + \left[0.62. Re^{\frac{1}{2}}. Pr^{\frac{1}{3}}. \left(1 + \left(\frac{0.4}{Pr}\right)^{\frac{2}{3}}\right)^{-\frac{1}{4}}\right]. \left[1 + \left(\frac{Re}{282000}\right)^{\frac{5}{8}}\right]^{\frac{4}{5}}$	- $Pr \geq 0.2$
Sphere	$Nu = 2 + \left(0.4. Re^{\frac{1}{2}} + 0.06. Re^{\frac{2}{3}}\right). Pr^{0.4}$	- $0.4 \leq Re \leq 4.10^5$ - $0.7 \leq Pr \leq 380$
Falling drop	$Nu = 2 + 0.6. Re^{\frac{1}{2}}. Pr^{\frac{1}{3}}$	No specific condition
Square (face)	$Nu = 0.158. Re^{0.66}. Pr^{n_1}. \left(\frac{Pr}{Pr_S}\right)^{\frac{1}{4}}$	- $5.10^3 \leq Re \leq 6.10^4$ - $0.7 \leq Pr \leq 500$
Square (coin)	$Nu = 0.304. Re^{0.59}. Pr^{n_2}. \left(\frac{Pr}{Pr_S}\right)^{\frac{1}{4}}$	- $6.10^3 \leq Re \leq 6.10^4$ - $0.7 \leq Pr \leq 500$
Hexagon (face)	$Nu = C_2. Re^{m_2}. Pr^{n_2}. \left(\frac{Pr}{Pr_S}\right)^{\frac{1}{4}}$	- $5.2.10^3 \leq Re \leq 1.05.10^5$ - $0.7 \leq Pr \leq 500$

Geometry of the solid immersed in a fluid	Nusselt correlations	Conditions
Hexagon (coin)	$Nu = 0.15.Re^{0.638}.Pr^{n_3}.\left(\frac{Pr}{Pr_S}\right)^{\frac{1}{4}}$	- $4.5.10^3 \leq Re \leq 9.07.10^4$ $0.7 \leq Pr \leq 500$
Thin plate (face)	$Nu = C_4.Re^{m_4}.Pr^{n_4}.\left(\frac{Pr}{Pr_S}\right)^{\frac{1}{4}}$	- $7.10^3 \leq Re \leq 10^4$ $0.7 \leq Pr \leq 500$

They are defined within the validity intervals of the Reynolds and Prandlt numbers. Furthermore, two kinds of Prandlt numbers have been defined depending on the place where the temperature is considered: Pr is the Prandlt number of the fluid considered infinitely far from the interface between the fluid and the solid, whereas Pr_S is the Prandlt number of the fluid at the interface between the fluid and solid, highly dependent on the solid initial temperature. If the difference of these two fluid temperatures is not very significant, it is can be assumed that $Pr = Pr_S$, then ($\frac{Pr}{Pr_S} = 1$).

The table of the Nusselt correlations for natural convection is given in appendix 3.

Finally, these experimental Nusselt correlations allow calculating the heat transfer coefficient h for different fluid parameters. As detailed in Table 19, the thermal resistance for the convection can be then calculated.

b. Fluid movement modeling

The convection analysis does not only consider the heat exchange between a fluid and a solid, it also addresses the heat transfer due to the movement of fluid, governed by the Navier-Stokes equation. As described previously, this equation is not yet solvable today, thus we consider a simpler model for our works.

We consider the following hypothesis:

- the flow is incompressible,
- the flow is non-rotational,
- the fluid is perfect (pressure drop and viscous effect are not considered)

The Bernoulli principle demonstrates that along a streamline (Figure 49), the following formula can be applied:

$$\frac{v^2}{2} + g.z + \frac{p}{\rho} = constant \quad (25)$$

Where;

Symbol used (French and US standard)	Name	Unit (S.I.)
g	gravity acceleration	$m.s^{-2}$
z	altitude of the considered point from a plane reference	m
p	fluid pressure at the point considered	$N.m^{-2}$

This equation requires another equation (i.e. the flow rate conservation) to calculate all the necessary variables.

$$v.S = constant \quad (26)$$

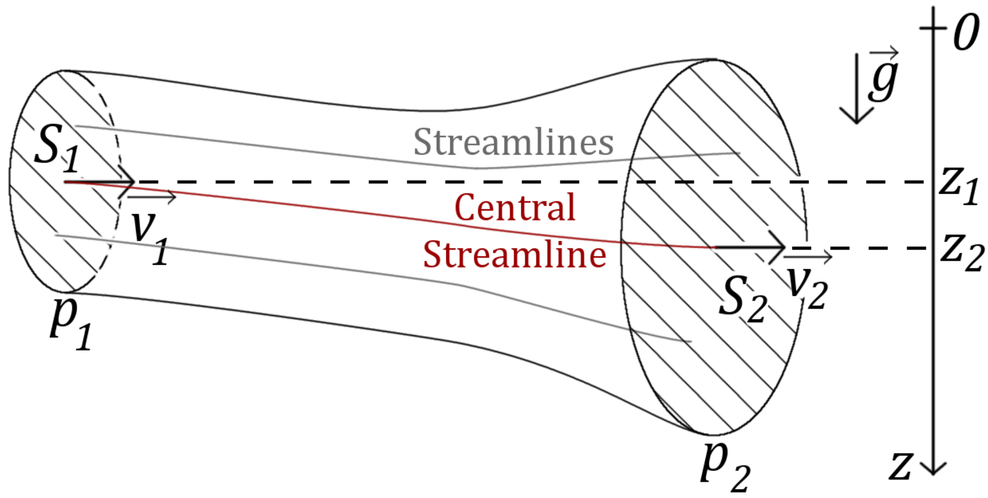


Figure 49: Description of variables used in the Bernoulli principle.

Thus, for the convection analysis of the 3D architecture, we use this simplified modeling by default, as the Navier-Stokes equation solving is not yet available. However, the resulting approximate values will be compatible with the 1.2.6. "Estimation of the order of magnitude of component temperatures" requirement. Whereas convection modeling is frequently needed, since usual industrial systems are cooled by convection, the radiation analysis is often considered negligible for the design of usual systems, except for systems such as satellite or jet engine.

3. Radiation modeling

Radiation occurs when considering two faces, where the first dissipates a heat flow by radiation and the second receives part of this heat flow from the first. To consider the proportion of heat that transmitted from one face to another, a multiplier coefficient is introduced. This coefficient, named the **view factor** and noted $F_{e \to r}$, allows calculating the received heat ϕ_r as a function of an emitted heat Φ_e.

$$\phi_r = F_{e \to r} . \Phi_e \tag{27}$$

The view factor $F_{e \to r}$ depends on the dimensions and the orientation of the faces, as well as the distance between these faces. It is calculated using the following equation:

$$F_{1 \to 2} = \frac{1}{\pi . S_1} . \iint_{S1} \iint_{S2} \frac{\cos(\theta_1) . \cos(\theta_2)}{r^2} dS_2 \, dS_1 \tag{28}$$

Where:

Symbol used (French standard)	**US standard symbol**	**Name**	**Unit (S.I.)**
dS_1	dA_1	infinitesimal surface element of S_1/A_1	m^2
dS_2	dA_2	infinitesimal surface element of S_2/A_2	m^2
θ_1		angle between the normal vector to dS_1/dA_1 and the line formed by the gravity center of dS_1/dA_1 and the gravity center of dS_2/dA_2	rad
θ_2		angle between the normal vector to dS_2/dA_2 and the line formed by the gravity center of dS_2/dA_2 and the gravity center of dS_1/dA_1	rad
r	s	distance between the gravity center of dS_2/dA_2 and the gravity center of dS_1/dA_1	m

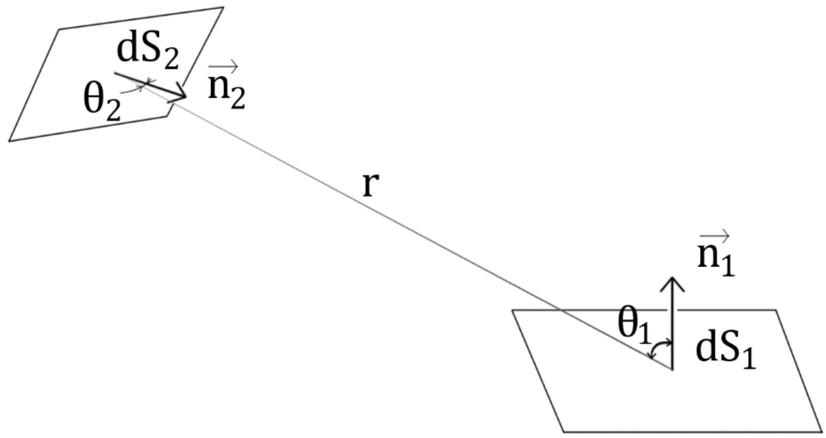


Figure 50: View factor calculation parameters.

When the receiving component receives heat energy by radiation, its temperature increases and it emits a radiation heat (according to Stephan Boltzman's law). Thus the receiving component becomes in turn an emitting component, and the initial emitting component receives in turn a part of the heat reflected from the receiving component, and so on.

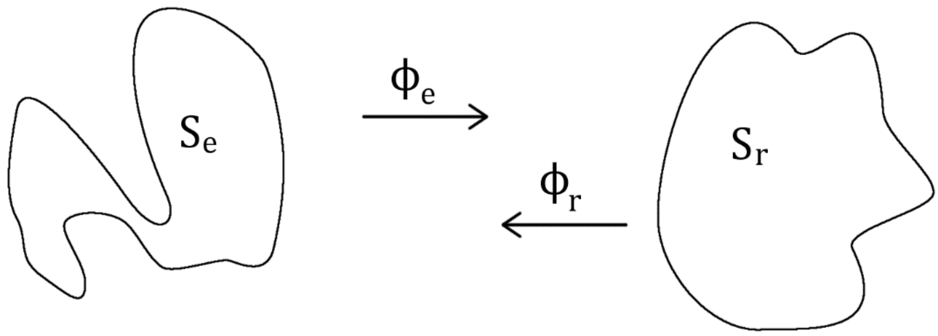


Figure 51: Radiation heat exchanges between two surfaces.

For the radiation modeling, when considering the initial emitting component, the transferred heat ϕ_{er} will be calculated as defined in the following equation:

$$\phi_{er} = F_{e\to r}.\Phi_e - F_{r\to e}.\Phi_r \qquad (29)$$

V. Thermal modeling for the SAMOS framework

In the SAMOS framework, the thermal requirement values will be compared to the resulting component temperatures in the heat-steady state.

A. Component, Medium, Interacting Face and thermal interaction network concepts

Since thermal modeling in the SAMOS framework requires its introduction in both the SysML environment (through a thermal SysML extension) and in the 3D environment, four concepts of interacting geometrical entities must be defined beforehand in addition to the usual simulation environment. The first three: Component, Interacting Face (IF) and Medium can be represented in the fourth, the thermal interaction network.

1. Component

A component is a geometrical element without any thermal behavior equation. Nevertheless, the component can have thermal properties (also named boundary conditions), for example, an emitting heat flow, a constant temperature, or a flow rate for a fluid component, but no thermal behavioral equations.

2. Medium

The concept of media has been proposed to provide a semantic support for the heat transfer considered (in accordance with the thermal propagation mode involved). A medium is a specific component with a thermal behavior. It includes a geometry (based on faces, as the heat flow is calculated from finite surfaces), in accordance with the physical reality, where the heat transfer is always supported by a given geometrical element.

Additionally, each medium contains the equations/behavior of the heat transfer mode considered. The heat transfer considered can be:

- intrinsic (when the thermal behavior occurs inside the component (e.g. volume conduction for solids or internal fluid movement for fluids),
- or coupling related between 2 different solids or between a fluid and a solid.

Therefore a medium is a component that integrates a thermal behavioral equation, whether for **intrinsic** or **coupling** behavior (that can be of a different nature: solid or fluid for intrinsic and solid-solid, solid-fluid, fluid-fluid for a coupling medium) between two elements.

Indeed, although in this chapter, we consider that the thermal behavior is analytic and can be represented by equations, the concept of medium is not limited to analytical equations and can be also used with empirical laws, experimental mapping, surrogate models, reduced models and business rules (for example, segregation distances).

3. Interacting Face

An Interacting Face (IF) is a finite surface defined by the common contact surface of two faces of two different components (with at least one medium), as described in Figure 52.

The IF can be distinguished, depending on the nature of the components (including medium) that define it, as follows:

- IF_{SS}: interacting surface between two solids,
- IF_{SF}: interacting surface between a solid and a fluid,
- IF_{SS}: interacting surface between two fluids.

Each IF will be defined with a set of average heat variables (each variable will be considered constant over the whole face), the initial value will be provided by the user and the resulting one will be calculated for the heat-steady state.

Figure 52 illustrates the IF when considering a component in contact with a medium (integrating a thermal behavior).

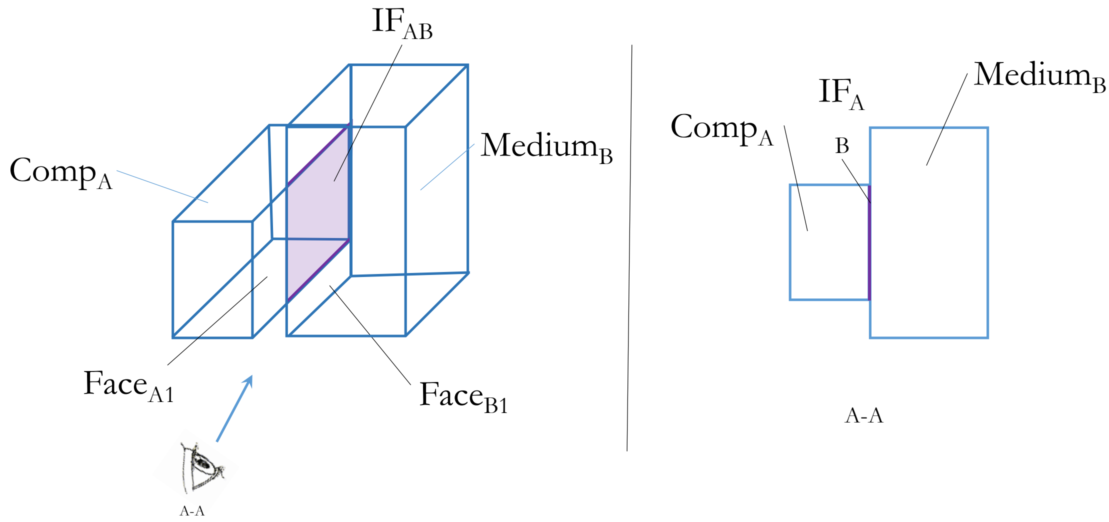


Figure 52: Example of the definition of an Interacting Face between a component and a medium.

4. Thermal interaction network

To support a consistent thermal model in the SysML, simulation and 3D environment, the previous elements will be represented in a thermal network model, and a single equivalent temperature will be allocated for each IF.

By transposing these concepts into a schematic representation:

- each component, including the medium (which can be a solid or a fluid, depending on the thermal mode propagation considered) will be considered as a node,
- faces are the port of each node, and
- a link between two faces can support an Interacting Face (**IF**) as described previously.

The previous elements will be modeled in the thermal network model by the following symbols.

Table 25 Representation of elements used in thermal interaction network

Component	**C**	
Media	**SIM**: Solid Intrinsic Medium	
	FIM: Fluid Intrinsic Medium	
	SS : Solid-Solid Coupling Medium	
	FF : Fluid-Fluid Coupling Medium	
	SF : Solid-Fluid Coupling Medium	
Interacting Face	**IF$_{SS}$** : Solid-Solid IF	
	IF$_{FF}$: Fluid-Fluid IF	
	IF$_{SF}$: Solid-Fluid IF	
Geometrical information	Bulk volume	
finite surface port	**fs**	

B. *Thermal modeling according to geometrical views*

In order to address the "1.2.12 Enabling model refinement" requirement, the geometrical multi-level view ((proposed in chapter 3) must enable choosing the geometrical element that has to be considered in terms of its heat exchanges with another component (solid or fluid): thermal specifications (boundary conditions) on the whole external component face (at the component level), 1D thermal behavior on a face, or finally 3D thermal laws on elementary faces.

As thermal analysis is based on faces, we detail the levels of geometrical refinement of the "face modeling" view according to the various thermal propagation modes.

- In the **Shell surface view**, a component or medium may exchange heat through its whole geometrical envelope considered in that case as a single interacting face (IF), which will be linked to a coupling medium towards another component or medium;
- In the **Face view**, the IF elements considered are the elementary faces of a component or medium.

1. Conduction heat transfers

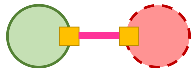
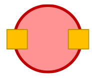
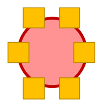
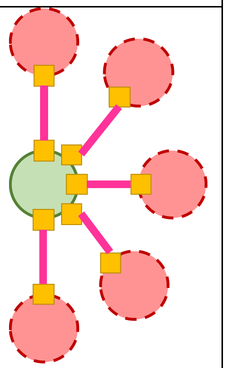
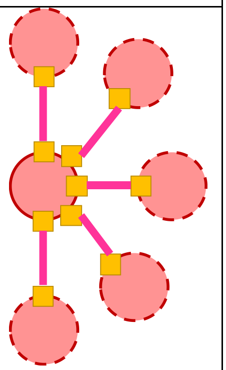
Conduction occurs in solids in two ways:

- between two solids, which can be modeled by two solid-solid components or coupling media (with their own conduction behavior), for each interaction we also need to define a coupling medium and the resulting IF$_{SS}$ is the contact face between the coupling medium and each component or intrinsic medium.

- inside a solid, this solid is represented by an intrinsic medium, where all its external faces can be considered as only one IF_{SS} when connected to a coupling medium, and it may also specify several thermal properties (viewed as the "boundary condition" of this face).

These phenomena are summarized in Table 26;

Table 26: Various conduction phenomena and their related thermal networking representation

Geometrical modeling view	Description		Is component (**C**) or medium (**M**)?	Number of linked IF	Connectable media	Network representation
Shell surface view	Contact conduction	through the whole surface	**C**	1	Coupling medium SS	
Face view	Volume conduction	between 2 faces	1D **SIM** T(s)	2	∅	
	Volume conduction	between N faces	3D **SIM** T(x,y,z)	N>2* * except for the sphere (N=1)	∅	
	Contact conduction	through N faces	**C**	N	N conduction coupling media SS	
	Volume & contact conduction	Internal conduction and through N faces	3D **SIM** T(x,y,z)	N>2	N conduction coupling medium SS	

2. Convection heat transfers

Convection occur in a fluid or at an fluid-fluid or fluid-solid interface in the three following ways:

- within a fluid (intrinsic) medium to represent the fluid motion behavior: IF_F are the external faces (perpendicular to the fluid flow direction) of the fluid in contact with another element (coupling medium SF or FF).
- between a solid and a fluid: it generates two IF_{SF}, one at the solid and one at the fluid. They are defined at the interface with a fluid-solid coupling medium: i.e. the common contact finite surface (fs) between the face of the solid element (whether component or medium) and the (solid-fluid) coupling medium, and the common contact finite surface between the face of the fluid element (whether component or media) and the (solid-fluid) coupling medium;
- between two fluids: the IF_{FF} is the common contact face between the two fluids.

These phenomena are summarized respectively in Table 27 for solids and Table 28 for fluids.

Table 27: Various convection phenomena for a solid and their related thermal networking representation.

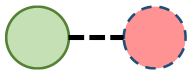
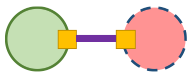
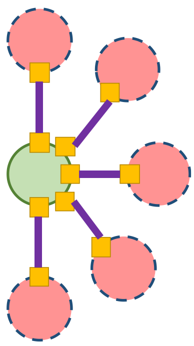
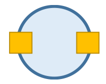
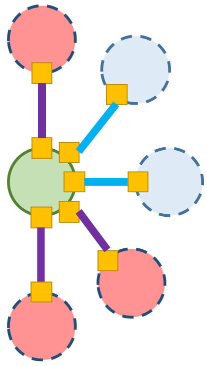
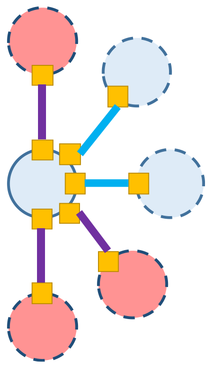
Geometrical modeling view	Description		Is component (**C**) or medium (**M**)?	Number of linked IF	Connectable medium	Network representation
Shell surface view	Volume	no heat transfer	**C**	0	Convection coupling medium SF	
	Contact convection	through the whole surface	**C**	1	Convection coupling medium SF	
Face view	Contact convection	through N faces	**C**	N	N convection coupling medium SF	

Table 28: Various convection phenomena for a fluid and their related thermal networking representation.

Geometrical modeling view	Description		Is component (**C**) or medium (**M**)?	Number of linked IF	Connectable medium	Network representation
Face view	Volume	Forced convection : fluid movement	**FIM**	2	Coupling convection medium FS or FF	
	Contact convection	Natural convection*: through N faces	**C**	N	Coupling convection medium FS or FF	
	Volume & Contact convection	Internal convection and through N faces	**FIM**	N	N convection coupling medium FS or FF	

*not covered in SAMOS

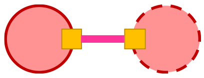
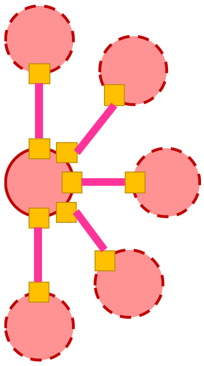
3. Radiation heat transfers

Radiation can occur on fluid or solid objects; nevertheless we will neglect fluid radiation here since it is usually neglected in conceptual design. Then, this phenomenon occurs according the two following ways:

- between two solids (to calculate their exchanged heat) there are two IF_{SS}, which are the mutual visible finite surfaces: the first addresses the interaction between the emitting component and the coupling medium, the second is defined between the coupling medium and the receiving component. Although there is no contact between the two IF considered, as these faces are solid, the coupling radiation medium will be considered as a solid-solid medium;
- for a solid emitter (to calculate the resulting component temperature), the resulting IF_{SS} is the whole external envelope surface of the intrinsic radiation medium.

These phenomena are summarized respectively in Table 27.

Table 29: Various radiation phenomena for a solid and their related thermal networking representation.

Geometrical modeling view	Description		Component (**C**) or medium (**M**)	Number of linked IF	Connectable medium	Representation
Shell surface view	volume radiation	through the whole surface*	**SIM**	1	Coupling radiation medium SS	
Face view	volume and mutual radiation	through N faces	**SIM**	N	N Radiation coupling medium SS	

* We suppose that the energy emitted by the inner faces is fully absorbed by the other inner faces.

The case where a "non-radiative" solid could simply be a geometrical barrier for the view factor considered has not been considered here, since it should consider a full-reflective medium, and it would be too difficult to calculate the additional energy ratio that will be emitted towards each element in front of it.

C. *Example of complete thermal modeling with a multiple component architecture*

a. Defining the media, components and interacting surfaces

Considering a 3D spatial distribution of components in a fluid, delimited by a boundary volume, the 3D space is cut into "slices" (that we will name layers), perpendicular to the main direction of the forced fluid flow (Figure 53). The slices are delineated for a defined set of IF: each time a new IF appears/disappears, whether through the presence of a new component (including medium) or a change of boundary condition, a new slice is created. The fluid volume within a slice is considered as a medium with a Bernoulli displacement, as described previously in (25). Finally, the whole boundary volume is then sliced into n fluid media, perpendicularly to the direction of the forced fluid flow, which we will represent through their curvilinear abscissa s.

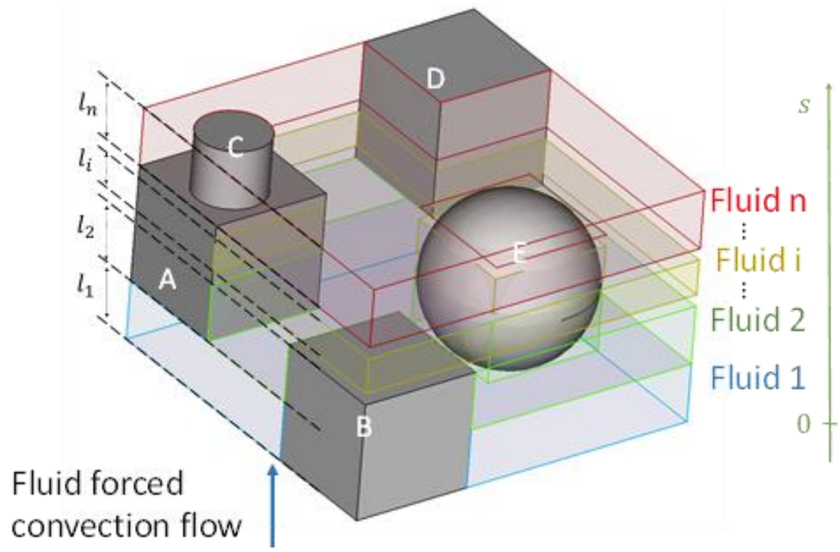


Figure 53: Medium slicing process for the example architecture.

Figure 54 presents the IF definition of the 3D architecture when considering a layer l_i.

For each fluid, medium corresponding to layer l_i (according to abscissa s), IF can be defined at each fluid-solid interface between the limited finite surface of a face (named **fs**) and the fluid, and at the fluid-fluid interface with the previous and next layers, according to the direction of abscissa s.

In parallel, the solid-solid interactions with components belonging to the previous or next layers will also generate IFs at the common fs.

Each IF related to a component K in the layer l_i will be identified by its supporting finite surface fs: Kfs_i. Then, the component-fluid interactions occurring between the finite surface fs of each component K and the fluid through (solid-fluid) will be based on a solid-fluid IF denoted $\boldsymbol{IF_{SF}Kfs_i}$, where:

- $\boldsymbol{i}$ is the layer of the fluid medium with l_i thickness,
- $\boldsymbol{K}$ is the name of the solid (medium or component) in contact with the interacting face,
- $\boldsymbol{fs_i}$ is the finite surface supporting the IF interacting with the fluid medium in layer l_i.

The fluid-fluid interactions occurring between the fluid and the fluid of a previous (l_{i-1}) or next (l_{i+1}) layer through (fluid-fluid) will be based on fluid-fluid IF noted respectively $\boldsymbol{IF_{FF}Fi_{i-1}}$, or $\boldsymbol{IF_{FF}Fi_{i+1}}$, where:

- $\boldsymbol{F}$ is the fluid considered.
- $\boldsymbol{i}$ is the layer of the fluid medium with l_i thickness,

The component-component interactions occurring through face fs of a single component K (sliced into layers) and M, belonging to two successive layers l_i and l_{i+1}, respectively, will be based on the solid-solid IF noted $\boldsymbol{IF_{SS}KFi_{i+1}}$.

- $\boldsymbol{K}$ is the name of the solid (medium or the component) in contact with the interacting face,
- $\boldsymbol{F}$ is the fluid considered,
- $\boldsymbol{i}$ is the layer of the fluid medium with l_i thickness.

The component-component interactions occurring between two components K and M, belonging to two successive layers l_i and l_{i+1}, respectively, will be based on the solid-solid IF noted $\boldsymbol{IF_{SS}KM}$**.** where

- $\boldsymbol{K}$ is the name of the first solid (medium or the component) in contact with the interacting face,
- $\boldsymbol{M}$ is the name of the second solid (medium or the component) in contact with the interacting face

The approximations of curved faces into plane faces are detailed in section VI.B

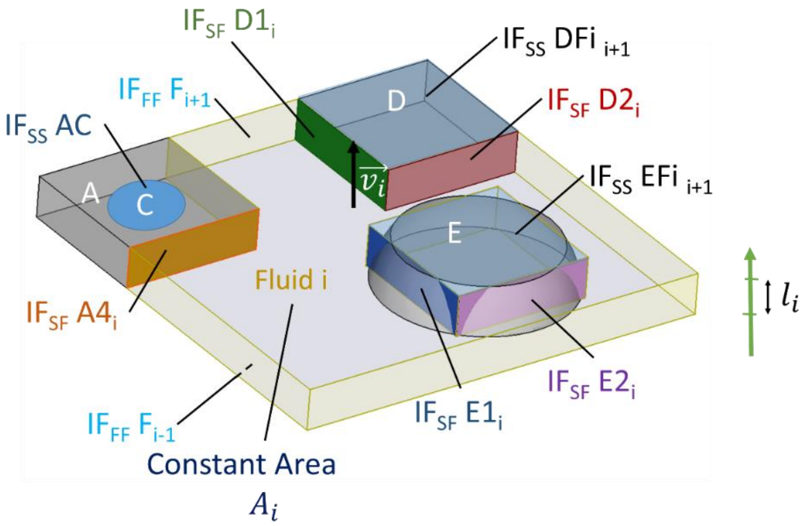


Figure 54: IF definition of the components architecture for the layer l_i.

b. Generation of the thermal interaction network

We illustrate the generation of the corresponding thermal interaction network through an example where the 2D representation is presented in Figure 55 in the case of two components A and B in layer l_2, from the lateral view, in a fluid subjected to a forced convection, and including mutual radiation.

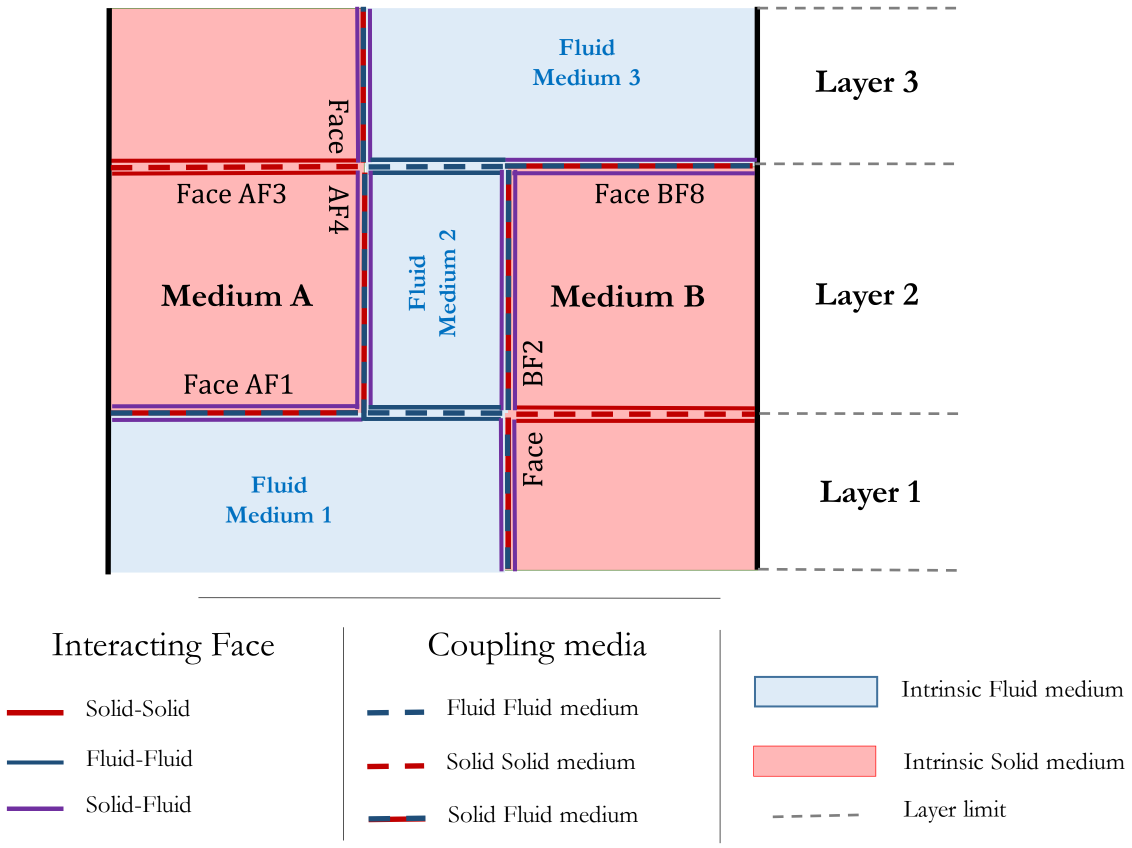


Figure 55: Example of thermal interacting faces (IF) and medium definition (2D view).

The corresponding thermal interaction network representation is shown in Figure 56. We defined a solid-fluid coupling medium for each convection interaction between the fluid and a component. Fluid-fluid coupling media are defined between each “slice” of fluid $l_i - l_{i+1}$. Indeed, in the same way, for the volume conduction of medium A (**M_A**), a conduction coupling medium has been added between the two IS_{FF} (within the **M_A)** defined between layers l_2 and l_1 and l_3, respectively.

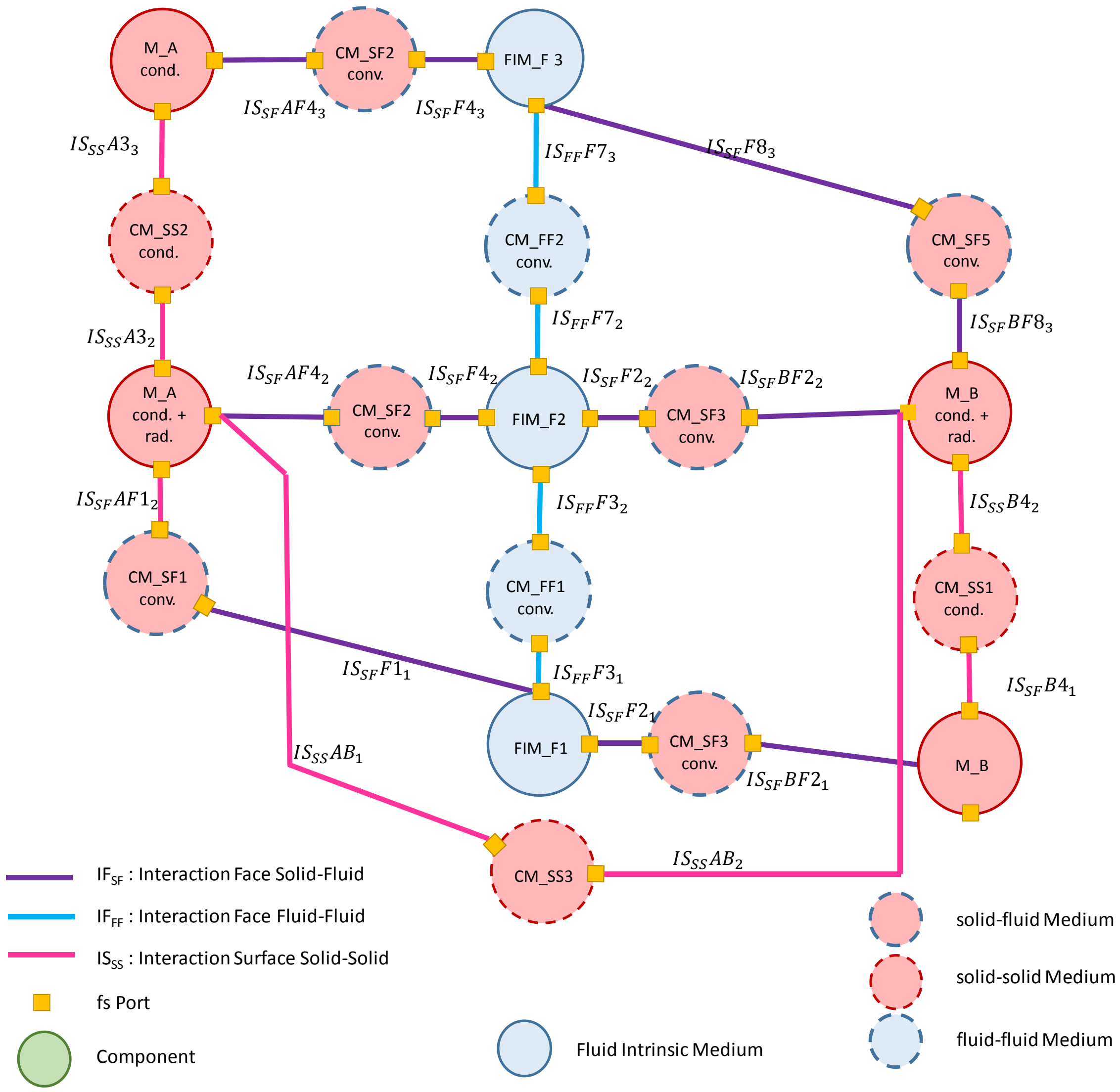


Figure 56: Example of a thermal interaction network.

Finally, this thermal interaction network will support the model both in SysML (notably in the thermal extension proposed in section VII) and in the simulation environment.

c. Temperature determination

- **The temperature of the fluid in layer l_i will result from:**

  - the component-fluid convection interactions between the components inside the fluid of layer l_i
  - the fluid-fluid convection interactions between the fluid of layer l_i and that of the previous and next layers.

- **The temperature of the components existing in layer l_i will result from:**
  - the component-fluid convection interactions (based on the IF_{SF}) between the components and the fluid contained in layer l_i;
  - the possible conduction interactions between two solids (based on the IF_{SS}), belonging to two successive layers $\{l_i$ and $l_{i+1}\}$ and $\{l_{i-1}$ and $l_i\}$, respectively;
  - the radiation between some solid component faces facing each other;
  - the possible volume and contact conduction through solid faces in contact.

Concerning the simulation:

- the initial conditions of the thermal analysis will be based on the components' thermal properties and/or on the temperature of each IF, defined by one approximate temperature provided by the user;
- then, after the simulation:
    - for each IF_{SF} included in a given layer, the heat variables (for example, temperatures) of the fluid and the corresponding component will also be considered as a constant and result from all the thermal interaction calculations occurring in layer l_i;
    - for each IF_{FF} at the interface between two layers, the Bernoulli principle provides the resulting temperature,
    - for each IF_{SS} at the interface of two components belonging to two successive layers, the temperature of contact IFs is calculated from the contact conduction medium.

VI. Use of the TTRS theory for the analytical thermal modeling

We will use the half-space concept to build the faces defining the volume of the components. They will be positioned in the 3D space using the TTRS constraints and thermally interact with each other. This geometrical construction, detailed in chapter 3 section III.B.1, facilitates the use of the TTRS theory as a support for thermal analysis. Then, we propose to take into consideration the impact of the geometrical object TTRS on the three main heat transfer modes (conduction, convection, radiation).

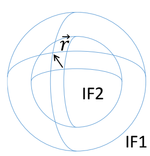
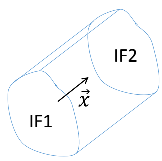
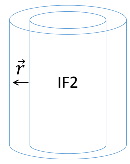
A. Conduction modeling

1. Simplification of analytical thermal laws solving

a. 1D thermal modeling

When considering the volume conduction between 2 faces (with 1D thermal law), thus with two IFs, the following table (Table 30) gives the simplified applicable laws.

Table 30: Resulting thermal analytical laws for 1D volumetric conduction according to the face symmetry.

	Sphere/Sphere	**Plane/Plane**	**Cylinder/Cylinder**
IFs geometry			
Condition	spheres are coincident	planes are parallel	cylinders are coaxial
Resulting Equation* * other faces are considered adiabatic	$\begin{cases} T(r) = \frac{A}{r} + B \\ \phi(r) = \frac{A}{r^2} \end{cases}$	$\begin{cases} T(x) = A.x + B \\ \phi(r) = A \end{cases}$	$\begin{cases} T(r) = \mathrm{A}.\ln(r) + B \\ \phi(r) = \frac{A}{r} \end{cases}$

Where: A and B are the integration constants, x is the distance between the two IFs, and r is the radius of the geometry considered.

b. 3D thermal modeling

The first intuition was to use the TTRS kinematic invariance class of each geometrical object to make the analytical thermal laws and corresponding parameters simpler.

When considering 3D heat propagation (number of IF >2) for a given 3D object, the Table 22 described in section IV.B.1, allows defining the propagation laws that have to be used. We propose to use this table when linking the coordinate system to TTRS classes (Table 31), in order to simplify solving the volume conduction laws according to the TTRS class of the TTRS resulting from the component.

Table 31: Projection bases choice for analytical thermal laws simplification for volume conduction regarding the TTRS class considered

TTRS class	Coordinate system	Analytical thermal law related projection bases
Prismatic	Rectangular	Exponential, circular, hyperbolic
Cylindrical	Circular cylinder	Bessel, exponential, circular
Prismatic	Elliptic cylinder	Mathieu, circular
Prismatic	Parabolic cylinder	Weber, circular
Spherical	Spherical	Legendre, power, circular
Revolute	Prolate spheroidal	Legendre, circular
Revolute	Oblate spheroidal	Legendre, circular
Revolute	Parabolic	Bessel, circular
Revolute	Conical	Lamé, power
Revolute	Ellipsoidal	Lamé
Revolute	Paraboloidal	Baer

Although we can use this analogy-for some simple TTRS classes (spherical, cylindrical) to simplify the analytical laws by projection on dedicated projection bases, when considering parabolic and conical geometries, which both belong to the Revolute TTRS Class, their associated projection bases are different: Bessel and Circular functions for parabolic and Lamé and power functions for conical.

Finally, it is not possible to generalize the analytical thermal law related to the projection bases according to any TTRS class of the component geometry, since it is only applicable to prismatic, spherical and cylindrical TTRS classes.

2. Thermal volume conduction analysis in complex geometries

When considering the volume conduction analysis of a medium with a more complex geometry than those with simple analytical laws, we propose to develop an automatic function to cut it into simpler geometrical objects based on its construction TTRSs. To automatically generate simple geometry for a medium, the initial medium is cut by extending all its faces according to its corresponding TTRS. An example is given in Figure 57 for an L-bar geometry.

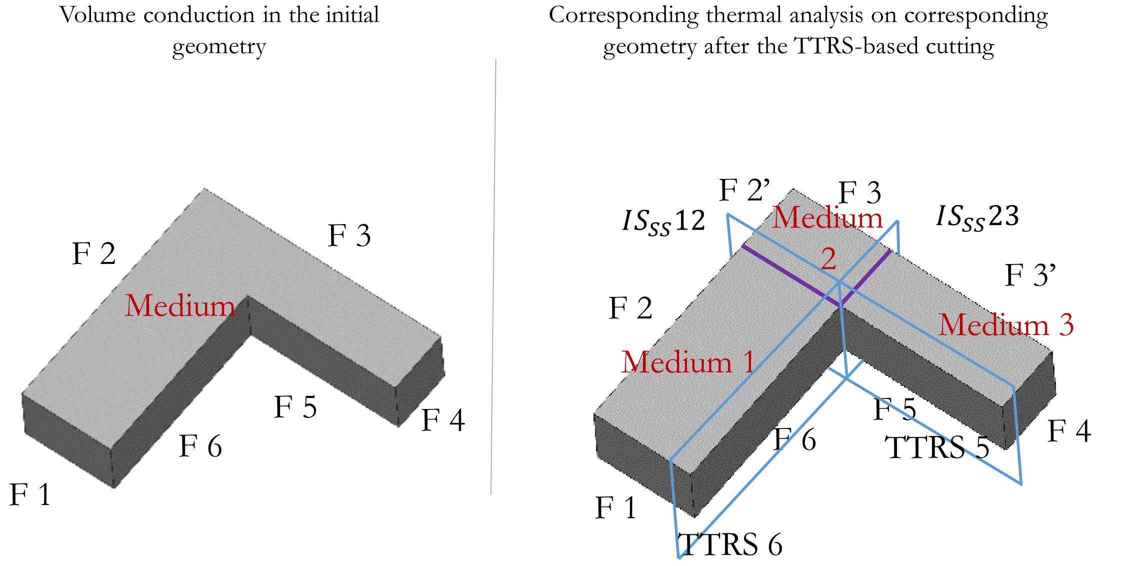


Figure 57: L-bar geometry cutting with TTRS extension.

These simpler media can be considered as a set of media with the same material including new additional contact conduction media. Then, when considering the corresponding thermal interaction network model, all the IF of the initial geometrical object (associated with the volume conduction medium) are kept and distributed on the simpler medium and new IF are added at the cutting locations (Figure 58).

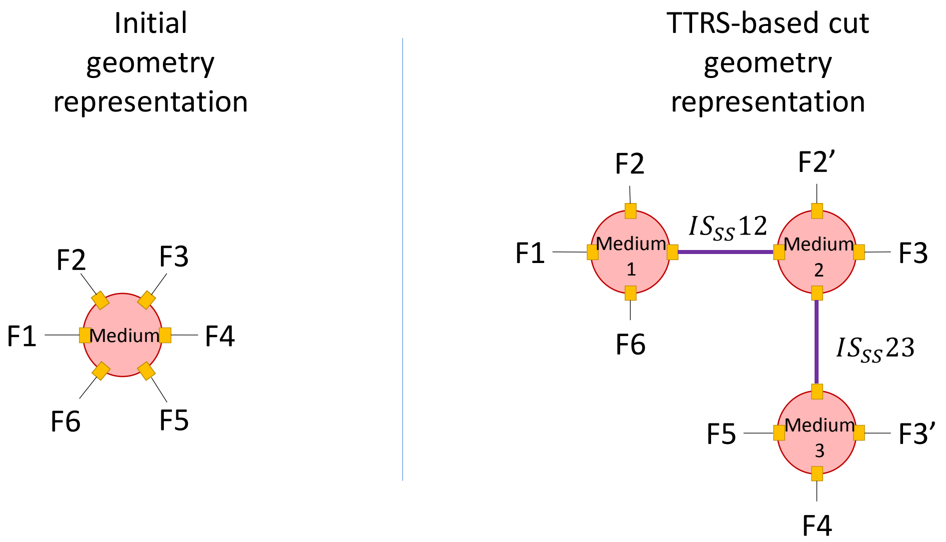


Figure 58: Thermal interaction network modeling of volume conduction in the initial geometry and in the "cut" geometry of the 2d projection using the example of an L-bar.

Using this automating cutting, it is possible to generate complex geometries, such as for example a hollow rectangle parallelepiped.

3. Contact conduction with partial contact between the faces of two solids

During the 3D spatial component positioning procedure the TTRS theory can be used to detect whether there is a contact between two solids and then identify where the contact conduction medium should be considered.

In the framework of conceptual design, when the contact between the faces of two solids is partial (i.e. the IF is not the complete area of the face), we approximate any contact area by a finite plane whose equivalent face must be determined according to the TTRS class, orientation and position of each solid. Then, the partial contact conduction modeling between two solids can be performed in the same way as the cutting procedure described previously for the volume conduction medium in the case of a complex geometry. The TTRS construction of all the faces of the first solid are used to cut the second solid. It is then possible to model the conduction from one solid to several solid faces, through a volume conduction medium in the second solid.

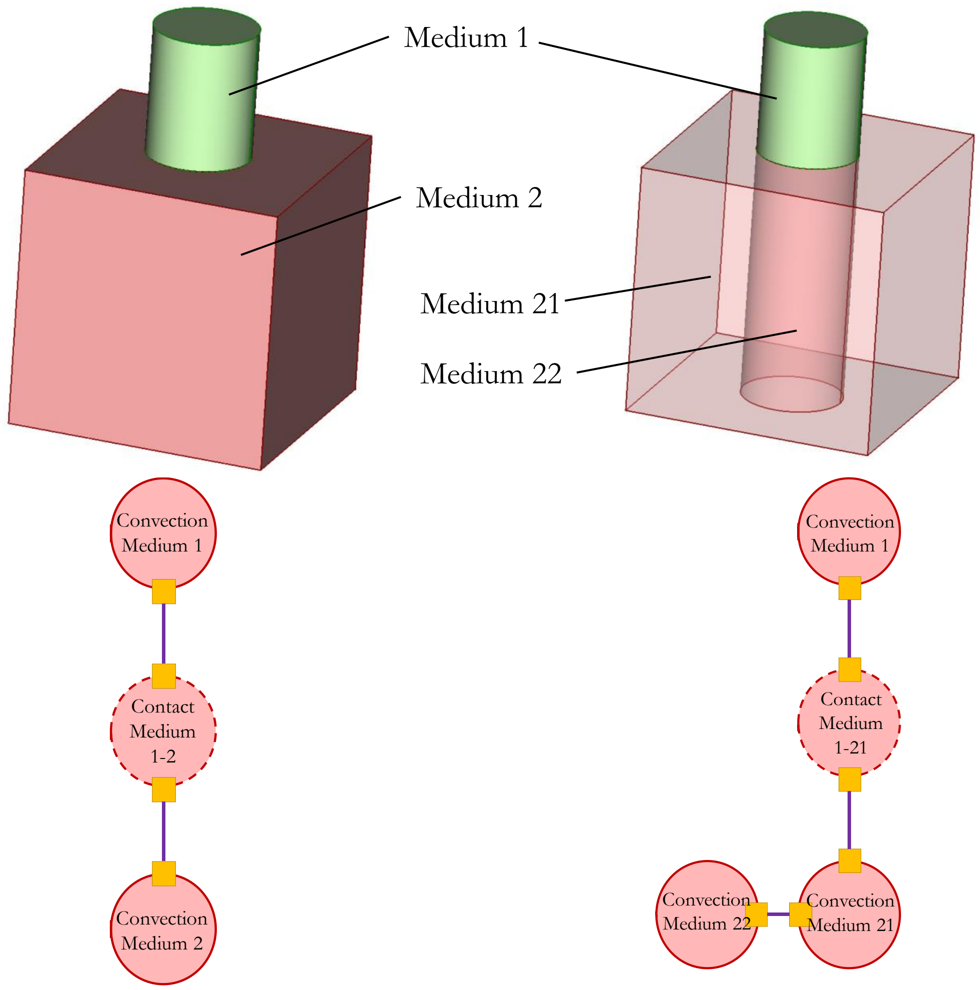


Figure 59: Coupled contact-volume conduction modeling between two components through a partial face, based on TTRS construction cutting process.

For example, when considering the conduction modeling between components A and C of the architecture described in Figure 54, the TTRS construction of the cylinder cuts the cuboid into two media (a cylinder and a cylindrical hollow cuboid) (Figure 59) to support the coupled contact-volume conduction modeling.

B. Convection modeling

The convection analysis, based on the Navier-Stokes' equations, is in fact one of the hardest phenomena to model. Since we are in the framework of conceptual design, the fluid modeling is only considered in one dimension, and the thermal convection analysis is based on the Bernoulli principle (25).

As described in the previous section, each fluid element (considered as a Medium) is cut into layers according to the IFs of architecture components. The height (li) of one layer is determined by the TTRS construction of each component face nearly perpendicularly to the main fluid direction. Considering the two external faces of each fluid medium perpendicular to the forced fluid flow, a constant average area (*Ai*) is allocated to each medium. Then the law of volumetric flow rate conservation (Eq. 26) is applied for each IF between two successive fluid media to determine the fluid velocity in this medium, based on the corresponding constant area *Ai*.

However, the limit of the proposed layer cutting process is reached if a component contains no plane TTRS normal to the main fluid direction, such as in the case of component E in Figure 54, which is a sphere. Therefore, we propose to approximate such components by a parallelepiped volume perpendicular to the fluid flow direction. In order to calculate the equivalent dimension of such a parallelepiped regarding the convection thermal analysis, we have compared two conservation equivalence models:

- The first considers the dimensions of the equivalent parallelepiped so that the contact area with the fluid is the same as for the original component geometry:

$$A_{solid} = A_{eq.parallelepiped} \quad (30)$$

The second is when the dimensions of the equivalent parallelepiped is given by the same volume as the original solid:

$$V_{solid} = V_{eq.parallelpiped} \quad (31)$$

The example of a cube chosen as the equivalent parallelepiped volume is presented in Figure 60. Its length dimension is given by the equations in Table 32.

Table 32: Calculation of the cube parameter according to the sphere size

Original geometry / **Dimension (edge a) of the equivalent cube based on**	**Sphere (radius r)**	**Cylinder (radius ρ, length l)**
Face conservation	$a = \sqrt{\frac{4.\pi}{6}}.r$	$a = 2.\sqrt{\pi.(\rho.l + \rho^2)}$
Volume conservation	$a = \sqrt[3]{\frac{4.\pi}{3}}.r$	$a = \sqrt[3]{\pi.\rho^2.l}$

We propose to evaluate the corresponding error generated by this approximation for the convection modeling, in order to verify the relevance of this approximation. Therefore, we have performed thermal convection simulations with the ANSYS® software. Two geometries were considered: a sphere approximated by a cube (Figure 60) and a cylinder approximated by a cube.

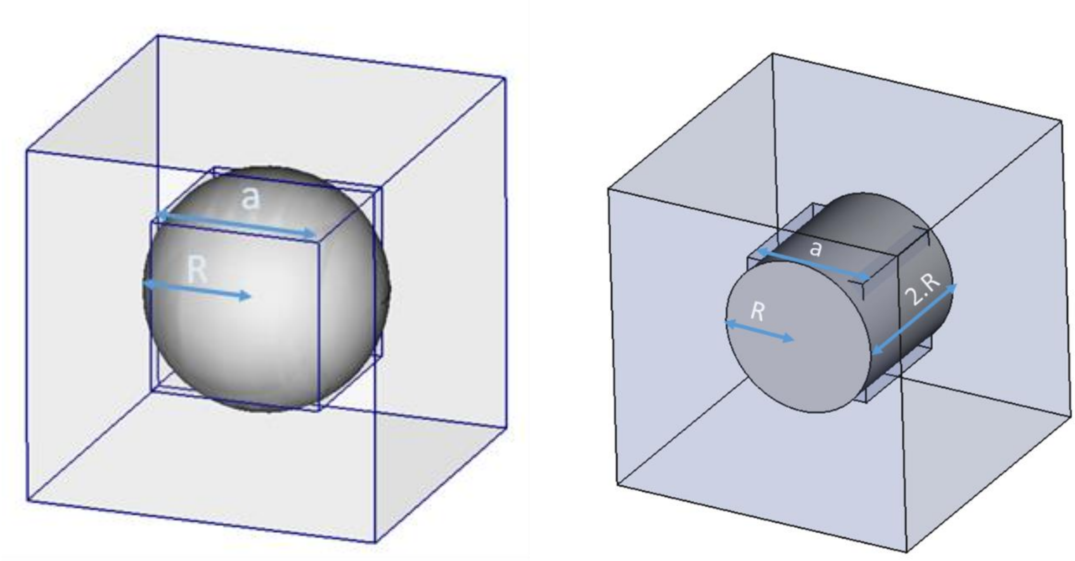


Figure 60: Geometry approximation for convection modeling

A set of simulations was performed, based on a small Design of Experiments, according to the factors and levels of factor summarized in Table 33.

Table 33: Levels definition for each DoE factor used for the convection simulation

Geometry	Heat flow	Flow	Conservation Equivalence model	Material
Sphere Cylinder (orientation normal to fluid velocity) Cylinder (collinear to fluid velocity) Parallelepiped	$\phi = 15\ W$ $\phi = 30\ W$	Laminar Turbulent	Face Volume	Water Air

Thus the number of simulations performed to verify the best adapted equivalence model is:

$$n_{simu} = ((n_{Geometry} - 1) * n_{Conservation} + 1) * n_{HeatFlow} * n_{Flow\ type} * n_{Material} \quad (32)$$

Where:

- n_{simu} is the number of simulations to perform
- $n_{Geometry}$ is the number of geometries to simulate ($n_{Geometry} = 4$)
- $n_{Conservation}$ is the number of conservation equivalence models considered ($n_{Conservation} = 2$)
- $n_{HeatFlow}$ is the number of heat flows considered as boundary conditions for the solid component ($n_{HeatFlow} = 2$)
- n_{Flow} is the number of flow types considered as boundary conditions for the fluid component ($n_{Flow\ type} = 2$)
- $n_{Material}$ is the number of fluid materials considered ($n_{Material} = 2$)

As the parallelepiped simulated is the same whatever the conservation equivalence model (only the dimensions of the sphere and the cylinder have to be modified), the resulting number of simulations to be performed becomes: $((n_{Geometry} - 1) * n_{Conservation} + 1)$, i.e. 56 simulations are necessary.

Although the heat energy chosen may appear low, since the area of the components is also small, the resulting heat flow is high:

$$\phi_l = 15\,W \Leftrightarrow \varphi_l = \frac{\phi_l}{S} = 542\,W.m^{-2} \quad (33)$$

$$\phi_h = 30\,W \Leftrightarrow \varphi_h = \frac{\phi_h}{S} = 1084\,W.m^{-2} \quad (34)$$

For each simulation result, the average temperatures at iso-volume were better than the average temperatures at iso-surface. The most critical results are highlighted by colors in Table 34. The most critical case is for an environment composed of air in the laminar regime, with a high heat flow.

Table 34: Simulation results with the maximal error found for each geometry

	Sphere	**Cylinder** orientation normal to fluid velocity	**Cylinder** collinear to fluid velocity)
Temperature at iso-surface	$306.42\,K$	$308.48\,K$	$309.72\,K$
Temperature at isovolumetric	$312.24\,K$	$311.89\,K$	$323.82\,K$
Temperature of parallelepiped	$317.21\,K$	$317.21\,K$	$317.21\,K$
Error at iso-surface	3.52%	2.83%	2.42%
Error at isovolumetric	1.59%	1.70%	2.04%

The sphere geometry has a larger error at the iso-surface than at the iso-volume (difference of $11K$ for the iso-surface, compared to a difference of $6K$ for the iso-volume). However, the error is larger for the cylinder when the axis of the cylinder is collinear to the direction of the fluid velocity.

In addition, we have also evaluated the error induced by such an approximation by employing the analytical modeling (conduction/convection coupling) that we developed, in the case of the same sphere approximated with a surface-equivalent cube, and a cylinder approximated with a surface-equivalent rectangular parallelepiped. The two geometric models will be developed as one dimensional models.

The analytical modeling supported by the different media has also been approximated by considering the following 1D analytic equations (Table 35).

Table 35: Analytic thermal equations associated with the equivalence models.

	Sphere ($var = r$)	**Cylinder ($var = \rho$)**
Volume conduction analysis in the solid (SIM)	$T(r) = \dfrac{k_1}{r + k_2}$ (35)	$T(\rho) = k_1.\ln(\rho) + k_2$ (36)
Convection analysis in the fluid (SIM)	$Tf(x) = k_3 \exp\left(\dfrac{h\ (2 * var)}{\rho\ V\ A\ C_p} x\right) + Tf_{inlet}$ (37)	
Flow imposed in the center of the solid	$-\lambda \dfrac{dT(var_{int})}{dvar} = \phi_0$ (38)	
Contact between the solid and the fluid (CM)	$-\lambda\ \dfrac{dT(var_{ext})}{dvar} = h\ (T(var_{ext}) - \dfrac{\int_0^l Tf(x)dx}{l})$ (39)	

Where:

Symbol used (French standard)	**US standard symbol**	**Description**	**Unit (S.I.)**
ϕ_0	Q_0	heat flow imposed at the center of the solid	W
Tf	Tf	temperature of the fluid	K
Tf_{inlet}	Tf_{inlet}	inlet temperature of the fluid	K
(k_1, k_2, k_3)	(k_1, k_2, k_3)	integration constants to be calculated	Dimensionless

We can see with (39) that the fluid temperature is considered constant along the IF. The term $\frac{\int_0^l Tf(x)dx}{l}$ allows calculating its average value.

Furthermore, we also evaluate the error difference between this analytical thermal coupled model, with an equivalent model using the ANSYS® software, including a CFD analysis with the CFX solver. Two geometries are considered: a cylinder and a sphere (Figure 61).

The average error between the finite elements analysis and the analytic model is 0.18% with a maximum of 0.21% for the sphere, and 0.85% with a maximum of 0.86% for the cylinder.

These results are promising for validating our approach. Thus this method is validated for these components. Nevertheless, a comprehensive analysis of other geometries must be performed in future work.

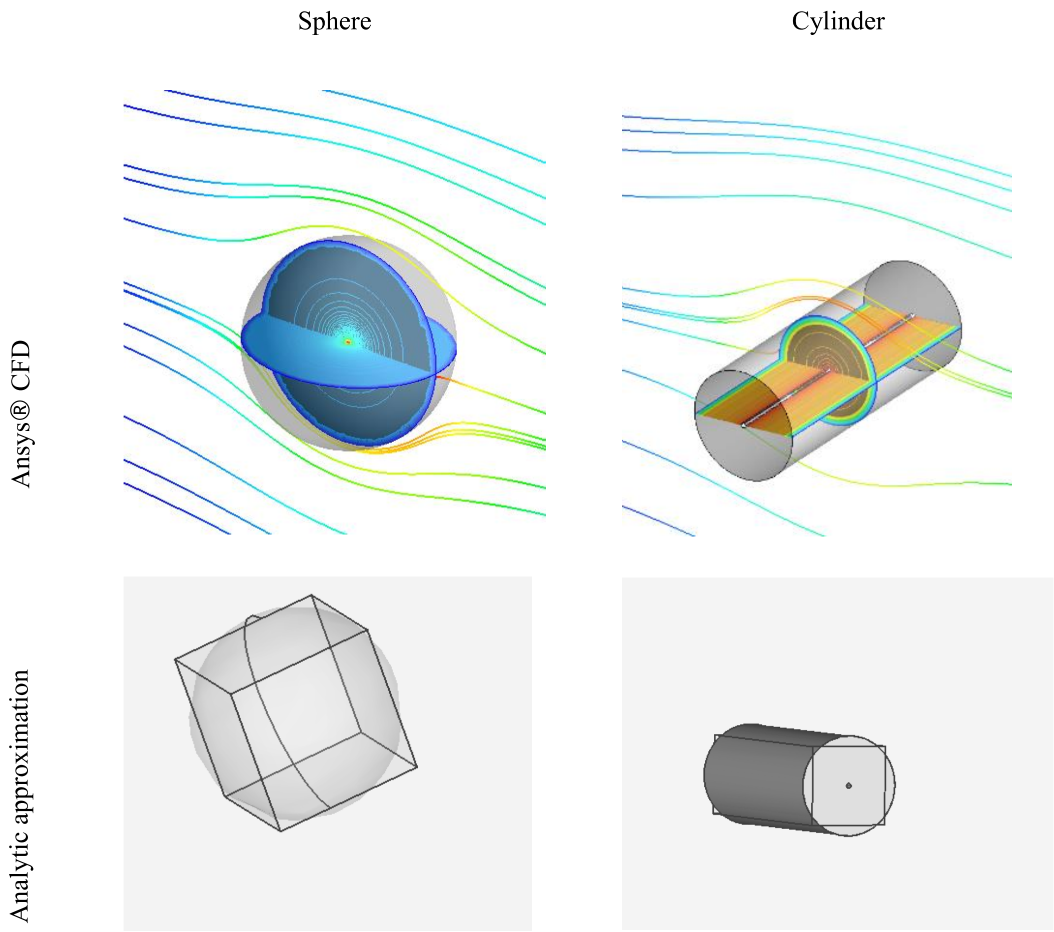


Figure 61: Finite element modeling for the geometry approximation evaluation.

C. *Radiation modeling*

The main issue in the radiation modeling lies in the calculation of the view factor. The view factor $F_{e \to r}$ r represents the part of the effective transmitted heat flow from an emitting component to a receiving component. It is usually calculated numerically by finite element analysis: the external area of a solid is divided into small squares whose view factors are provided via the Strokes theorem (Walton, 2002). Then, a summation of all the view factors is performed in order to calculate the global view factor of the solid.

However, some authors propose correlations for simple geometries (Krishnaprakas, 1997) (Mathiak, 1985) (Feingold & Gupta, 1969). Thus we propose to use them in order to calculate the view factor between two components regarding their TTRS and associated positioning constraints, to determine the equivalent IF.

For example, the view factor between two spheres is different depending on whether we consider a distance (C2 TTRS constraint) or a coincidence (C1 TTRS constraint) constraint between them (Howell, 2014) (Figure 62).

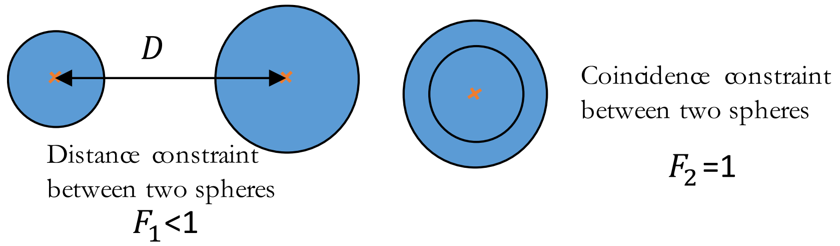


Figure 62: Different view factors between two spheres depending on the TTRS constraint considered.

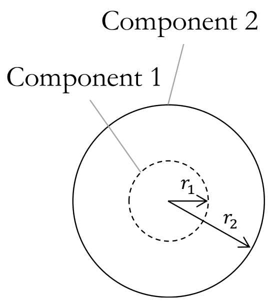
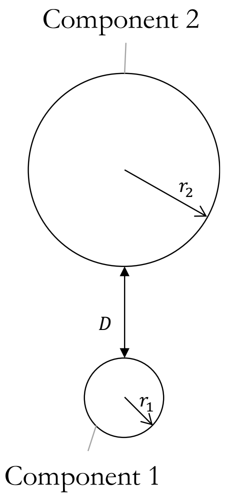
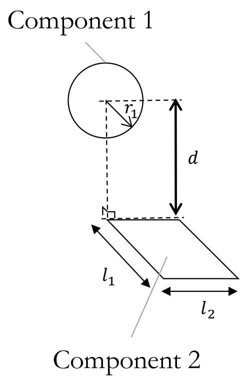
Thus we determine the view factor between 2 TTRS depending on their TTRS class and their relative TTRS positioning constraint, considering the 3 following TTRS classes: Planar, Cylindrical and Spherical, based on the works of Howel (Howell, 2014).

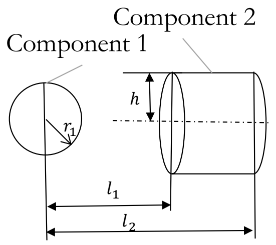
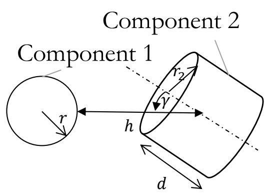
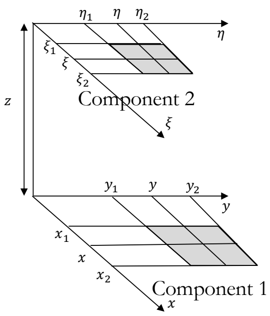
The set of constraints available between these three TTRS classes are summarized in Table 36.

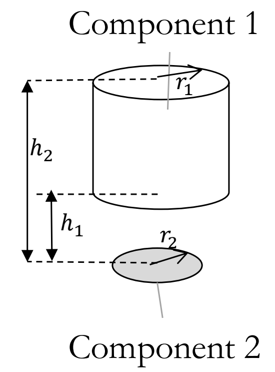
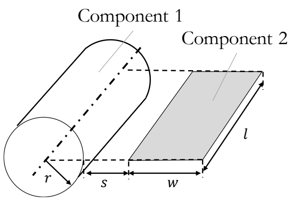
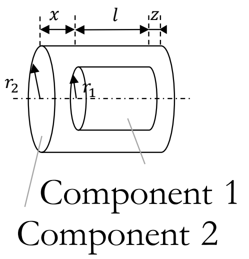
Table 36: Associated constraints between the three TTRS classes considered for the view factor calculation.

TTRS class	Cylindrical (MRGE=line)	Planar (MRGE = plane)	Spherical (MRGE = point)
Cylindrical (MRGE=line)	D1=D2 (C_C): C11 D1//D2 & D1≠D2 (C_T): C12 Else (C_X) : C13	D2⊥P1 (CR) : C8 D2 // P1 (CT) : C9 Else (CX) : C10	O1∈D2 (C_R) : C4 Else (C_X) : C5
Planar (MRGE = plane)		P1//P2 (C_P) : C6 Else (C_T) : C7	(C_R) : C3
Spherical (MRGE = point)			O1 = O2 (C_S) : C1 Else (C_R) : C2

Table 37: Expression of View factors according to the TTRS constraints involved.

Constraint	Figure	Equations
C1		$\begin{cases} F_{1\to2} = 1 \\ F_{2\to1} = \left(\frac{r_1}{r_2}\right)^2 \end{cases}$
C2		Abacus
C3		$F_{1\to2} = \frac{1}{4.\pi}.\tan^{-1}\left(\frac{l_1.l_2}{d.\sqrt{{l_1}^2 + {l_2}^2 + d^2}}\right)$

Constraint	Figure	Equations
C4		$F_{1\rightarrow 2} = \frac{1}{2.\pi.h}\left\{ l_2.\tan^{-1}\sqrt{\frac{r^2-h^2}{{l_2}^2+h^2-r^2}} - l_1.\tan^{-1}\sqrt{\frac{r^2-h^2}{{l_1}^2+h^2-r^2}} - \frac{l_2}{h.\sqrt{{l_2}^2+h^2}}.\cos^{-1}\sqrt{\frac{h\sqrt{{l_2}^2+h^2-r^2}}{r.l_2}} + \frac{l_1}{h.\sqrt{{l_1}^2+h^2}}.\cos^{-1}\sqrt{\frac{h\sqrt{{l_1}^2+h^2-r^2}}{r.l_1}} \right\}$
C5	 Only if $r_2 \gg d$	Abacus
C6		$F_{1\rightarrow 2} = \frac{1}{4.l.L}\sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{2}(-1)^{(i+j+k+l)}.G(x_i, y_j, \eta$
C7		N/C

Constraint	Figure	Equations
C8		$F_{1\to 2} = \frac{r_2^{\,2}}{4.r_1(h_2-h_1)}\left[\frac{h_1^{\,2}-h_2^{\,2}}{r_2^{\,2}} - \sqrt{\left(\frac{h_1^{\,2}+r_1^{\,2}+r_2^{\,2}}{r_2^{\,2}}\right)^2 - 4.\left(\frac{r_1}{r_2}\right)^2} + \sqrt{\left(\frac{h_2^{\,2}+r_1^{\,2}+r_2^{\,2}}{r_2^{\,2}}\right)^2 - 4.\left(\frac{r_1}{r_2}\right)^2}\right]$
C9		Abacus
C10		N/C
C11		$F_{1\to 2} = 1 + \frac{x}{l}F_x + \frac{z}{l}F_z - \frac{l+x}{l}F_{l+x} - \frac{l+z}{l}F_{l+z}$
C12		N/C
C13		N/C

Where:

$$G = \frac{1}{2.\pi}\left((y-\eta)\sqrt{(x-\varepsilon)^2+z^2}.\tan^{-1}\left\{\frac{y-\eta}{\sqrt{(x-\varepsilon)^2+z^2}}\right\} + (x-\varepsilon\,)\sqrt{(y-\eta)^2+z^2}.\tan^{-1}\left\{\frac{y-\varepsilon}{\sqrt{(x-\eta)^2+z^2}}\right\} - \frac{z^2}{2}.\ln\{(x-\varepsilon)^2+(y-\eta)^2+z^2\}\right)$$

$$F_\varepsilon = \frac{\varepsilon^2 - {r_1}^2 + {r_2}^2}{8 * r_1 * \varepsilon} + \frac{1}{2\pi}\left\{\cos^{-1}\left(\frac{\varepsilon^2 + {r_1}^2 - {r_2}^2}{\varepsilon^2 - {r_1}^2 + {r_2}^2}\right) - \frac{1}{2.\varepsilon}.\sqrt{\frac{(\varepsilon^2 + {r_1}^2 + {r_2}^2)^2}{{r_1}^2} - 4}.\cos^{-1}\left(\frac{(\varepsilon^2 + {r_1}^2 - {r_2}^2).r_1}{(\varepsilon^2 - {r_1}^2 + {r_2}^2).r_2}\right) - \frac{\varepsilon^2 + {r_1}^2 - {r_2}^2}{2.\varepsilon.r_1}.\sin^{-1}\left(\frac{r_1}{r_2}\right)\right\}$$

VII. TheReSE SysML extension

To carry out an MBSE approach (with the possibility of specifying and tracing thermal requirements), the SysML System model must be semantically enriched by a thermal extension. The SysML extension for thermal modeling we propose (called **TheReSE**: **The**rmics **Re**lated **S**ysML **E**xtension) is based on the previous SysML extension called GERTRUDe (**G**eometrical **E**xtension **R**elated to **T**TRS **R**eference for a **U**nified **De**sign) (Chapter 3 67IV). Indeed, as described previously, a thermal behavior is based on geometrical information, thus it normal that the thermal SysML extension is built using geometrical data.

This extension has to include each heat transfer mode described previously. Moreover, when developing this new extension, the difficulty is also to take into consideration the different interactions between the heat transfer modes, as they cannot be studied separately from the others. Thus the extension has to include the type of the coupling considered, whether weak or strong, and the different influence parameters. For example, the convection -conduction coupling may differ from one application to another, thus these different coupling equations must be included in this SysML new extension.

The data model of TheReSE based on GERTRUDe is shown in Figure 63. TheReSE allows enriching blocks with another stereotype called "Component" which already contains its geometry model that conforms to the TTRS theory.

When the System Architects wish to define thermal specifications, the extension should enable them to create "dynamic" quantifiable requirements. We have defined the corresponding stereotype of a SysML requirement, called "*Quantified requirement*". The System Architects must ensure both the consistency of parameters used in the behavior equations and the traceability of the simulation results which have to respect the threshold values previously defined by the requirements in the SysML system model. Although this model currently provides only textual requirements, the integration of required parameter intervals for a component or a material are interesting. It will be much easier to transfer requirements to the MA, if the blocks have automatically inherited the acceptable temperature limit interval which the component has to resist. These "required" parameters could also be used efficiently in the behavioral diagram with thermal equations.

Regarding the modeling of thermal propagation modes, we defined a new diagram stereotype in TheReSE, based on a component structure diagram, called *internal physics diagram*. For convection and conduction, the corresponding thermal equation must be considered as linear following a thermal resistance simplification. With this approximation, it is possible to link several components on the same port and to apply Kirchhoff's law, by adding the thermal flux of each contribution algebraically. This approximation cannot be applied for radiation, and we assume that only one radiation source will interact for a given surface. In the case of two sources to be taken into account, the System architects can compare the contribution of each source and see if one can be neglected. If this is not the case, the Simulation team, geometers and System architects are warned and they can then decide to take precautions by adding the two heat contributions.

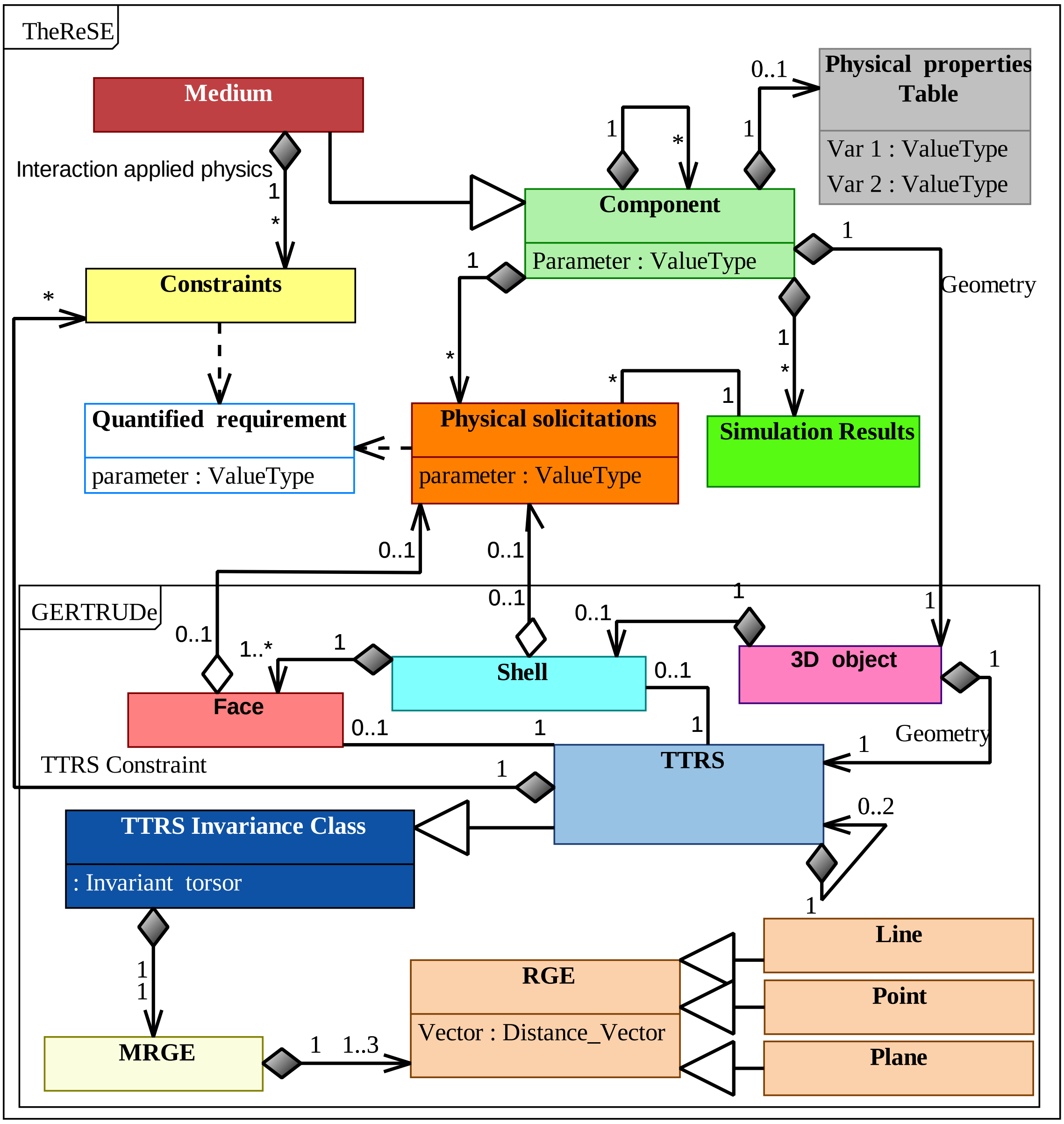


Figure 63: Data model of TheReSE.

To take into consideration a connection component through which the flow passes between two components, TheReSE proposes a stereotype block called "*Media*". *Media* is a component with all the properties (concrete geometry, emitting and receiving thermal geometry) and parameters required for management by thermal simulation. This medium can be a solid, fluid, gaseous, or even empty. If the medium is solid, for example a physical support between two components, it is possible in all cases to add its geometry with GERTRUDe and its material properties (even if the medium is empty). The medium also contains the *Constraints Blocks* of thermal propagation equations, because the thermal exchanges occur in/through the medium. It is important to note that if a finite surface fs port of a component is not connected to a medium in the internal physics diagram, it is connected to the "external environment" media by default (in order to facilitate the use of TheReSE). Every time a connection between an fs port of a component and a medium is added, a new constraint (through a ϕ rate of heat flux term) is added to the heat balance. The information enrichment provided by TheReSE has to be substantial enough to prepare simulation modeling. However the System architects can sometimes provide a physical architecture with partial geometrical and thermal data from TheReSE. It is therefore important to allow the simulation teams to propose completing this information to simulate the architecture's behavior under thermal constraints and to trace these complementary data back to the System architects, by integrating them in the SysML model through TheReSE.

VIII. Conclusions

This chapter described the thermal modeling procedures that will be implemented in the SAMOS framework. Before describing the thermal modeling approach chosen, the corresponding derived requirements were generated and thermal hypotheses and reminders provided. Then, the three main existing thermal modeling that can be used for SAMOS, were described: analytic calculation, thermal resistance and finite element analysis. Regarding the previous requirements, the analytic calculation was chosen and detailed. After having described the concepts of medium, component and interactive faces that will be used in the thermal interaction network, attention was given to the different thermal propagation modes relating to the geometrical view addressed. The TTRS concept was applied regarding different thermal analysis cases. Finally, the development of the Thermal Related SysML Extension (TheReSE) according to this thermal modeling approach was described.

Although the data model of TheReSE was provided, its implementation was been described. The development of this implementation involves deriving new requirements, notably industrial requirements. The next chapter will present all the implementation developments, including model transformation from GERTRUDe to 3D CAD modeling, and from TheReSE to simulation modeling.

Chapter 5 – Implementation of the Thermal 3D Sketcher

I. Expression of needs and corresponding tool selection ... 130

A. Demonstrator implementation requirements 130

B. Tool selection 132

1. SysML tool 132
2. 3D CAD tool 133
3. Simulation language and tool 135
4. Validation of data interactions management for the tools selected 138

II. Algorithmic modeling of the Thermal 3D Sketcher 140

A. Description of the geometrical model transformation 140

1. Geometrical modeling transformation process 140
2. Transformation Metamodel 141
3. CAD2GERTRUDe transformation 144
4. GERTRUDe2CAD transformation 145

B. Descriptions of the thermal model transformations and algorithms 149

1. Thermal modeling transformation process 149
2. Model transformation metamodel 150
3. Transformation Modelica2TheReSE and TheReSE2Modelica 152
4. Volume slicing algorithm for convection modeling 152

III. Developments in SysML (Atego-PTC environment) ... 153

A. GERTRUDe 153

1. Profile diagram 153
2. Geometry library: the Geometry library 157

B. TheReSE 157

1. Profile diagram 157
2. Scripts 159

C. Layer Management to improve readability 159

IV. Development in 3D CAD (FreeCAD environment) 162

A. Geometric constraint solving 162
1. Different constraint solving methods 162
2. Model transformation 165
B. SAMOS processing 166
1. Definition of geometry 166
2. Simulation setup 166
V. Development in Modelica (OpenModelica environment) 174
A. Development of a Modelica Library related to the shell view. 174
1. New ports added 174
2. Boundary conditions 176
3. TTRS-based convection-coupled conduction modeling elements 178
B. Model transformation and perspectives 179
VI. Conclusions 180

The previous chapters present the various approaches and available theories to meet the issue of the 3D architecture assessment under geometrical (chapter 3) and thermal (chapter 4) constraints during the conceptual design, and its corresponding design interaction management (chapter2).

The global framework of the sketcher (Figure 64) deals with the management of two SysML extensions. These extensions allow enriching SysML with geometrical and thermal semantics, thanks to GERTRUDe and TheReSE. The GERTRUDe and TheReSE models allow automatically generating a geometrical model in the 3D CAD tool, and simulation modeling. The results are traced back in the 3D CAD tool. The 3D CAD tool should allow constructing the parts, adding or modifying thermal behavior, managing the simulation, and displaying the results. This 3D CAD tool is called the Thermal 3D Sketcher.

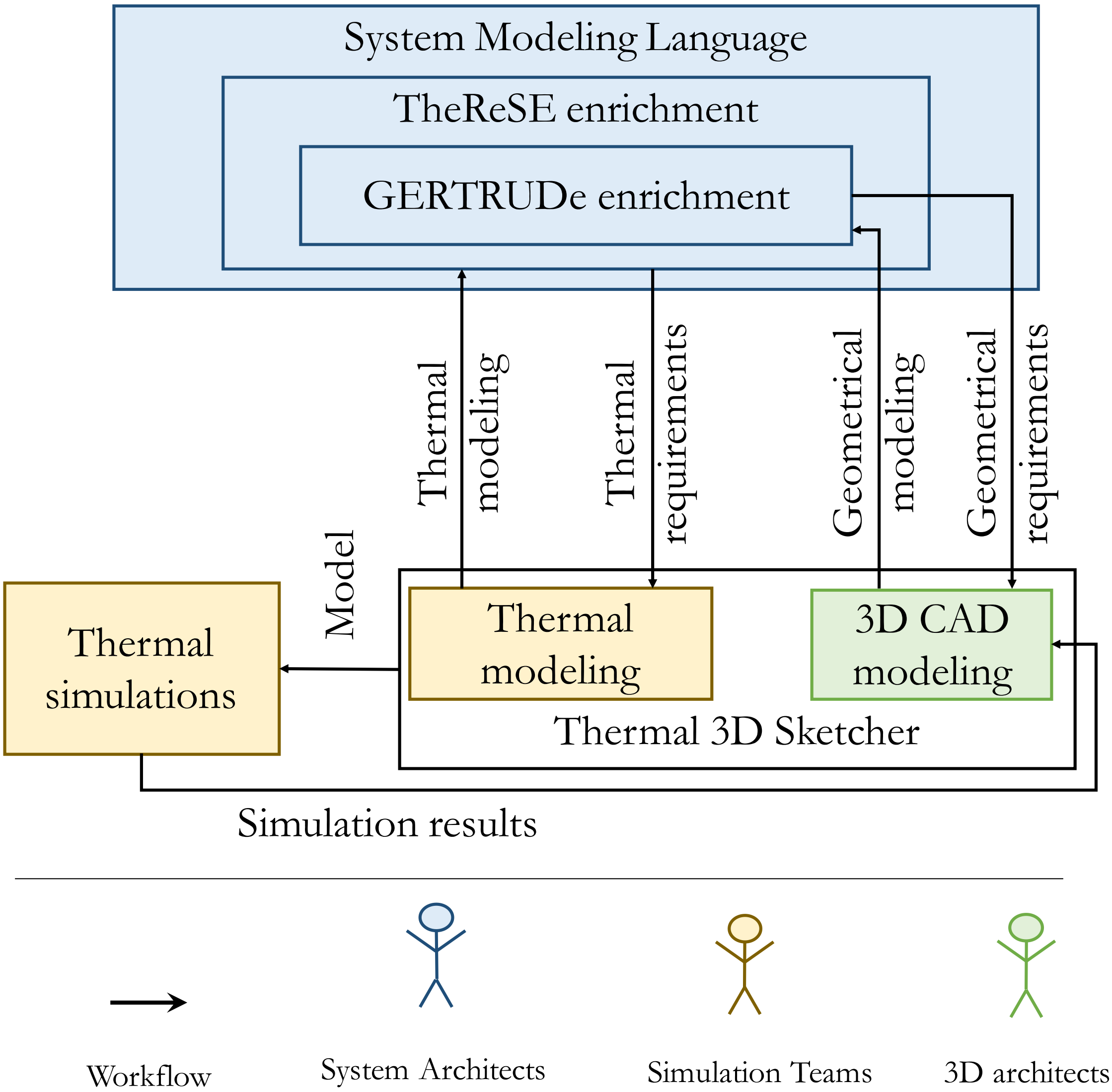


Figure 64: Global framework of the thermal 3D Sketcher.

In this chapter, we present the Thermal 3D Sketcher, which is the tool developed to demonstrate the ability of the proposed approaches to be used in an industrial context. We first choose the tools used to develop the thermal 3D Sketcher. Then the algorithms used to develop the thermal 3D sketchers are provided. Afterwards, the development is provided for the SysML environment, the 3D environment, and the Simulation environment.

I. Expression of needs and corresponding tool selection

A. Demonstrator implementation requirements

As the geometrical modeling is usually performed in a 3D CAD tool, and thermal analysis has to be performed using analytic modeling based on the component geometry defined by the TTRS theory, the Thermal 3D Sketcher has to meet the implementation derived requirements (described in Figure 65) and the model transformation related requirements.

The additional requirements related to these implementation constraints are described in Table 38

Table 38: Implementation derived requirements

#Id	Name	Description
1.1.7	Customizability of the metamodel	As the transformation has to be bidirectional (as Gross et al. demonstrated (Gross & Rudolph, 2016)), the metamodel of each tool needs to be modifiable in order to guarantee that information can also be exchanged from and toward the tool considered.
1.1.9	GUI (ergonomics)	Good GUI ergonomics is essential for the main tool that will support the SAMOS framework, to support industrial use, notably for collaboration during the conceptual design between design actors.
1.1.10	Communication with other tools	An important requirement is that each tool must enable communication with other tools involved in the model transformation.
3.3	Scripting language	In order to satisfy the ease of development requirement, the scripting language of the tool must be accessible to the developer.
3.4	Open source	Another requirement that can satisfy the ease of development is to use an open source software application.
3.5	Known software	Finally, the "ease of development" requirement can be satisfied provided that the developer has good knowledge of the software.

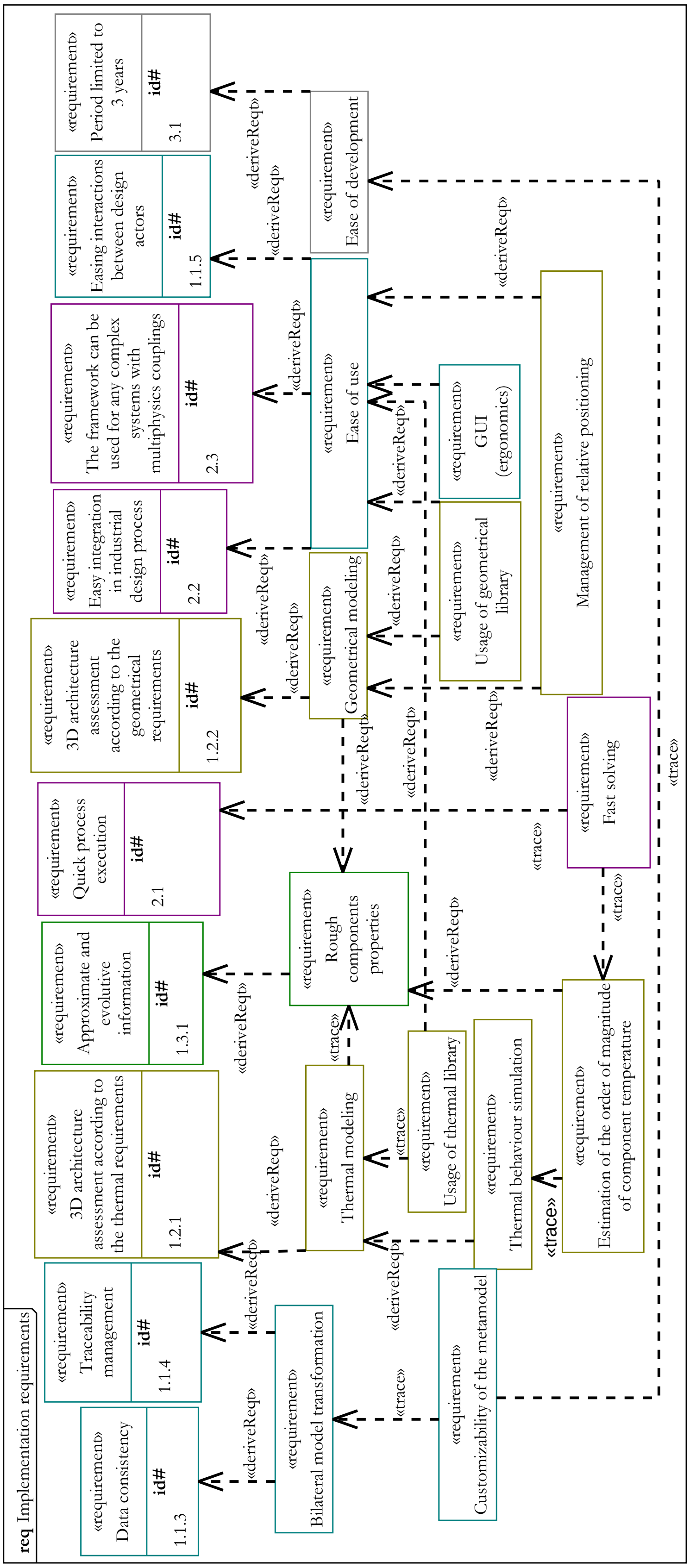


Figure 65: Geometrical and thermal related implementation derived requirements.

The tool that will be used as the main support will be the 3D CAD tool, as it is the most suitable tool for 3D architecting, thus it will be the main user interface for exchanging information from/to both the SysML tool and the simulation tool.

B. *Tool selection*

The selection of the software is more related to the period limited to 3 years requirement, as the demonstrator illustrates a feasibility concept which could be developed using other software. However, the judicious choice of these tools can reduce the development time of the demonstrator. Thus an initial analysis of the needs concerning these tools is performed before evaluating several common software tools.

1. SysML tool

The requirements related to the selection of the tool implementing SysML are described in Figure 66. The red contours indicated the requirements used as criteria for the corresponding tool selection.

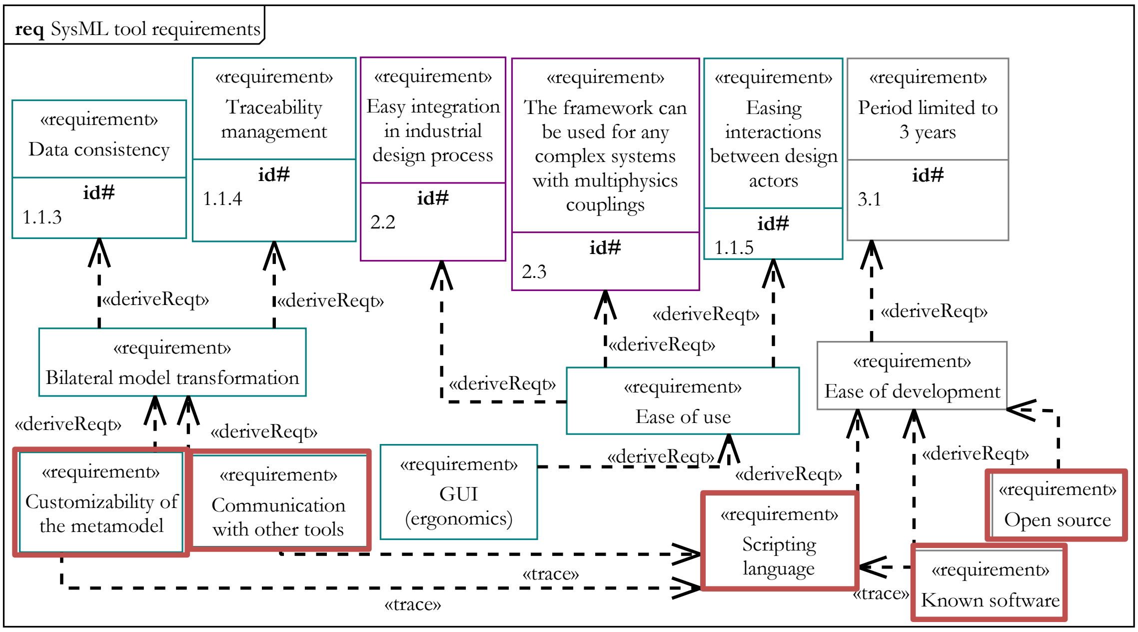


Figure 66: SysML tool selection related requirements

We propose to compare several best-known SysML software editors in Table 39, to select that which meets the previous requirements.

Table 39: Analysis of main tools based on SysML regarding implementation requirements.

Name *(SysML spec. version supported)*	**Scripting language** *(good knowledge by the developer)*	**Customizability of its metamodel**	**Communication with other tools** *(available languages)*	**Good knowledge of software**	**Open source of SysML modeler**
Artisan Studio 7.4 /verified with 8.2 *(0.98 / 1.2)* (PTC, 2017)	VB.Net (✓)	✓	COM Interface *(VB, Python)*	✓	✗
Papyrus 1.9.1 *(1.1 & 1.4)* (Eclipse, 2017)	Java (✗)	✓	COM Interface *(Jacob library)*	✗	✓
Modelio 3.4 *(adapted from 1.2)* (ModelioSoft, 2017)	Jython (✗)	✓	COM interface *(VB, Python)*	✗	✗
MagicDraw 18.2 *(1.3)* (No magic, 2017)	Jython, BeanSheel, Groovy, JRuby, or JavaScript. (✗)	✓	COM Interface *(Jacob library)*	✗	✗
IBM Rhapsody 8.0.6 *(1.3)* (IBM, 2017)	Java (✗)	✓	COM interface *(VB, Python)*	✗	✗

As the development of the demonstrator will be performed by myself and due to my knowledge of the **Artisan Studio tool**, and the ease of programming in the VB language, this tool appears to be that best- adapted for the SysML modeling.

2. 3D CAD tool

The selection of the 3D CAD tool is central, as this tool will be the main support for the user interface. Thus it must also meet the "GUI (ergonomics)" requirement.

The requirements related to the 3D CAD tool are described in Figure 67, with a red contour used to highlight those used as selection criteria.

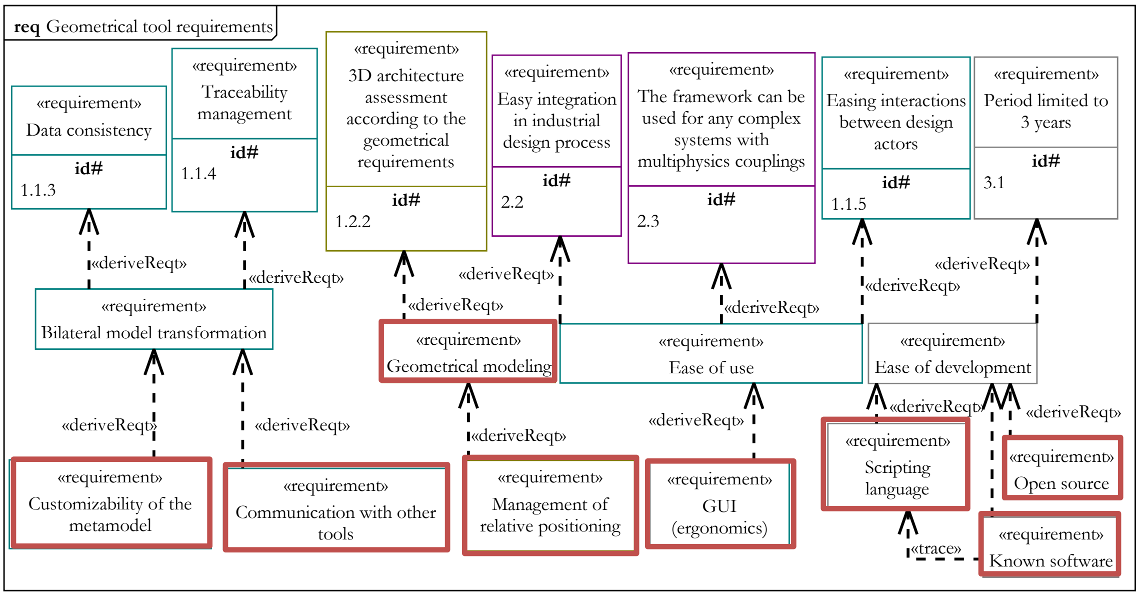


Figure 67: Requirements related to the 3D CAD tool.

Five 3D CAD tools are benchmarked according to these criteria in Table 40.

Table 40: Benchmark of main 3D CAD tools.

Criteria / Tool Name	Customizability of its metamodel	Communication with other tools	Geometry modeling	Management of relative positioning	Scripting language (*good knowledge by the developer*)	GUI (ergonomics)	Open source
Catia V5 R21 (Dassault systemes, 2017)	✗	COM interface	B-Rep / CSG	✓	VBScript (✓)	Toolbar ✓	✗
Siemens NX 8.5 (Siemens, 2017)	✗	COM interface	B-Rep	✓	VB.Net (✓)	Toolbar ✓	✗
SolidWorks 2014 SP0 (Dassault systems, 2017)	✗	COM interface	CSG	✓	VB.Net (✓)	Toolbar ✓	✗
FreeCAD V0.15 R4227 (FreeCAD, 2017)	✓	COM interface via python library	CSG	✗	Python (✓)	Toolbar ✓	✓
OpenCascade 6.4.0 (OpenCascade, 2017)	✓	COM interface via python library	B-Rep	✗	Python (✓)	Functions available via scripts ✗	✓

An important requirement addresses the customizability of the metamodel. Usual commercial tools do not provide this feature. For this reason, FreeCAD and OpenCascade appear to be the most adapted tools. Another important requirement is that the tool implementing the demonstrator must be user-intuitive with GUI toolbars. Consequently, the FreeCAD tool fits the most requirements, although it does not yet manage the relative positioning between geometrical elements.

3. Simulation language and tool

The choice of the simulation language and tool is based on different requirements, since the geometrical and system modeling data do not need to be added in their metamodel: only parameters and the simulation structure are transmitted to them.

Therefore, the metamodel of the simulation language does not need to be modified. The relevant data will only be "projected" to other tools (SysML and 3D CAD), as in the case of model federation. Considering this approach, the requirements for the simulation language are given in Figure 68.

The additional requirements are described in the Table 41.

Table 41: Additional requirements related to the simulation language

#ID	Name	Txt
1.2.12	Thermal analytic modeling based simulation	Chapter 4 proposed an approach based on thermal analytic modeling. Thus the language which will support the simulation must solve thermal analytic equations.
1.1.12	Acausal modeling	Acausal or non-oriented modeling includes a set of equations which are symbolically solved to obtain a computable sequence before being solved numerically. The simulation model must be acausal, since it allows a high level of abstraction for writing equations.

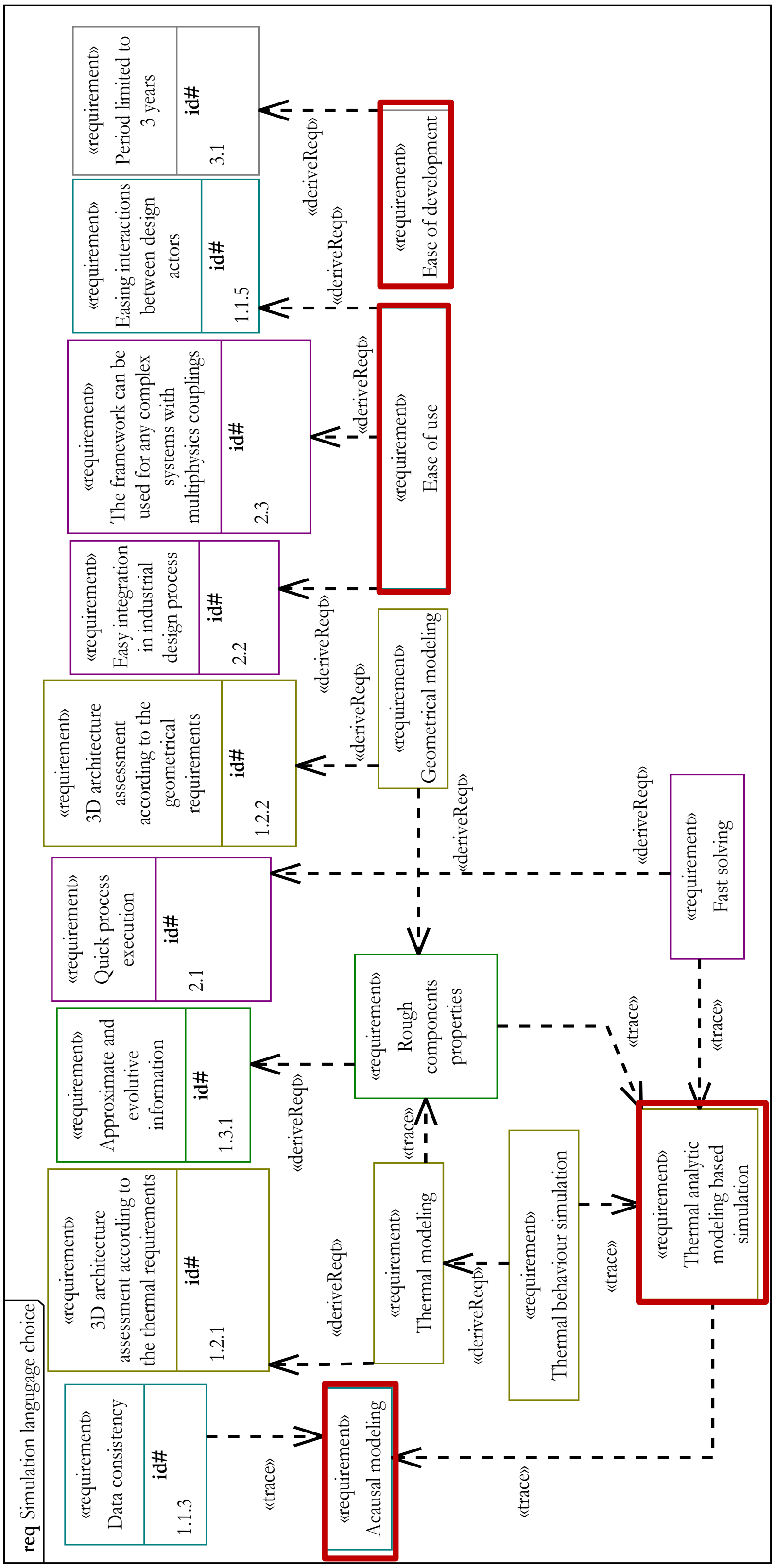


Figure 68: Simulation language related requirements

Regarding SysML, the suitability of the Modelica language regarding these requirements is checked in the following table (Table 42).

Table 42: Validation of Modelica language adequacy.

Name	ID	Modelica
Thermal analytic modeling based simulation	1.2.12	Modelica solves analytic equations.
Acausal modeling	1.1.12	The Modelica language is acausal.
Fast simulation	2.4	Modelica equations are resolved quickly, although this can take longer than other causal competitor languages, due to the time required by the symbolic analysis.
Ease of development	3.2	Thanks to acausal calculations, which allows a high level of abstraction, Modelica modeling is easy to perform.
Ease of use	1.1.8	Modelica can be used with graphical objects, such as SysML, ensuring easy use of this language.

Since the Modelica language can be supported by various tools, the criteria chosen to compare them are highlighted by a red contour in the requirement diagram in Figure 69.

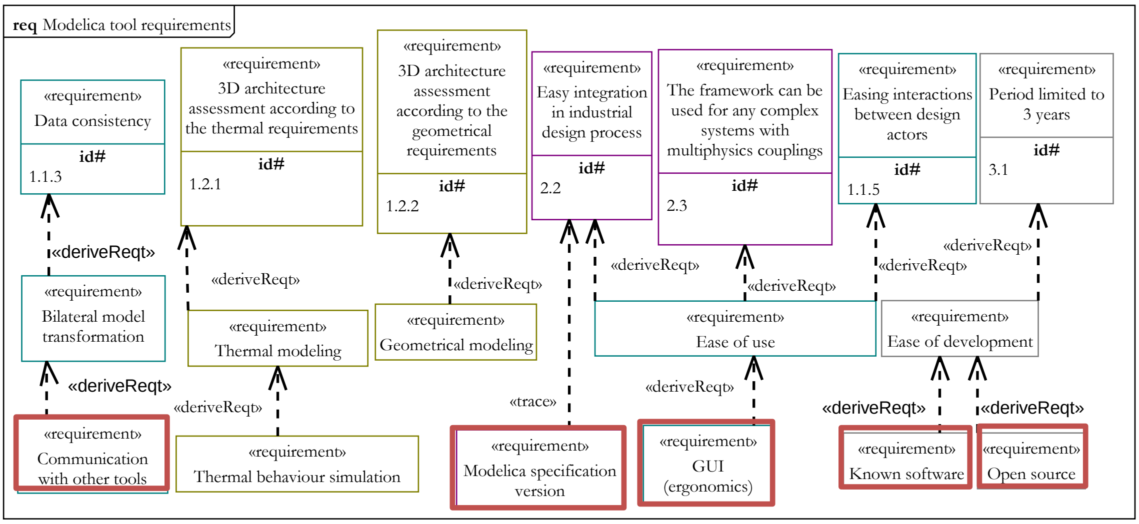


Figure 69: Modelica tool selection related requirement

Benchmarking has been established with different tools supporting the Modelica language and is detailed in Table 43 .

Table 43: Comparison of several usual Modelica tools.

Name	Modelica specification version	GUI (ergonomics)	Communication with other tools	Good knowledge of software	Open source
Dymola 2015 (Dassault systemes, 2017)	3.3 R1	✓	✗	✓	✗
Scicos 4.4.1 (Scicos, 2016)	2.0	✓	Scicos library	✗	✓
Open Modelica 1.9.5 (OpenModelica, 2017)	3.2	✓	OMPython library	✗	✓
JModelica 1.17 (JModelica, 2015)	3.1	✗	JModelica library	✗	✓
MapleSim 2015 (Maplesoft, 2017)	3.2	✓	✗	✗	✗

The previous table shows that only two tools meet all the requirements except good knowledge of the software OpenModelica and Scicos, since data exchange with other tools is available through a library, and their GUI allows visualizing the model generated. Finally, as OpenModelica has the advantage of supporting a more recent version of Modelica than Scicos, it will be selected as the Modelica tool implemented in the demonstrator.

4. Validation of data interactions management for the tools selected

Based on the previous paragraphs, in which comparisons and analyses of languages and tools were presented regarding the requirements for implementing the thermal 3D sketcher (Figure 70), the selected set consists of:

- the SysML language that will be supported by the Artisan Studio v7.4 (ATEGO) software, and then extended to the Integrity Modeler v8.2 (PTC) software (PTC, 2017);
- the 3D CAD modeling will be supported by the FreeCAD v. 0.15.4527 (FreeCAD, 2017) freeware;
- the Modelica language will be supported by the Open Modelica v. 1.9.1 r19512 (OpenModelica, 2017) freeware.

As these languages must be correctly interfaced, the choice of these tools will be validated only if these tools can be connected.

The Python interpreter contained in FreeCAD will play the role of transformation platform.

Concerning the connection between PTC and FreeCAD software, the Python library "Pywin32" allows connecting a python script with a COM interface. Thus connection between Artisan Studio and FreeCAD can be done using the Pywin32 library.

Regarding the connection between FreeCAD and OpenModelica, the "OMPython" Python library ensures the connection between Python scripts from FreeCAD and OpenModelica. This library can be used to launch simulations and read results. However, OMPython cannot be used to automatically generate a Modelica file, thus it must be generated manually.

Finally, the interface between these different software tools for the different model transformations will be ensured by the Python language and FreeCAD will play the role of coordinator, as described in the following paragraph.

Furthermore, as FreeCAD software is an opensource software application, it can be compiled with additional libraries, such as Pywin32 (to provide the interface between PTC software, Python language, and OMPython). The compiled version of FreeCAD is therefore portable and a batch script has been implemented in order to install the compiled FreeeCAD model to facilitate the further reutilization of the demonstrator.

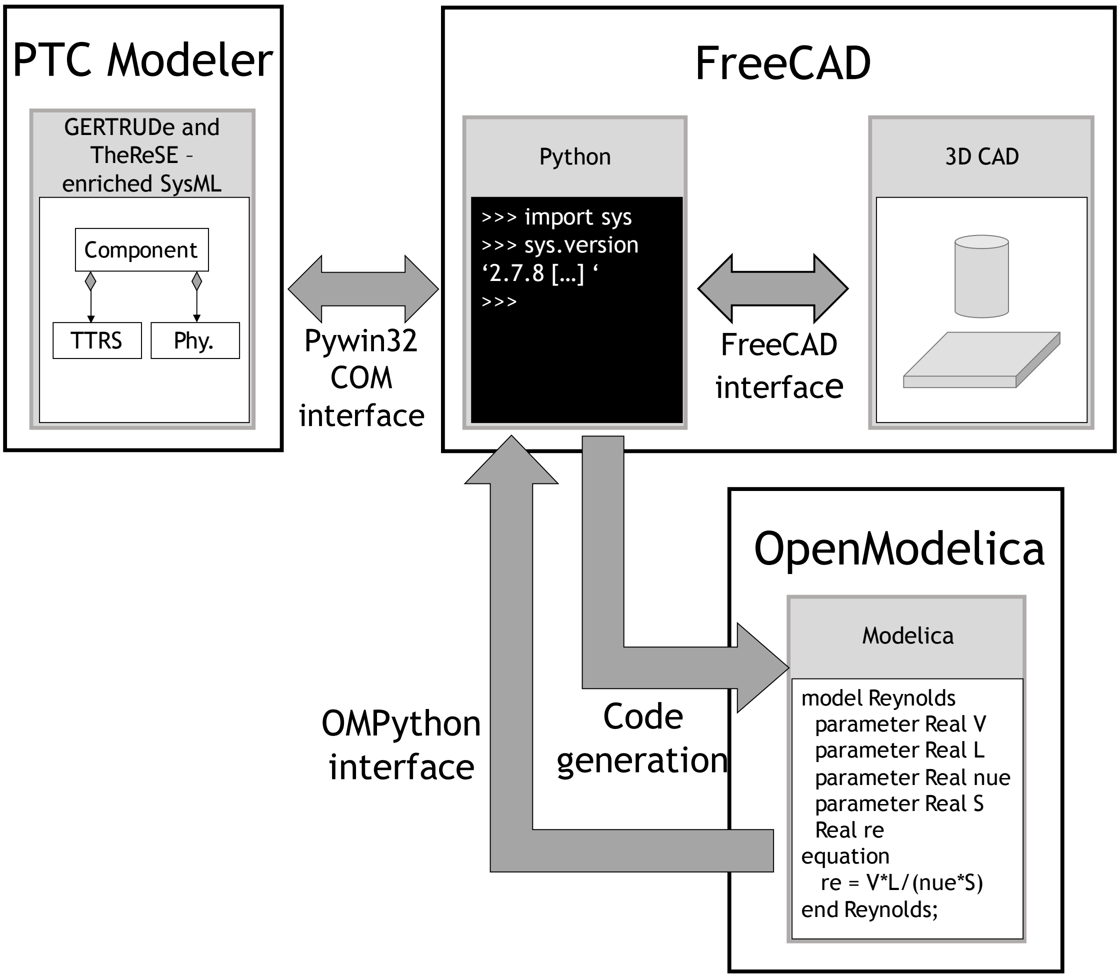


Figure 70: The implemented SAMOS framework with Software and their related mapping links.

For each view, an existing tool has been selected and required specific developments, in order to manage the model transformations (Chapter 1 II.C.4.b) between each view.

II. Algorithmic modeling of the Thermal 3D Sketcher

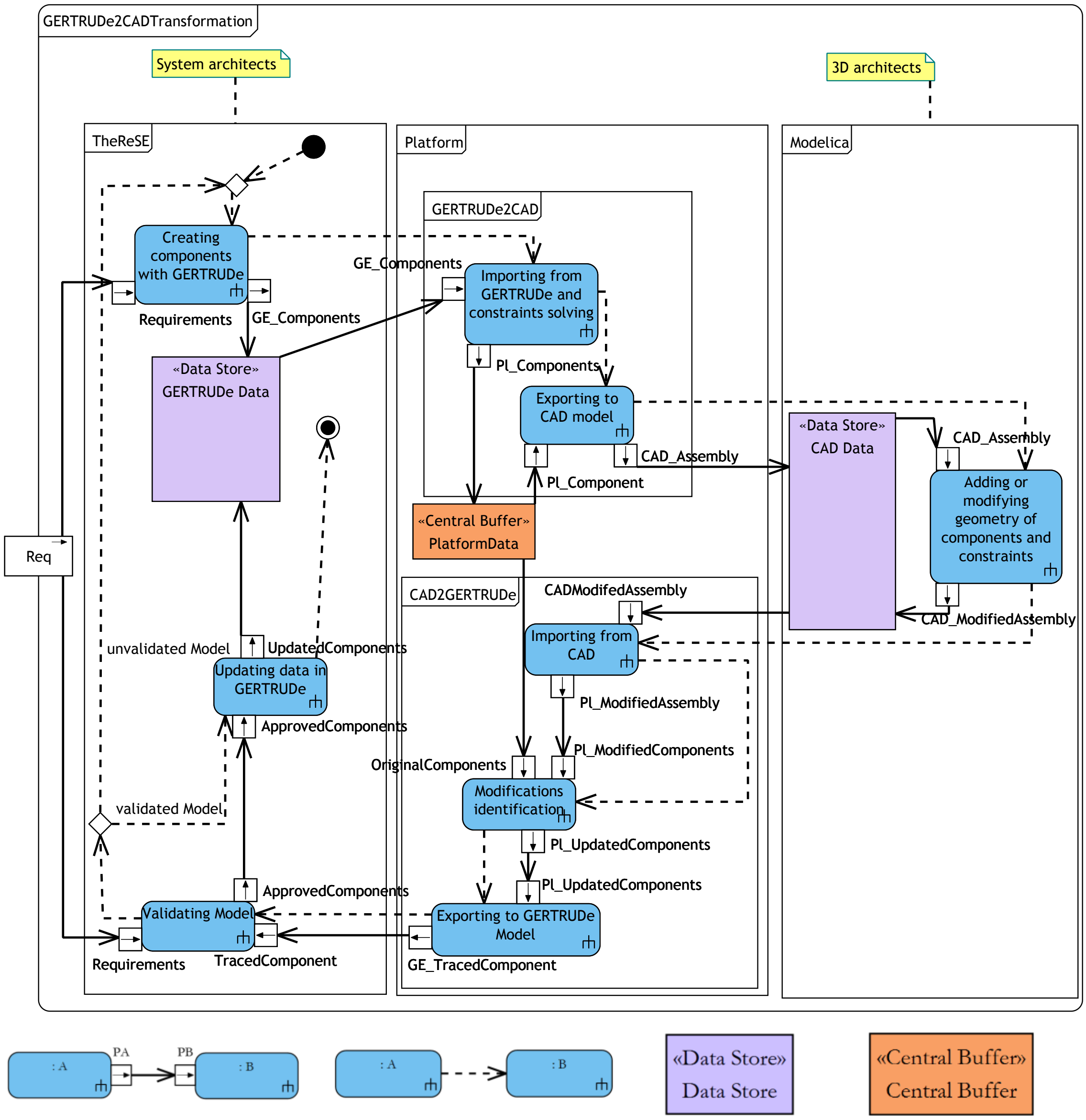
A. Description of the geometrical model transformation

Concerning the geometrical model transformation, we consider only the exchange from SysML to 3D CAD tools that contain the intrinsic parameters and enriched-TTRS related data. The face-related data are not described in the model transformation process.

1. Geometrical modeling transformation process

The detailed description of the model transformation process from GERTRUDe to 3D CAD is presented in Figure 71 by an Activity diagram. The actors of the activities have been added using yellow notes. The black spot and the circled black spot signify the beginning and the end of this activity process, respectively.

First of all, the System Architects create all the components of the physical architecture in the SysML model and enrich them by assigning geometry with GERTRUDe to those whose geometrical features need to be specified for the 3D Architects. Then, these data are stored in the *GERTRUDe data store*. GERTRUDe data are then imported to the Python Platform, to be stored in Python in the *Central Buffer*. Next, the transformation of these GERTRUDe data in Python to 3D CAD data can be launched and performed. Once the 3D components have been generated in the FreeCAD environment, the 3D Architects can then add or modify the geometry of the components, and some of the geometrical constraints between them. When finished, these data are stored in the *CAD data store*. 3D CAD data are then imported to the Python Platform. Next, a comparison with the initial GERTRUDe data is performed through the "*modification identification*" activity to identify modified and new data generated by the 3D Architects. The identified "*updated*" data are then transformed into the GERTRUDe Model. The System Architects will subsequently validate or invalidate the updated elements proposed. If the architecture model is validated, the GERTRUDe model is then updated and the activity process is finished; otherwise, if the model is not validated, the System Architects will go back to the first activity and modify certain component specifications or other geometrical requirements with GERTRUDe to formalize the reason why the previous proposed architecture was not validated.



Object Flow

Object flow is a path along which objects or data can pass.

Control Flow

A control flow shows the flow of control from action A to action B.

Data Store

A Datastore defines permanently stored data.

Central Buffer node

A Central Buffer Node is an object node for managing flows from multiple sources and destinations.

Figure 71: Detailed model transformation process presented in an activity diagram.

2. Transformation Metamodel

The previous activity diagram shows the different steps involved in transmitting the geometrical-enriched physical architecture between the different data stores and the central buffer. The representation of the architecture is not the same in these different stores. The metamodels of these different stores and the links between these different architecture views are provided by the class diagram representing the transformation metamodel (Figure 72).

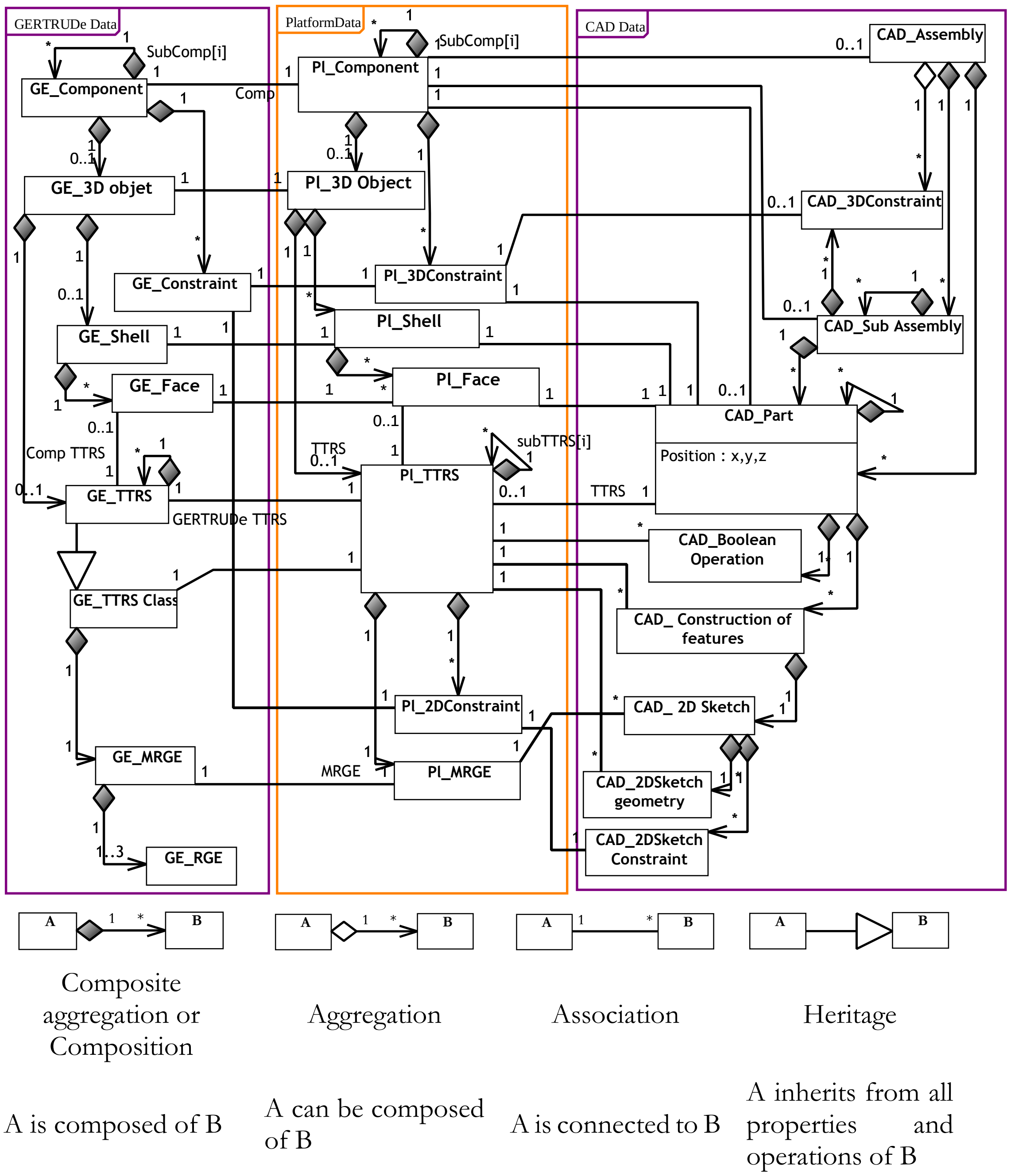


Figure 72: Model Transformation Metamodel using class diagram

The GERTRUDe metamodel was detailed previously in chapter 3 section IV , by identifying its different classes by the prefix *GE_*.

The FreeCAD metamodel:

- The highest class is that of "*CAD_Assembly*". A *CAD_Assembly* can be composed of *CAD_SubAssembly* elements or not.
- *CAD_Part* elements composing *CAD_Assembly* and *CAD_SubAssembly* elements contain geometry data relating to their construction process.
- *3D constraint* is used to position two *CAD_Part* elements or *CAD_Sub-Assembly* elements and belongs to the parent element, *CAD_SubAssembly* and *CAD_Assembly*. Some instances of this class can be, for example, contact constraint or coincident constraint. As not all 3D CAD tools, including the

FreeCAD tool, allow the positioning of parts using constraints, an aggregation link is used between *CAD_Assembly* and *CAD_Constraint*. A *CAD_SubAssembly* element can be composed of different *CAD_SubAssembly* and *CAD_Part* elements.

- A *CAD_Part* element can be composed of other *CAD_Part* elements, and some *CAD_Boolean Operation* elements like intersection and subtraction if it is built from other (sub)parts.
- A *Part* is also composed of *CAD_Construction of features* elements, which correspond to the method used to build 3D geometry from 2D sketches which can be Extrusion, Revolution, Multi-section, and Spine scanning.
- CAD Construction of features contains the CAD_2D Sketch on which it is based.
- Finally, this *CAD_2D Sketch* is composed of *CAD_2DSketch Geometry* elements which can be points, lines, circles, arcs and ellipses.

The Platform metamodel:

To manage the transformation, the Platform metamodel is composed of the class

- *Pl_Component*, as in the GERTRUDe metamodel. A component can be composed of different *Pl_Components*. This *Pl_Component* is connected to *GE_Component, CAD_Assembly*, *CAD_SubAssembly* and *CAD_Part*. Regarding simplification, the GERTRUDe metamodel does not distinguish subassembly, assembly and part, which are all defined as *GE_components*. *Pl_Component* is linked to one GERTRUDe class and three 3D CAD classes.
- *Pl_Component* contains *Pl_3D object* in order to manage the geometrical data of the *Pl_Component*.
- *Pl_TTRS constraint* elements allow positioning Components in relation to each other. *Pl_TTRS constraint* is not included in *Pl_3D object* because the Constraint belongs the Assembly. *Pl_Constraint* is associated with *GE_TTRSconstraint*, *CAD_3DConstraint*, and finally *CAD_Part,* since in the case where the 3D CAD tool does not contain constraint management, the position(x,y,z) of parts must be solved in the Platform.
- *Pl_3D object* is composed of *Pl_TTRS* to generate geometry. *Pl_TTRS* element contains its *Pl_MRGE* element and is composed of zero or more (sub)TTRS and their corresponding *Pl_constraints* to position them in relation to each other.
- *Pl_3D object* is also composed of *Pl_Shell* that manage only geometry data without Topology, which is necessary for thermal modeling. *Pl_Shell* is associated with *GE_Shell* and *CAD_Part.*
- *Pl_Shell* is composed of *Pl_Face*. Which is associated with GE_Shell and *CAD_Part*. *Pl_Face* is associated with *GE_Face* and *CAD_Part.*
- *Pl_TTRS Class* is then connected to the *GE_TTRS* and *GE_TTRSClass* classes, in order to rebuild TTRS modeling from a 3D CAD geometry, and to *CAD_Part* and *CAD_Boolean Operations* classes, in order to perform the intersection operations of all the new 3D CAD parts which have been generated by the transformation process from TTRS elements.

3. CAD2GERTRUDe transformation

The CAD2GERTRUDe transformation depends on the geometrical modeling of the 3D CAD tool. As FreeCAD is based on the CSG modeling procedure, the model transformation proposed is adapted for this purpose.

Three kinds of 3D CAD data traceability processes (into SysML) are available:

- when there is no modification of the geometry. Thus, only face data must be traced to GERTRUDe;
- when the modification only concerns dimensional parameters. Thus Face data are added, and dimensional parameters are modified;
- when the geometrical shape of a *CAD_Part* is modified. Thus the TTRS modeling in SysML has to be reconstructed.

When comparing these updated 3D CAD data with the initial data provided by the SysML into FreeCAD model transformation, the value properties of the related *GE_Components* have to be updated if there is no modification or if only dimensional parameters have changed. If the shape has changed, it is necessary to proceed to the identification of the new TTRS of the updated *CAD_Parts*. To do this, the *CAD_Part* tree is browsed by starting from the corresponding *CAD_Root Parts*, which could have been built using *Boolean operations* between other *CAD_Sub-Parts* (Figure 73). As *Boolean operations* do not have any equivalence class in the GERTRUDe extension, each operation, whether an addition or a subtraction, must be analyzed in order to correctly orientate the outgoing vector of each generated *GE_TTRS* (Figure 74). This process is performed successively for each *CAD_Sub-Part* until an elementary *GE_TTRS* level existing in the GERTRUDe extension is reached.

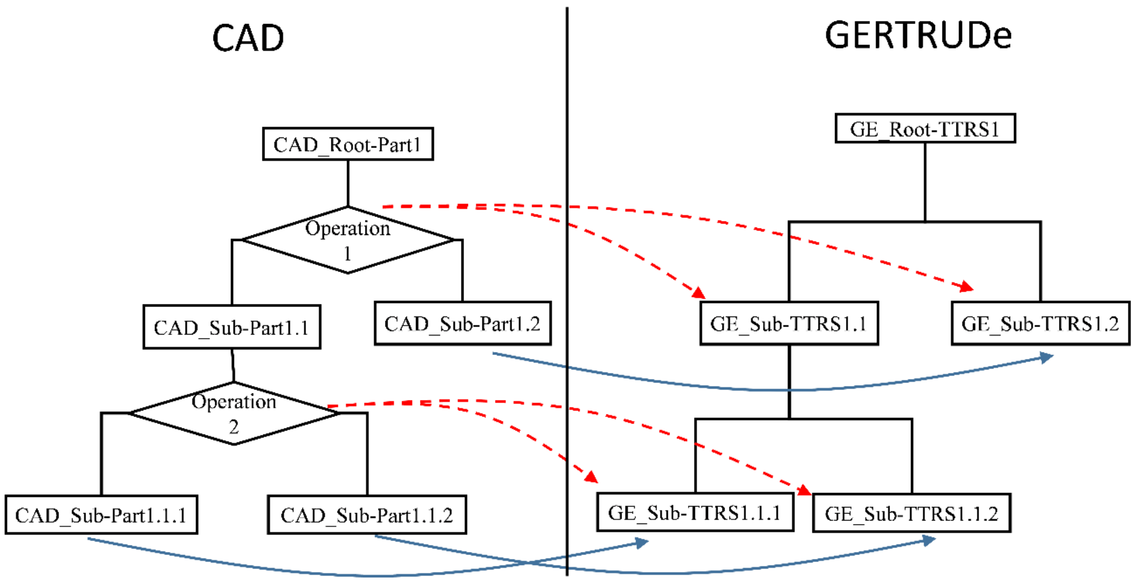


Figure 73: Transformation of a 3D component into GERTRUDe.

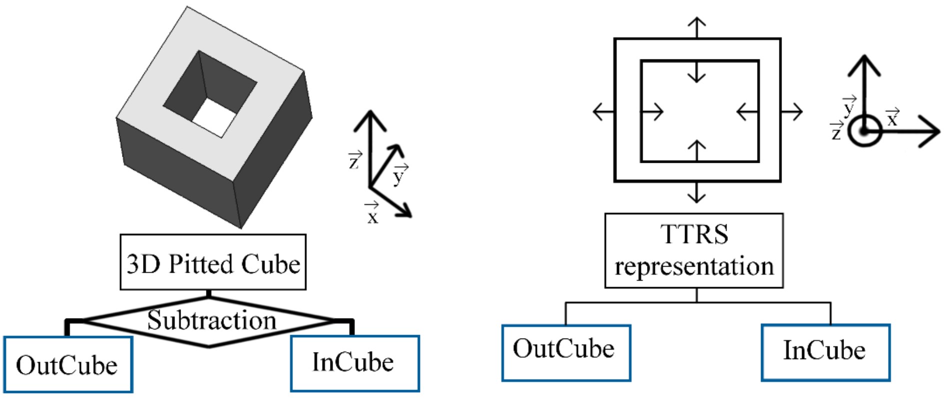


Figure 74: 3D CAD into TTRS representation transformation

4. GERTRUDe2CAD transformation

a. Import from GERTRUDe

Before describing the "*Transform GERTRUDe to CAD model*" activity, we will describe the *"Importing from GE"* activity. In this activity, an importation of the metamodel from GERTRUDe is imported to the Python language. The interface between the SysML Model and the transformation (Python) platform is provided using the distributed COM interface (Microsoft, 2015). For this reason, the Python language has to be enriched with the PyWin32 library (Hammond, 2015), in order to provide the COM Client. The COM Server is ensured by PTC

The compilation process and the TTRS equation solving is described in the following stages Figure 75:

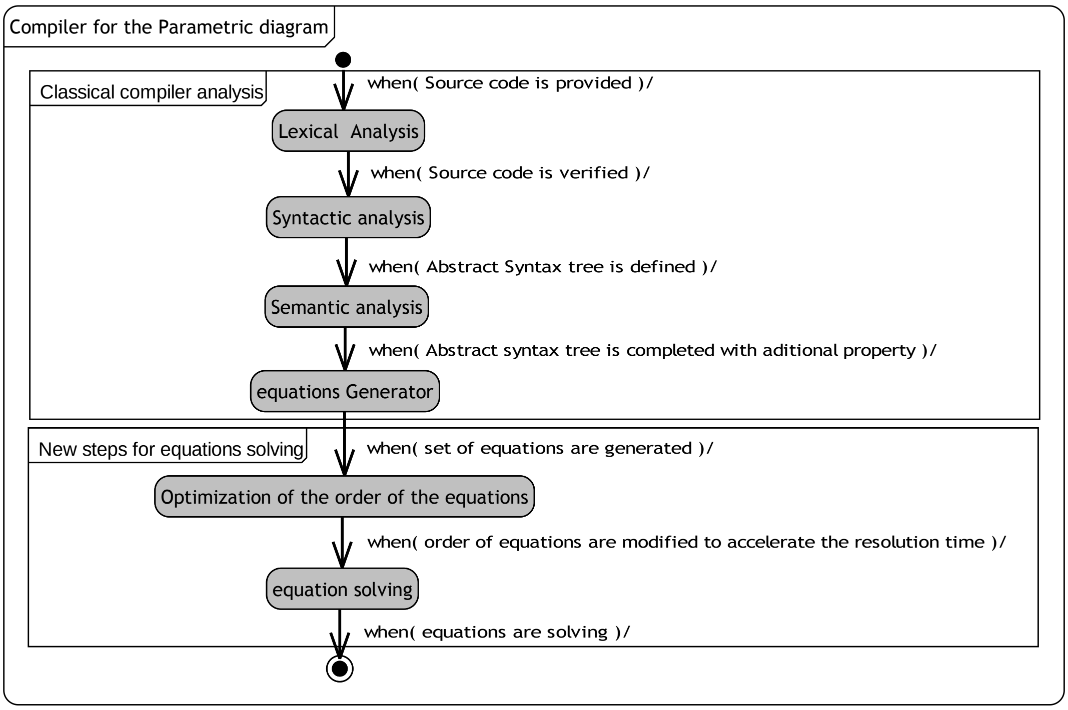


Figure 75: Compilation process.

The **lexical analysis** developed (called lexer) for equation solving is described by the following state machine diagram (Figure 76). In this diagram, each transition corresponds to the changeover to the next character of the textual expression ("string") of the equation, provided that the transition condition fits the corresponding character recognition. When this condition is not fulfilled, the process is broken by a transition (not graphically represented for the sake of readability) to the final *Exit with error* state. When all the characters of the string expression have been browsed successfully, the state switches to the *Exit without Error* state. Moreover, the parentheses are counted through a variable (parentheses counter *pc*) to verify their symmetry (each open parenthesis increments this variable, and when a parenthesis is closed, a decrement is applied to the variable). The variable must be equal to 0 at the end of the expression. This diagram is not exhaustive, as it does not integrate vector operators like the cross and scalar products. Nevertheless it fits our needs according to the equation modeling method selected. Indeed, this lexer deals with sinus and cosine operators (*sin* and *cos*), in accordance with the expressions of TTRS constraints chosen.

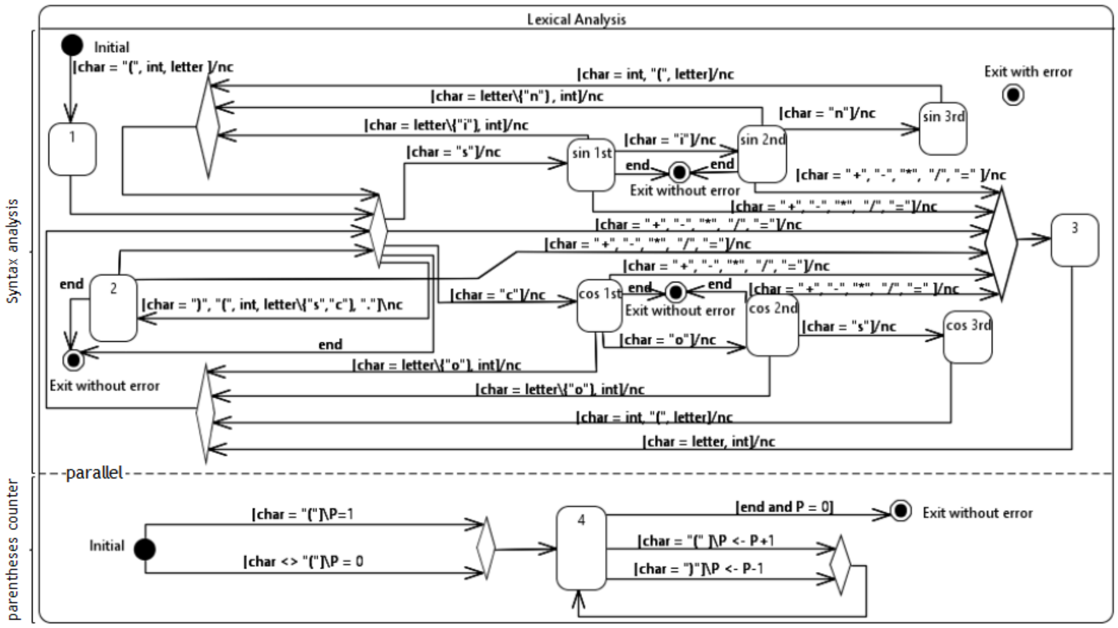


Figure 76: State machine diagram of the lexical analysis stage.

The previous **syntactic and semantic analyses** are based on tree building detailed in

The **equation generator** used simply relates to the formal reconstruction of the mathematical expression from the previous semantic tree.

Concerning the **optimizer** stage, it addresses the optimization of the order of equations to be processed. When a set of equations is generated, the optimizer developed orders the different equations according to the value of their complexity metrics. This metric has been built based on the number of unknowns, and the degree of complexity of the mathematical functions used. For example, a sum operator is simpler to calculate than a cosine function. Then the more complex equation will be placed at the end of the set, in order to facilitate the resolution of other variables in the previous equations.

Expression of Complexity Metrics

For each equation, a complexity metric "m" is calculated, using the following formula:

$$m = n_{var} \cdot \sum_{i=1}^{n_{op}} op_{W_i}$$

Where:

- m is the metric of the equation
- n_{var} is the number of variables
- n_{op} is the number of operators/functions
- op_{W_i} is the complexity weight of the operator/function i ($i \in [\![1, n_{op}]\!]$)

The complexity weight of the operators and functions (used for TTRS constraints equations) has been defined experimentally and is given in Table 44:

Table 44: Complexity weight attributed to each operation or function occurring in the equations to be solved.

Operator/function	Complexity weight
Addition	1
Subtraction	1
Multiplication	2
Division	2
Sinus	5
Cosine	5

This "optimization stage" permits reducing the resolution time by 3.

Once the equations set has been ordered, the **equations solving** step is performed with the Sympy Python library.

b. Export to CAD

Finally, the next activity "*Transform GERTRUDe2CAD model*" is launched. This activity is described using a sequence diagram (Figure 77).

On this sequence diagram the interaction between the *Central Buffer Platform Data*, delimited by an orange framed box is considered as the system to be designed, and the *CAD data* data store is delimited by a purple framed box.

The "Actor elements" of the *CAD data* on the left side and of the *Central Buffer Platform Data* on the right side, are those of the CAD and the Platform metamodels, respectively, presented previously in Figure 72. This sequence diagram is recursive since the tree of the components created with GERTRUDe is browsed to visualize the components and subcomponents that will then be associated with *CAD_Assembly*, *CAD_SubAssembly* and *CAD_Parts*.

Since a component can be a *CAD_Assembly*, a *CAD_subAssembly*, or a *CAD_Part,* it is important to first distinguish these different component levels. Therefore, if the component considered has subcomponents, this means that it is an assembly or a sub-assembly. Then, if it is the main component, it must be associated with an *Assembly* element; otherwise, if it has a root component, it must be associated with a *SubAssembly* element. Finally, if it has no subcomponent, it must be associated with a *Part* element.

Next, if the component has a defined geometry i.e. it is associated with a TTRS, the geometry is exported to a CAD Part using the TTRS based-construction method explained previously (Figure 77); otherwise, if the component considered has no geometry, i.e. any TTRS to build the part, a "void" part is created with no geometrical specifications and assigned the name of the corresponding SysML component.

Then, the function *TransformationAndExport2CAD (i)* is called for each subcomponent *i*. When finished, if the 3D CAD tool has no constraint solver (which is the case of the FreeCAD tool), the TTRS constraints specified by System Architects are solved by the transformation platform, and the components are then positioned.

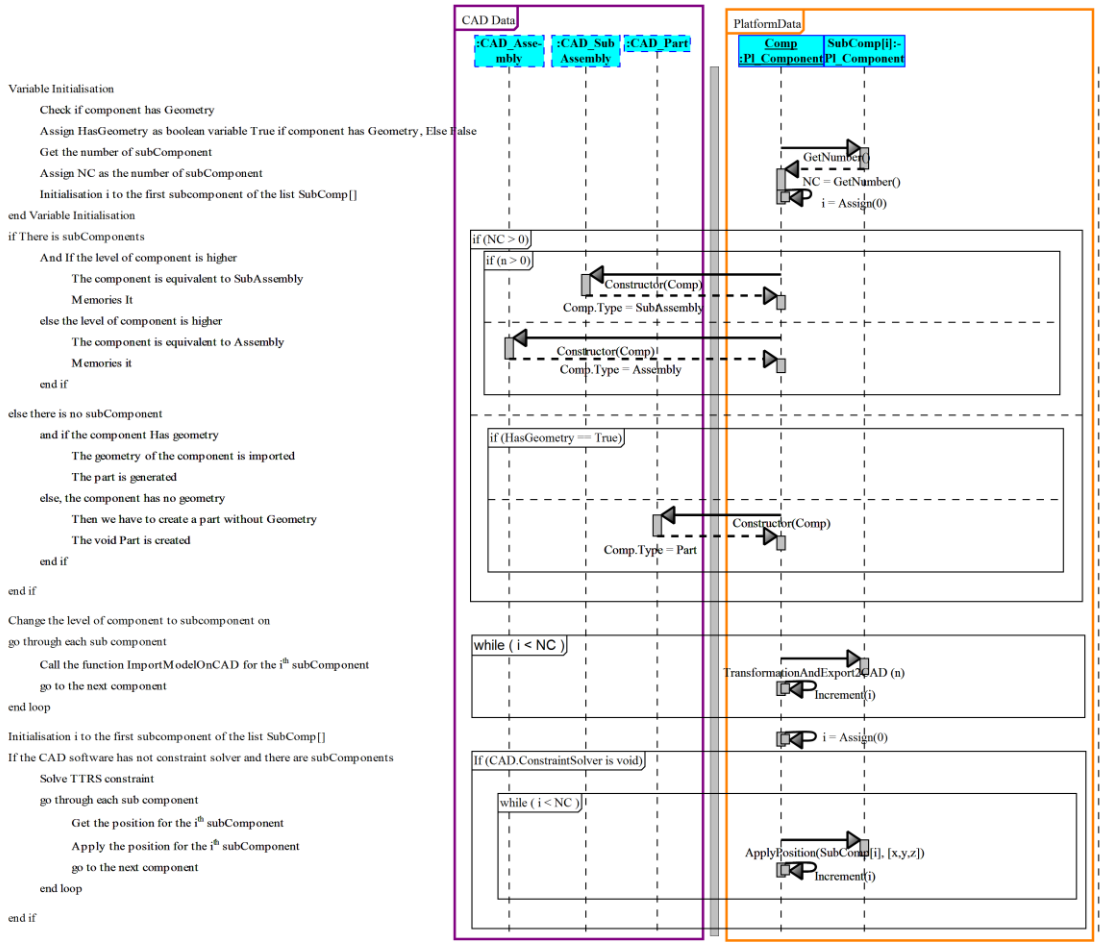


Figure 77: Detailed description of the "Transform GERTRUDe2CAD model" activity, using sequence diagram.

B. *Descriptions of the thermal model transformations and algorithms*

1. Thermal modeling transformation process

The detailed description of the model transformation TheReSE - Modelica process is presented in Figure 78 by an Activity diagram. As described for the GERTRUDe – CAD model transformation, the actors for each activity have been added using yellow notes. The black spot and the circled black spot respectively signify the beginning and the end of this activity process. Since TheReSE contains GERTRUDe, all the previous steps described with the GERTRUDe analysis have to be completed while this process is being performed. For example, the activity *Importing from TheReSE* also contains the activity *Importing from GERTRUDe*

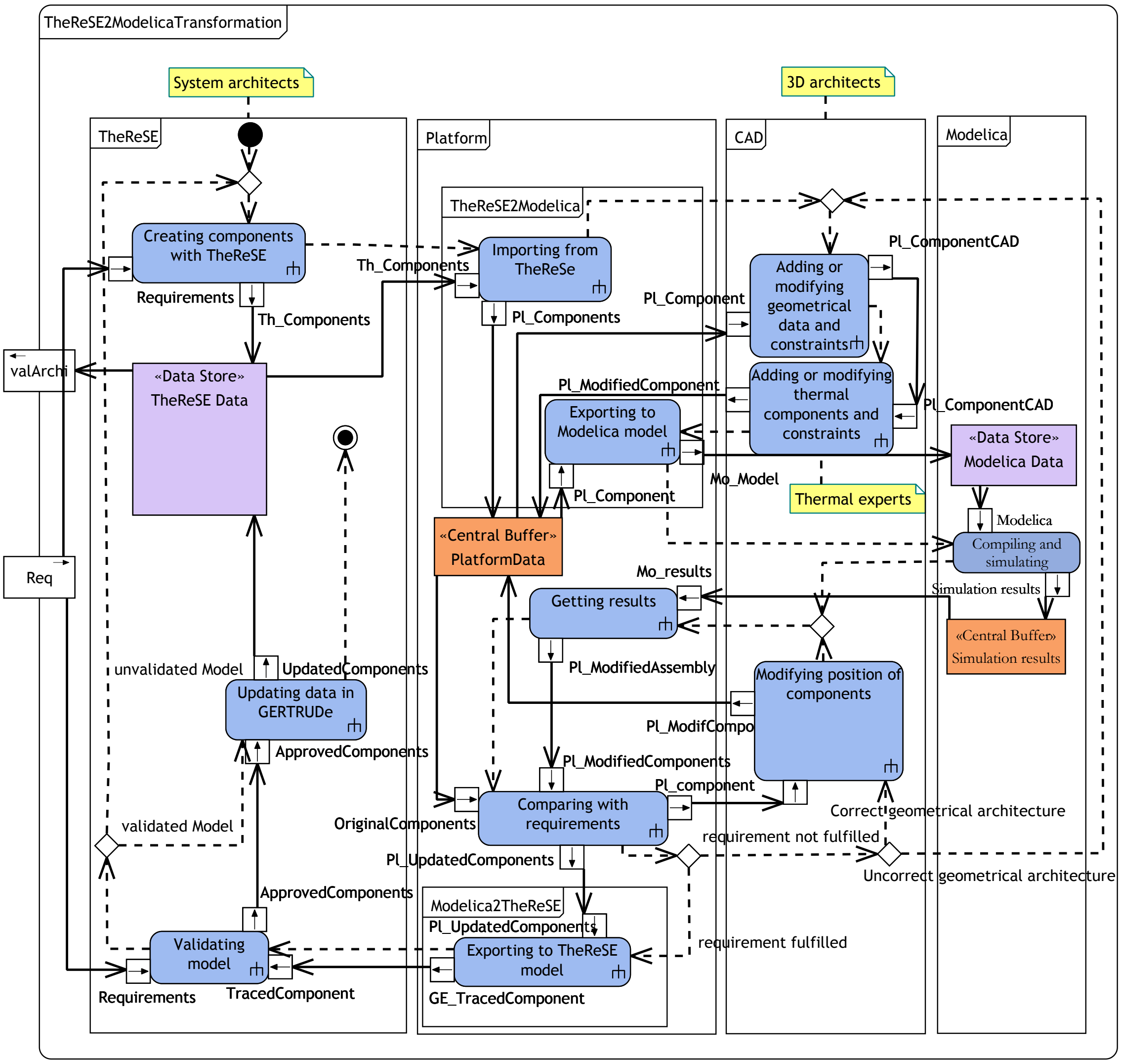


Figure 78: Detailed model transformation process represented in an activity diagram.

First, the activities described are the same as in the previous model transformation process, except that the System Architects create thermal components by using TheReSE. Once the System Architects have finished the thermal modeling (components and thermal specifications), the model is then stored in a TheReSE data store. Regarding the geometrical modeling process, the Python platform reads the thermal architecture and writes it in the

Python central data buffer. Next, the 3D architects enrich the 3D architecture and position each element in relation each other. Finally, the thermal experts add or modify certain thermal components and constraints using the 3D CAD tools (the GUI of the FreeCAD tool has been enhanced to meet this need). Then, the thermal specifications are stored in the Python central data buffer. Once these specifications have been stored, a Modelica file is generated from these specifications and the Modelica code is compiled thanks to the OMPython library. Thus, when the 3D architects move components under geometrical constraints, the positioning parameters (for example, the relative distance between two components) are then written in OMPython as parameters.

Therefore, if no change has occurred in the topological model, the Modelica file does not need to be compiled and the simulation becomes faster.

If a change has occurred in the topology of the system model (e.g. the thermal experts have modified certain thermal constraints), a new simulation loop is required.

The results are given by OMPython, and a Python script compares the simulation results with the requirements. When the requirements have been fulfilled, the corresponding simulation results are traced back in the TheReSE model.

2. Model transformation metamodel

Regarding the geometrical data transformation, the following activity diagram shows the different steps involved in transmitting the geometrical/thermal enriched physical architecture between the different data stores and the central buffer. The representation of these architecture views is not the same in these different stores. The links between these different architecture views are provided in a class diagram representing the transformation metamodel in Figure 79 . This metamodel does not include the full representation of the data model of the Python, Modelica, and TheReSE data: only the information necessary for the model transformation from TheReSE to Modelica is provided. This metamodel must be completed using the GERTRUDe – 3D CAD model transformation metamodel (Figure 72).

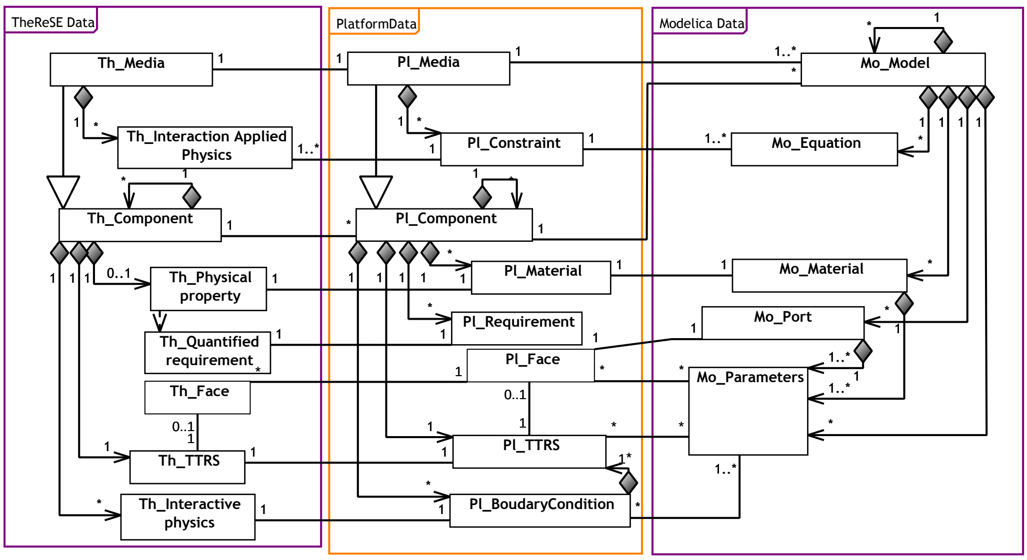


Figure 79: TheReSE - Modelica model transformation metamodel represented by a class diagram.

The model transformation metamodel TheReSE – 3D CAD is composed of 3 parts. On the left we can see the TheReSE metamodel extract, on the right the Modelica metamodel extract, and in the center the Python metamodel extract. We follow the same convention as for the geometrical modeling; the prefixes "Th_", "Pl_", and "Mo_" designate a class of TheReSE, Platform, and Modelica, respectively.

The TheReSE metamodel previously detailed in chapter 4 section VII is not be described again here.

The Modelica metamodel

Here, we only describe the elements used in the model transformation metamodel, since the modeling elements used from the existing Modelica library are listed further on in section V .

- First, the "*Mo_Model*" designates the Modelica model, where all the simulated elements are included. It can also contain specific behaviors, for example thermal radiation, or a fixed temperature.
- In order to model this behavior, "*Mo_Class*" can be composed of "*Mo_equation*", "*Mo_Material*", "*Mo_parameters*", "*Mo_Port*" which describe the input and output variables.
- "*Mo_parameters*" are the parameters that users can add.
- Finally, "*Mo_equation*" are the equations that combine the parameters, ports, and material parameters in order to provide the behavior of the class.

The Platform metamodel:

- The "*Mo_Model*" is associated with the "*Pl_Component*" which is also linked to the "*Th_Component*". The "*Pl_Component*" does not include any behavior; it mainly contains the boundary conditions.
- However, the "*Pl_Media*" also associated with a "*Mo_Class*" and with a "*Th_Media*" contains the thermal behavior. For example, the convection media contains partially solved equations of Laplacian functions (appendix 3)
- Then, this behavior equation "*Mo_Equation*" is related to "*Pl_Constraint*", while the equivalent class of the "*Pl_Constraint*" in the TheReSE metamodel is "*Th_Interaction applied physics*". These equations must have parameters and ports to be solved.
- "*Mo_Parameter*" can be generated by the "*Pl_TTRS*" provided that the parameters are geometrical, or also given by the "*Pl_Boundary Condition*" for thermal parameters.
- The "*Pl_TTRS*" is associated with the "*Th_TTRS*". It is also associated with "Pl_Face" which contains information on the face.
- The "*Pl_Boundary Condition*" is associated by the "*Th_Interactive Physics*" in the TheReSE metamodel. As described in (chapter 4 section VII), the thermal analysis deals with the heat crossing a face.
- Then, the "*Mo_Port*" is associated with a "*Pl_Face*" and with a "*Th_TheReSE*".
- Finally, the "*Mo_Material*" is associated with the "*Pl_Material*" which is linked to the "*Th_Physical Property*".
- The "*Pl_Requirement*" is not linked to the Modelica metamodel, because the comparison of the Modelica simulation results with requirements is not provided in the Modelica environment, but by Python scripts. Then, the "*Pl_Requirement*" is linked to the "*Th_Quantified_Requirement*".

3. Transformation Modelica2TheReSE and TheReSE2Modelica

After having detailed the model transformation metamodel given previously, the model transformation can now be described. Due to the time constraint, the **Modelica2TheReSE** and **TheReSE2Modelica** model transformations need only to be implemented on the basis of thermal models included in the Modelica standard library. The principle is that a Modelica model is associated with each heat mode, declared as a *Medium*. The details of these classes are given in the next paragraph. If we have had more time, we could extend this model transformation to many thermal models of other specific libraries, through a Python script, which would offer the possibility for the users to select the most suitable mode as a function of their objectives.

4. Volume slicing algorithm for convection modeling

This algorithm (Algorithm 1) splits the component geometry into different slices. Each floor corresponds to a constant interface between the solid and the fluid elements. These interfaces are automatically detected to generate fluid volumes with a constant section, and added in the second frame of Physical Tab (Figure 100). The volume slicing algorithm is detailed below, and the algorithms of the sub-programs are described in .

Algorithm Automatic component cutting

Input:
- Let *comp2Cut* the component that has to be cut

Output:
- Let *cutComps[ ]* be a list of cut components

```
BEGIN
    Initialize cutComps as a void list of components
    Let interComps the components inside comp2Cut
    interComps ← comp2Cut.sonComps
    Let shellParts be the list of exterior parts of component interComps
    Initialize shellParts as a void list of Parts
    if the material of comp2Cut is a fluid and comp2Cut.hasInlet and comp2Cut.hasOutlet, then
        Let faces be a void list of faces
        for each interComp in interComps, do
            shellPart ← call getShellGeom(interComp.Part)
            shellParts.add(shellPart)
        (inFace, outFace) ← call searchFaces(shellPart,Inlet,Outlet)
        faces ← call sortFaces(faces, inFace, outFace, Inlet, Outlet)
        for each cutPlane in faces, do
            (compCut, comp2Cut) ← call cutComp(comp2Cut, cutPlane, Inlet, Outlet)
            cutComps.add(compCut)
        return cutComps
    Else
        call displayErrorMessage()
END
```

Algorithm 1: Automatic component cutting.

Where:

- *cutComps* is a list of fluid component cuts.

- *comp2Cut* is the fluid component in one part to be cut
  - .*sonComps* is a list of solid components inside the Fluid
  - .hasInlet is a Boolean which has the value "True" if the fluid is considered as inlet, "False" otherwise
  - .hasOutlet is a Boolean which has the value "True" if the fluid is considered as outlet, "False" otherwise
- *interComps* is a list of solid components inside the fluid
  - .*Part* is a parameter that corresponds to the 3D CAD Part of the component
- *shellParts* is the external face of the Part
- *faces* are the finite faces of the Part
- getShellGeom(Part) is a subfunction which gives the external face of the Part
- searchFaces(ShellPart, FaceI, FaceO) is a subfunction which gives the two faces (IFace, OFace) that cut the fluid component
- sortFaces(Face[ ], IFace, OFace, FaceI, FaceO) is a subfunction which adds IFace and OFace in the Face list [ ] which is the list containing the faces that cut the fluid element, ordered such that the distance between face[i] and faces[i+1] is minimal.
- cutComp(Component, CutPlane, FaceI, FaceO) cuts the component into two components according to the plane cut.

III. Developments in SysML (Atego-PTC environment)

The first development addresses the SysML environment with the PTC-environment.

A. GERTRUDe

1. Profile diagram

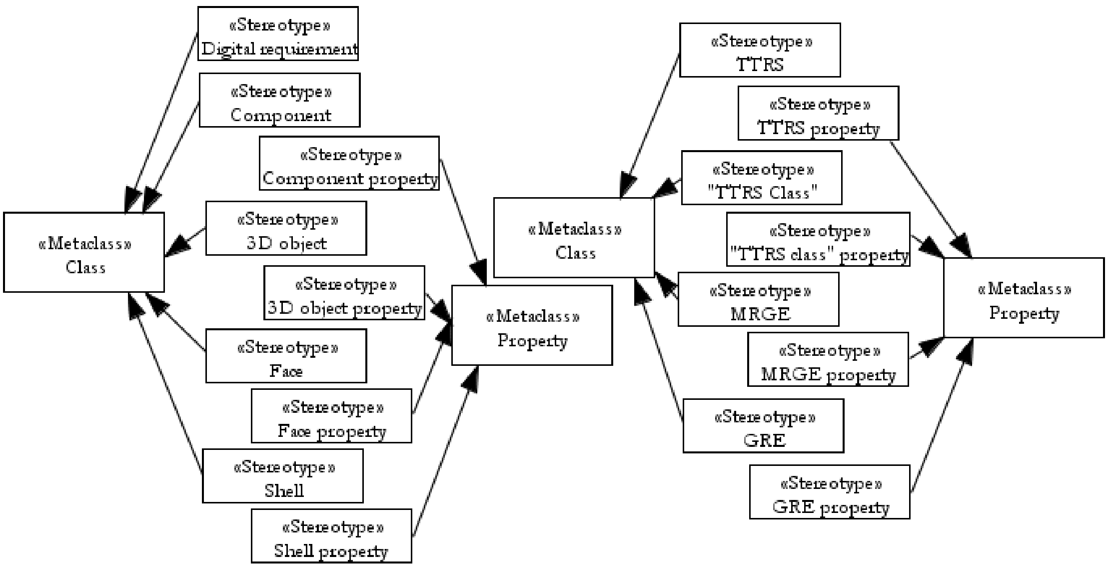


Figure 80: Profile diagram of GERTRUDe

A description of the metamodel developed in SysML is provided in chapter 3 IV . The corresponding profile diagram for the implementation of the GERTRUDe metamodel is given in Figure 40. This profile diagram does not show the new "custom diagrams" (cf the following paragraph), since the UML specification does not allow it. However, these new custom diagrams could have been implemented in the PTC integrity modeler environment, as presented below.

a. New custom diagram

As SysML is a graphical language, it is important to explain the GUI elements associated with the new diagrams which will impact the user's GUI. These diagrams are UML class diagrams, or composite structure diagrams, with customization based on scripts and a toolbar provided by tools, since it can be defined by a standard stereotype.

In the GERTRUDe extension, two diagrams have been added: the CLD AssemblyDiagram based on a UML class diagram, and the STRD AssemblyParam built from a UML composite structure diagram.

CLD AssemblyDiagram

The *AssemblyDiagram* describes the geometrical assembly of the system elements. It manages the creation of geometrical elements and the different links related to elements. This diagram can be used by both the design actors and by the 3D CAD developers (to enrich existing libraries).

- Design actors usage:

First, the *AssemblyDiagram* allows defining the composition of subcomponents in a geometrically enriched physical architecture, through a composition link (Figure 81). These (sub)components can be added from the *Geometry library* developed (see Chapter 6 section I.B.1).

It can also be used to build the geometrical elements of the *Geometry library*, by using the generalization links to define the heritage between the specific geometrical components (finite or infinite TTRS) and one existing generic component (Figure 35).

Finally, this diagram enables using the *Satisfy* link between a component and a *quantified requirement.*

Scripts for each corresponding GUI button have been developed in a specific toolbar for the *AssemblyDiagram,* in order to offer users only authorized elements that can be placed in this diagram.

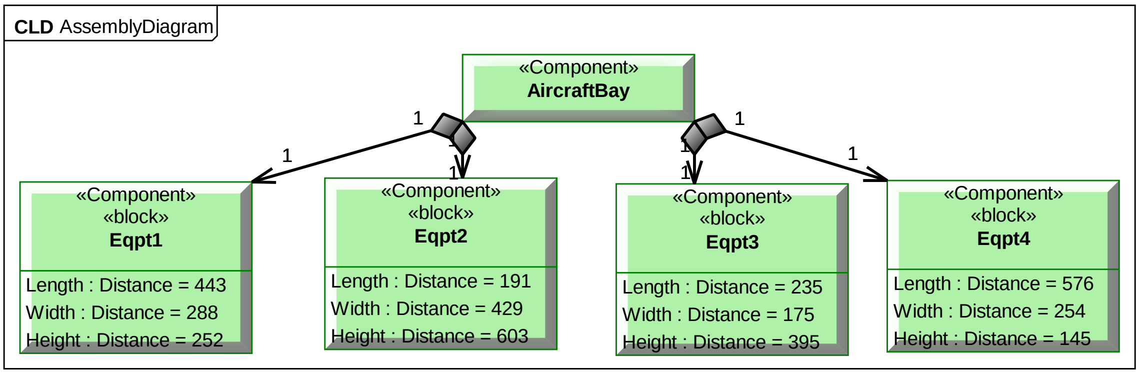


Figure 81: Example of AssemblyDiagram used by design actor user

- Developers usage:

This diagram has made it possible to create the TTRS classes and their corresponding *MRGE* and *RGE* elements.

This diagram also enables the creation of a new *TTRS*, as described in Chapter 6 section I.B.2, since a *TTRS* can be composed (through the *composition link*) of different sub*TTRS*, but also inherits from an existing *TTRS class*.

These different functionalities have been implemented via scripts through a dropdown menu and corresponding buttons in the toolbar of the *AssemblyDiagram,* as presented in Figure 35, chapter 3.

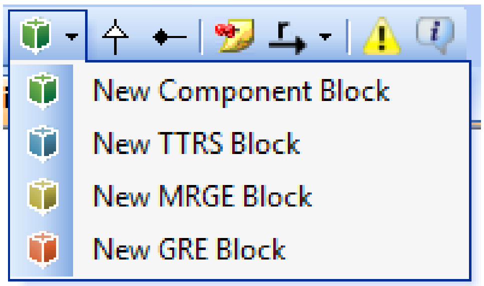


Figure 82: Toolbar of the AssemblyDiagram.

STRD AssemblyParam

As described in chapter 3, the *AssemblyParam* diagram is a parametric diagram dedicated to geometrical relations between the parameters of components and TTRS (block stereotypes), and also of MRGE and RGE, which are not block stereotypes but simply classes and thus not eligible for inclusion in a SysML parametric diagram (Figure 36).

This diagram associated with one component enables specifying the quantitative relations between the parameters of all the components (even itself) described in an *AssemblyDiagram*.

Here again, this diagram can be used by both the design actors and the developers for the following respective usages:

- Design actors can specify the mathematical relations between the parameters of (sub-)components and the constraints defined in the *AssemblyDiagram,* to precisely position them in relation to each other.
- Developers needs this diagram to define TTRS class and Geometry library elements (Figure 35), by specifying the geometrical relations between the RGE of the subTTRSs contained in a TTRS. These relations allow positioning and orienting the subTTRS in relation to each other.

In order to enable these different functionalities in the GUI of PTC environment, the following elements have been added in the right-click menu in the *STRD AssemblyParam,* via scripts (Figure 83).

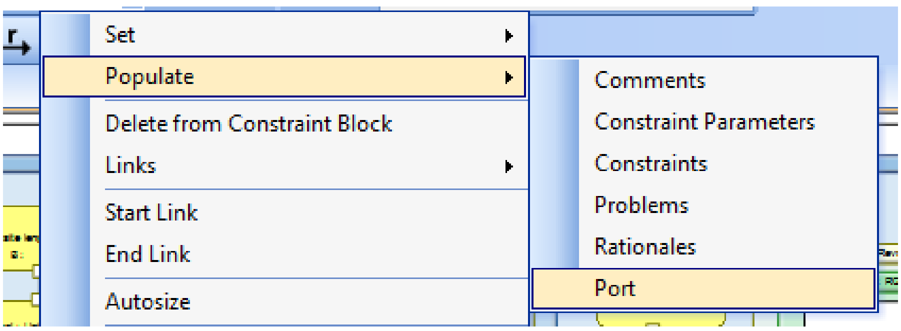


Figure 83: Contextual Menu (right-click) developed by scripts.

b. Scripts

New elements

Scripts have been used to improve the PTC GUI, by managing contextual menus. For example, for each stereotyped element defined in the *profile diagram* (Figure 80), a right-click menu has been developed using scripts to help users add these elements in custom diagrams, as presented in Figure 84.

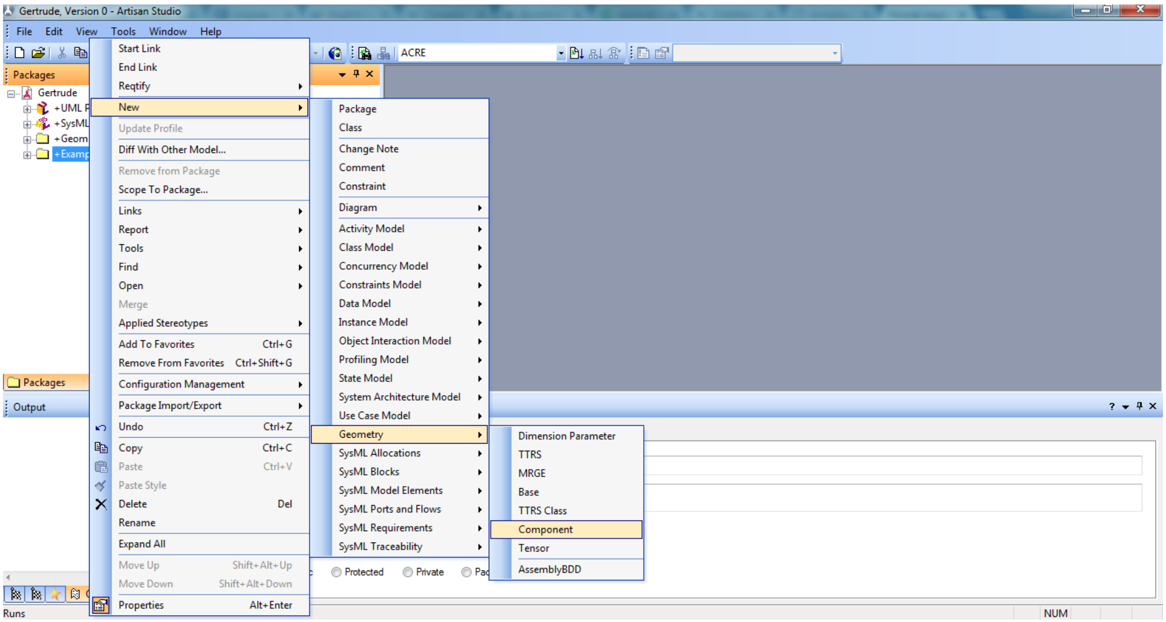


Figure 84: GERTRUDe GUI: example of a component addition.

User GUI

These scripts can also be applied directly in custom diagrams to prevent unauthorized elements. Figure 85 describes an error message generated by a script with an invalid drop from an unauthorized element of the model tree into an *AssemblyDiagram*.

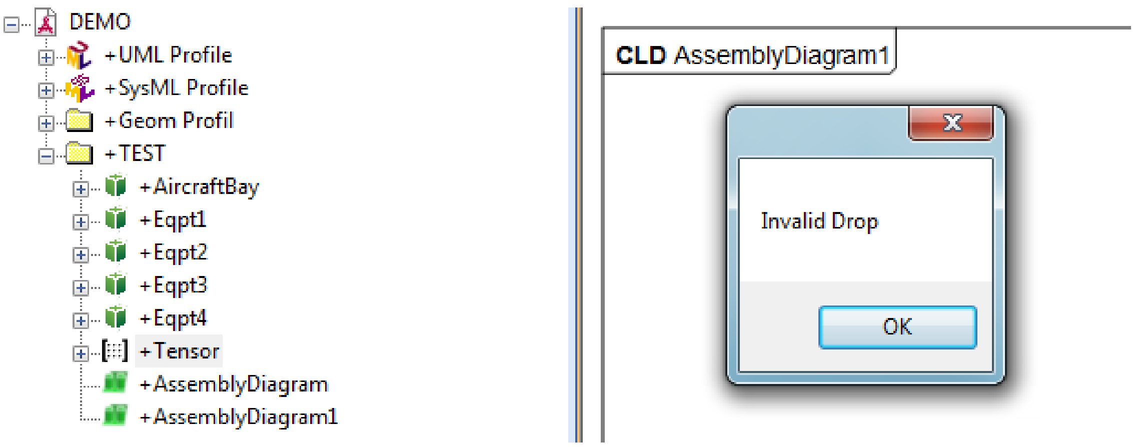


Figure 85: Example of error generated by a script included in GERTRUDe.

SysML extension-based constraints management

A Metamodel describes the semantics of the SysML extension or the UML profile, but the links between their elements are ensured by scripts which define its syntax. Without syntax, a semantic is not sufficient, since the syntax describes the relation between lexical elements and corresponds to the grammar of a programming language. Scripts form the syntax of the SysML extension or the UML profile, by allowing or prohibiting relations between certain elements. For example, in the GERTRUDe extension, the script "LimitTTRS" allows creating a composition link between a component and only one TTRS, since we stipulate that a component has only one (global) TTRS, and even its subcomponents have their own TTRS. A standard language named OCL (Cabot & Gogolla, 2012), is used to describe these rules independently of the programming language. An example of OCL rules in the GERTRUDe extension is given below (Figure 86). These rules have been written in VB.Net to be implemented in scripts in Artisan Studio.

	Rule LimitTTRS
1.	context Component inv:
2.	self.TTRS < 2

Figure 86: OCL rules to limit the number of TTRS involved by constraint.

2. Geometry library: the *Geometry library*

This library contains the definition of:

- the TTRS classes including their invariance torsor and corresponding MRGEs;
- some TTRS and related TTRS constraints of simple finite or infinite geometries;
- the 13 TTRS constraints for the relative component positioning.

The *Geometry library* developed contains the TTRS description of simple geometries. The geometries implemented are: Sphere, Cylinder (hollow and full), Parallelepiped (hollow and full), Cone (hollow and full), and Torus. Since *AssemblyParam* describes the TTRS and the constraint links between these TTRS, the Figure 35 shows the *AssemblyParam* of the Finite Cylinder TTRS model. A Finite Cylinder is an *Infinite Cylinder* cut by two planes, as described in the left of the figure. Constraint *C08* specifies the position of the *Infinite Cylinder* in relation to one plane. This constraint ensures the orthogonality between the plane and the axis of the *Infinite Cylinder*. Then, the corresponding built *Finite Cylinder,* which belongs to the *Revolute* class, inherits its MRGE from the corresponding TTRS class. As its TTRS belongs to the *Revolute* class, its MRGE is a point and a line. Next, we choose to position the line coinciding with the axis of the *Infinite Cylinder*, and define the point as the barycenter of the Cylinder. Thus we use constraint *C11* between the *RGE line* of the *Finite Cylinder* and the *RGE line* of the *Infinite Cylinder*. Afterwards, constraint *C03* is used between the *RGE point* of the Finite Cylinder and a plane of this Finite Cylinder to locate it.

Finally, the *Equality* constraint returns the output identical to the input. *Equality* is necessary because in the SysML parametric diagram, it is not possible to equal two attributes without any constraint. The same formalism was used for the GERTRUDe Assembly IBD.*TheReSE*

1. Profile diagram

The profile diagram developed to define a stereotype for each class is described in Figure 87. To simplify, this profile diagram does not present the GERTRUDe stereotypes. The stereotype *Component* is similar to the UML stereotype *Block* in SysML. Moreover, as a UML class can get more than one stereotype and so be composed for example of both stereotypes: block and component, this allows reusing existing models. In addition, the *Component* stereotype cannot correspond to only one real object, like a calculator, a motor, etc., but it also consist of many components, for example, a plane is composed of calculators, motors, and others.It is possible to add an "*Interactive physics*" stereotype for each component. The stereotyped class *Interactive physics* applied to thermic analysis corresponds to a field, whether emitting or receiving, which is represented by a vector for radiation and convection, or scalar for convection. If it is stereotyped as a *medium*, the kind of thermal propagation mode considered has to be stipulated: radiation, convection, conduction. A TTRS is associated with each propagation mode, in order to define its (emitting or receiving) geometry. A component has to fulfill a *Receiving physics req* (requirement) value (threshold, temperature acceptable range, etc.) that can be a vector, or a scalar. This value must be automatically transferred to the *Receiving Physics* when a dependency flow is added to the *Interactive Physics* stereotype.

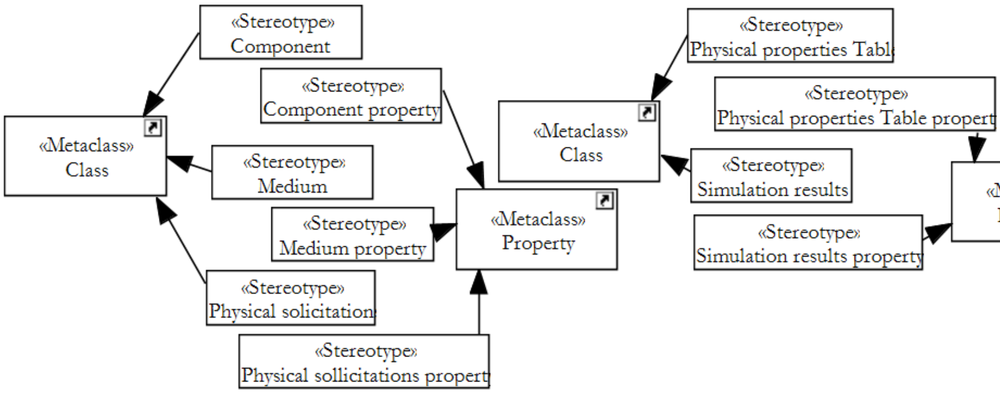


Figure 87: Profile diagram of TheReSE.

Concerning the custom diagram of TheReSE, two diagrams have been added: the first is stereotyped from the class diagram (CLD). It has been developed and called *PhysicsDiagram*. The TheReSE metamodel Figure 63 described in chapter 4 VII highlights that a component is composed of one material and different physics elements. Physics elements correspond to various boundary conditions depending on the face considered. The GUI of the *PhysicsDiagram* is shown in Figure 88.

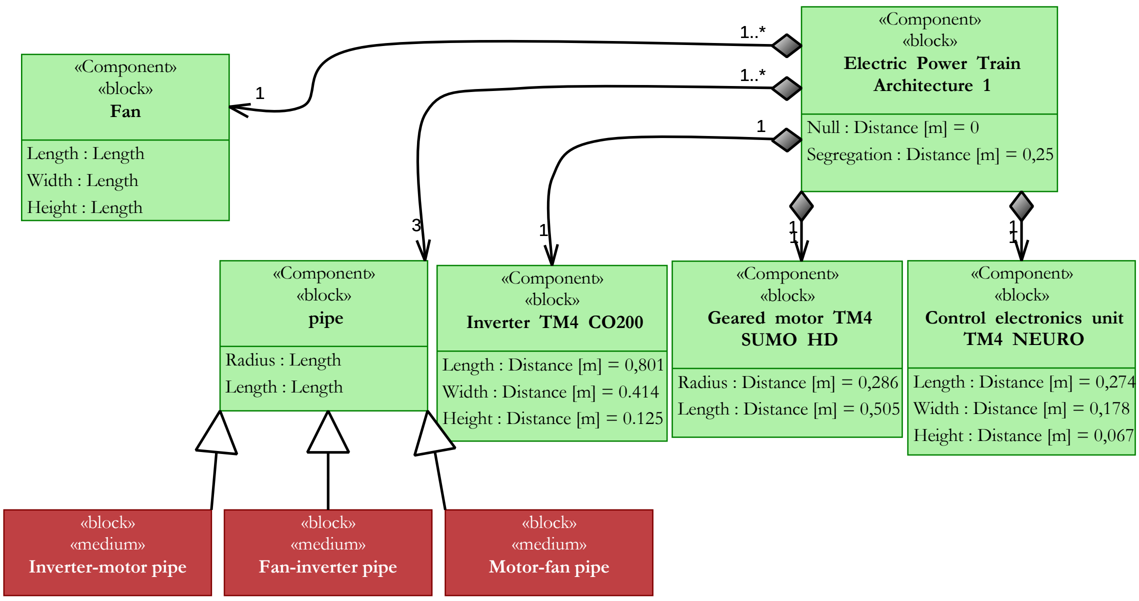


Figure 88: Example of a PhysicsDiagram GUI.

The second is the *InteractionDiagram* (Figure 89) which is a custom diagram from the STRD (composite STRucture Diagram) that presents the same topological view to describe the links (Interactive Face cf Chapter 4 section V.A.3) between the physical media and the components, as in Modelica language, which facilitates the corresponding model transformation.

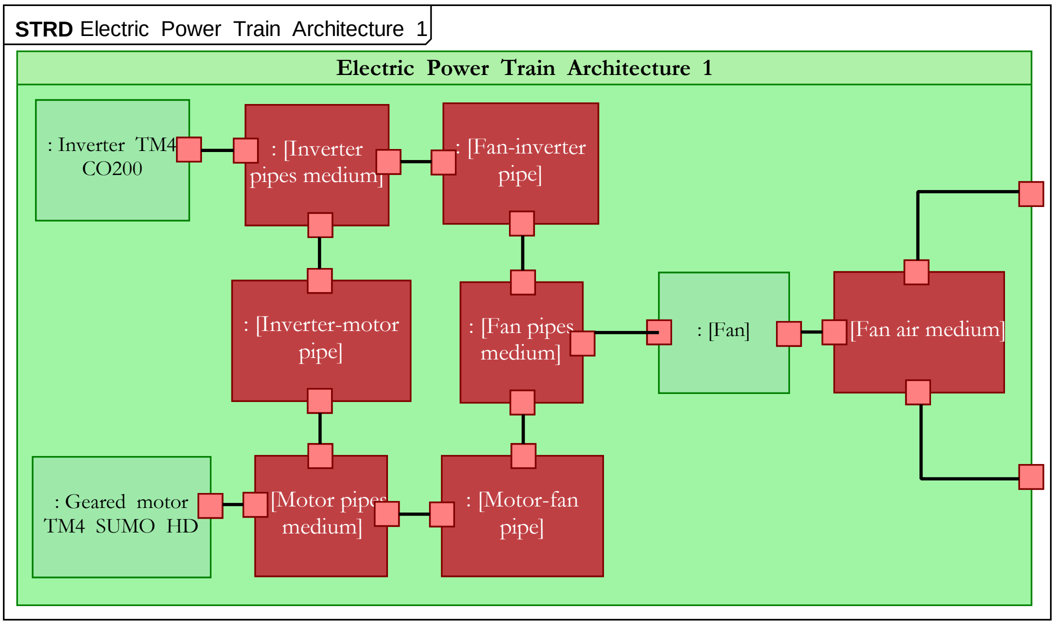


Figure 89: Example of an Interaction Diagram.

2. Scripts

The scripts that ensure that the rules are respected and the automation tasks, such as menus, have not yet been developed in TheReSE given the requirement that the demonstrator must be built in 3 years. The profile has been implemented, but the scripts are not developed.

C. Layer Management to improve readability

The GERTRUDe and TheReSE extensions developed increase the model's complexity: e.g. a classical SysML block, which becomes a component with TheReSE, has at least 3 new items (material, TTRS, boundary condition). In addition, when it is a medium, it can also include the propagation law. Thus, as the complexity of the SysML model discourages its industrial use, as stated previously, we have enriched our work with several GUI implementations to demonstrate the interest of the GERTRUDe and TheReSE developments in improving the design process to the IRT industrial partners. Therefore, we have first built new diagrams for the geometry and thermal modeling, as described in section III.A.1.a Also, concerning the GUI for the geometry layer management, only three layers have been implemented in the *AssemblyParam,* as described in Figure 90. The first layer corresponds to the main TTRS (linked to a component) considered, called "*Root TTRS*", and its positioning in relation to another component. The second links the positioning of each component *subTTRS* in relation to each other, called "*subTTRS"*. Finally, the third and last layer describes the relative positioning of *subTTRS MRGE* between the corresponding *RGEs,* called *"MRGE position"*. This layer model has been implemented in VB language. The modeling items are automatically distributed in the different layers according to the level of the TTRS (e.g. *RootTTRS* or *subTTRS*) considered in the diagram. First, the *MRGEs* of the *RootTTRS* and *subTTRS* are stored. Then, the constraints that link the *subTTRSs* are set in the *subTTRS* layer, and the constraints that link the *MRGEs* and *subTTRS* are set in the *MRGE* position layer.

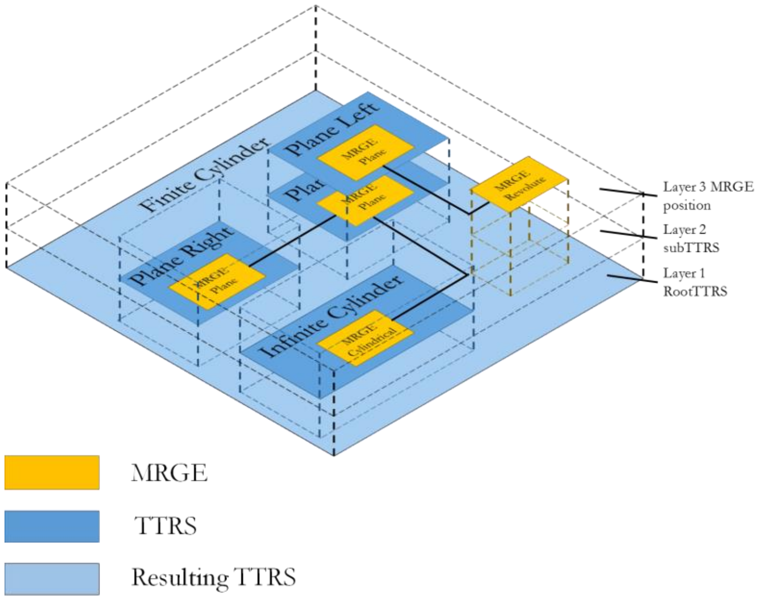


Figure 90: TTRS layer management.

The algorithm (Algorithm 2) supporting this model item distribution is described below.

Algorithm Automatic layer determination

Input :
- Let *rootTTRS* be the component that has to be cut

Output :
- Let *subTTRSLayer[ ]* be a list of items which belong to the subTTRS layer
- Let *MRGELayer[]* be a list of items which belong to the MRGE position layer

```
BEGIN
    Initialize subTTRSLayer and MRGELayer as a void list of items
    Let subTTRSs[] a list of the TTRS inside rootTTRS
    Let MRGE the MRGE of the rootTTRS
    subTTRS ← rootTTRS.subTTRS
    subTTRSLayer.add(subTTRS)
    MRGE ← rootTTRS. MRGE
    MRGELayer.add(MRGE)
    for each subTTRS in subTTRSs, do
        for each link in subTTRS.links, do
            for each linkc in link.constraint.link, do
                if linkc.TTRS in subTTRS, then
                    subTTRSLayer.add(link.constraint)
                Else
                    MRGELayer.add(link.constraint,)
    return subTTRSLayer, MRGELayer
END
```

Algorithm 2: Automatic layer determination.

Where:

- *subTTRSLayer* is a list of items which belong to subTTRS (Layer2).
- *MRGElayer* is a list of items which belong to the MRGE position (Layer3).
- *rootTTRS* is the TTRS that belongs to the RootTTRS layer (Layer 1)
    - .*subTTRS* is the list of the subTTRS which belong to *rootTTRS*
    - . *MRGE* is the MRGE that belongs to the *rootTTRS*
- *subTTRSs* is the list of TTRS which belong to the *rootTTRS*
    - .*links* is the list of graphical connections with TTRS
        - .*constraint* is the constraint associated with the connection
        - .*TTRS* is the TTRS associated with the connection
- *MRGE* is the *MRGE* that belongs to the *rootTTRS*. *MRGE* is in the MRGE layer (Layer3)

Finally, this demonstrator shows the feasibility of such an approach.

The developments in SysML mainly concern GERTRUDe, because this work addresses a feasibility study. However, all the elements developed in the demonstrator with GERTRUDe can be transposed in TheReSE. For example, it would also be possible to create a library of materials, such as the geometry library. Regarding thermal interaction modeling, we propose to model one layer per propagation mode, as shown in Figure 91.

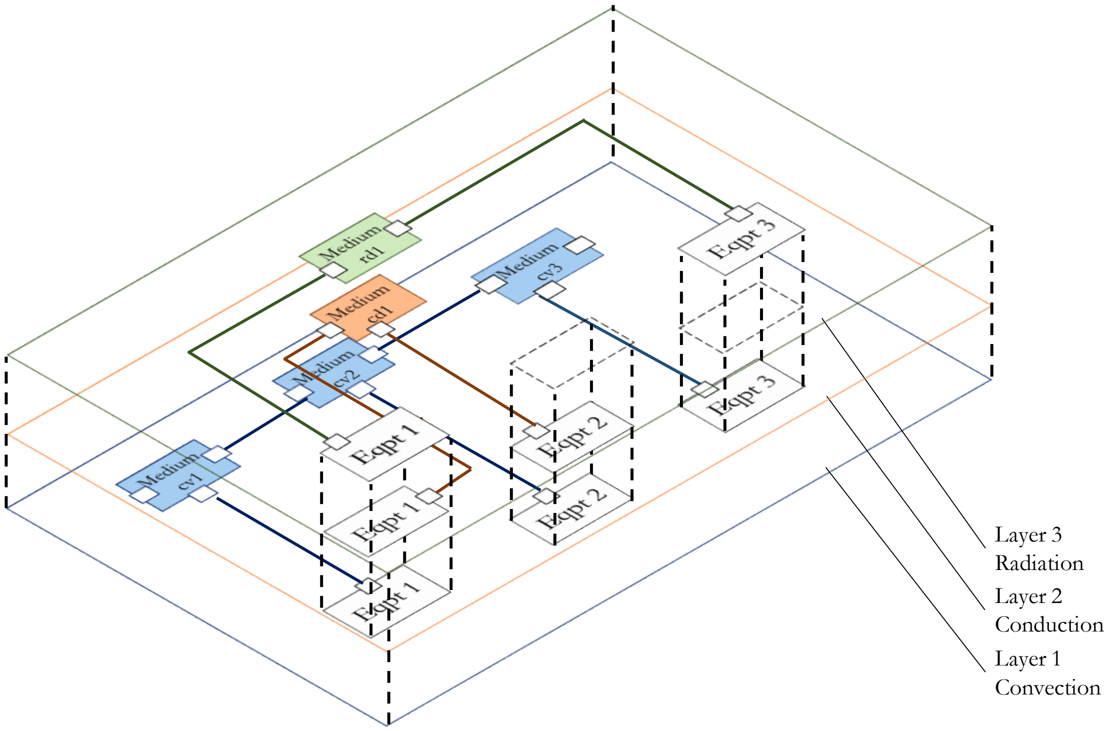


Figure 91: Thermal Physical Layer Management.

IV. Development in 3D CAD (FreeCAD environment)

Concerning the 3D environment, the first script produced addresses the model transformation from GERTRUDe to the 3D environment in FreeCAD. The main difficulty of this model transformation consists of the geometric constraint solving, since FreeCAD does not integrate a 3D constraint solver.

A. Geometric constraint solving

1. Different constraint solving methods

Different methods can be used to position two geometric elements associated by constraints. Two main geometrical constraint solving method are compared here. The first is the Constructive Approach while the second is the degree of freedom analysis. To compare the different approaches, we choose the example of the finite cylinder. The RGE line, plane, and point are represented by vectors. Other representations are available to define lines and planes, as described in (Cubélès Valade, 1998).

a. Constructive approach

The constructive approach solves geometric constraints by steps. Each constraint is solved one after the other, and the surfaces are moved from their locations to closer constrained positions. An example of a finite cylinder is provided in Figure 92. In this example, we consider the following parameters for the two necessary steps:

$$\begin{cases} \overrightarrow{n_{Pr}}\ normal\ vector\ of\ the\ right\ plane \\ L\ a\ point\ of\ the\ left\ plane \\ \overrightarrow{d_c}\ director\ vector\ of\ the\ cylinder\ axis \\ C\ a\ point\ of\ the\ cylinder\ axis \\ \overrightarrow{n_{pl}}\ normal\ vector\ of\ the\ left\ plane \\ R\ a\ point\ of\ the\ right\ plane \\ O\ is\ the\ origin\ point \end{cases}$$

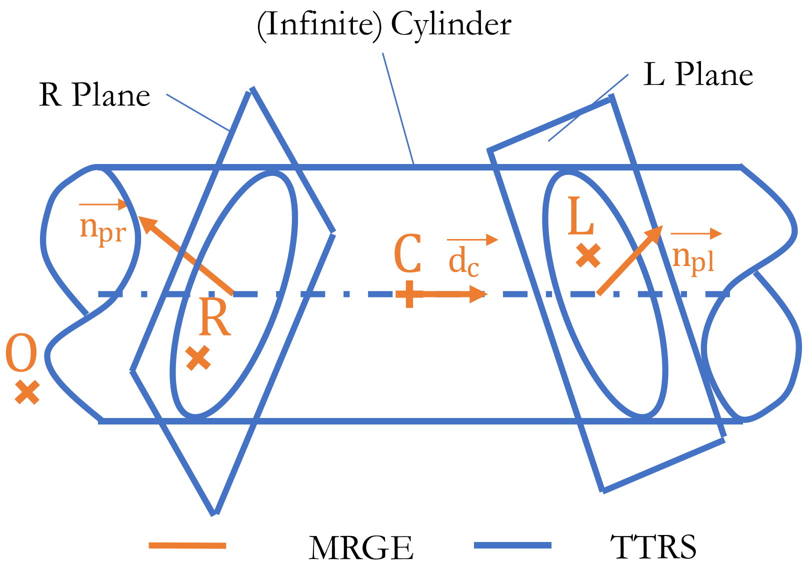


Figure 92: Constructive approach used on the finite cylinder example.

The first step is to solve the *parallelism constraint*, which is a perpendicularity constraint between the right plane and the axis of the infinite cylinder. The perpendicularity between a plane and a line is an equality constraint between the normal vector of the plane and the director vector of the cylinder axis. However, the *parallelism constraint* does not restrict the plane orientation (normal vector). Consequently, a vector product is required to ensure the parallelism of the normal vector of the plane with the cylinder axis without the need to specify planar chirality. Then the *parallelism constraint* between two planes (C6) is defined as follows:

$$(C6) : \begin{cases} \overrightarrow{n_{pr}} \wedge \overrightarrow{d_c} = \vec{0} \\ \overrightarrow{CR} \wedge \overrightarrow{d_c} = \vec{0} \end{cases} \tag{40}$$

After this first step, point R is definitively positioned thanks to 6 equations. The second step first addresses the solving of the *perpendicularity constraint*, which ensures parallelism between two planes. With this constraint the planes are oriented, as the normal vectors are opposites. Then the *perpendicularity constraint* between a line and a plane (C8) is described as follows:

$$(C8): \begin{cases} \overrightarrow{n_{pr}} = -\overrightarrow{n_{pl}} \\ \overrightarrow{RL} \,.\, \overrightarrow{n_{pr}} = d \end{cases} \tag{41}$$

Afterwards, the cylinder is positioned by way of 4 equations.

Thus, solving Finite Cylinder constraints with this constructive approach requires resolving 10 equations (6 for the first constraint and 4 for the second one).

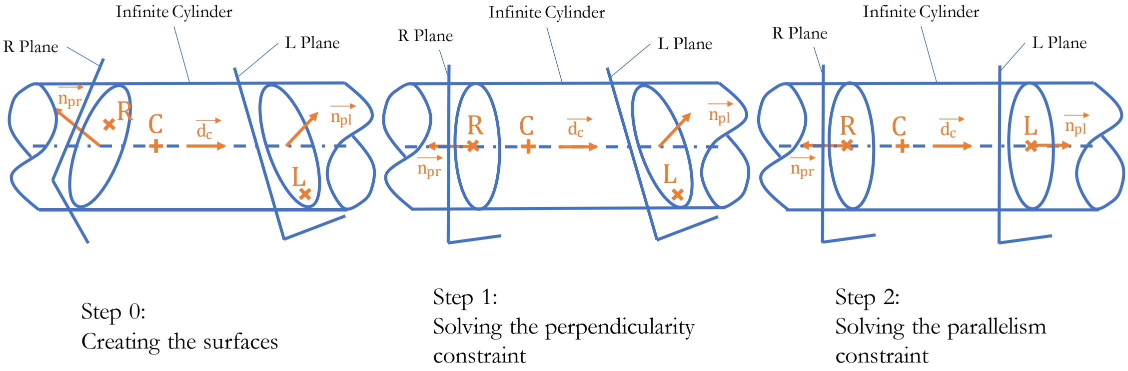


Figure 93: Solving geometrical constraints of the finite cylinder surfaces using the constructive approach.

b. Degree of Freedom Analysis

Degree of Freedom Analysis is a solving method for assembly (Kramer, 1991). Each degree of freedom is parametrized (including the chirality parameter) and the user can change this parameter to move the component under constraints. The chirality parameter is a new parameter added through the use of the half space concept. This parameter allows adding a degree of freedom relating to the direction of the face (inside and outside). An example is given with the previous example of the Finite Cylinder.

The *Parallelism constraint* between two planes (C6) is now given by the following equations:

$$Parallelism\ plane/\ plane\ (C6): \begin{cases} \overrightarrow{n_{pr}} = \delta.\overrightarrow{d_c} \\ \overrightarrow{CR} = \alpha.\overrightarrow{d_c} \end{cases} \quad (42)$$

Where;

- δ is a parameter which can be -1 or 1, depending on the plane chirality.
- α is the distance between the point R and the point C.

The perpendicularity constraint between a plane and a line (C8) is given by the following parameter:

$$(C8): \begin{cases} \overrightarrow{n_{pr}} = -\overrightarrow{n_{pl}} \\ \overrightarrow{LR} = d.\overrightarrow{n_{pr}} + (\beta\cos(\theta) - \gamma.\sin(\theta)).\overrightarrow{x_L} + (\beta.\sin(\theta) + \gamma.\cos(\theta)).\overrightarrow{y_L} \end{cases} \quad (43)$$

Where:

- $\overrightarrow{x_L} = \begin{cases} \overrightarrow{n_{pr}} \wedge \vec{x} & \text{if } \overrightarrow{n_{pr}} \wedge \vec{x} \neq 0 \\ \overrightarrow{n_{pr}} \wedge \vec{y} & \text{if } \overrightarrow{n_{pr}} \wedge \vec{x} = 0 \end{cases}$
- $\overrightarrow{y_L} = \overrightarrow{n_{pr}} \wedge \overrightarrow{x_L}$
- θ is the angle between the Right plane and the Left plane
- β is the distance between point R and point L along axis $\overrightarrow{x_L}$
- γ is the distance between point R and point L along axis $\overrightarrow{y_L}$

The perpendicularity constraint solved by the degree of freedom analysis requires the creation of a local frame $\{\overrightarrow{n_{pr}}, \overrightarrow{x_L}, \overrightarrow{y_L}\}$

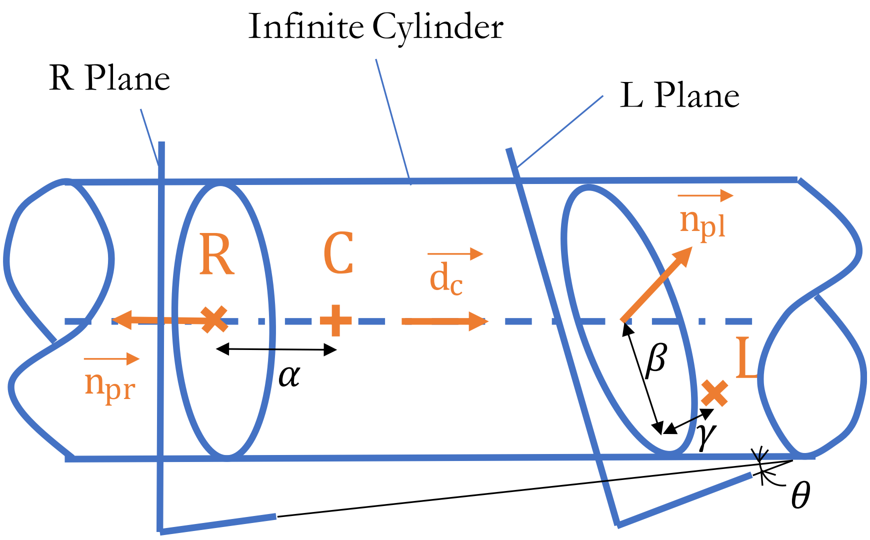


Figure 94: Degree of Freedom modeling of the Finite cylinder.

Finally, to generate a Finite cylinder, the following equation system of 24 equations must be solved:

$$\begin{cases} \overrightarrow{n_{pr}} = \delta.\overrightarrow{d_c} \\ \overrightarrow{CR} = \alpha.\overrightarrow{d_c} \\ \overrightarrow{n_{pr}} = -\overrightarrow{n_{pl}} \\ \overrightarrow{RL} = d.\overrightarrow{n_{pr}} + (\beta\cos(\theta) - \gamma.\sin(\theta)).\overrightarrow{x_L} + (\beta.\sin(\theta) + \gamma.\cos(\theta)).\overrightarrow{y_L} \end{cases} \quad (44)$$

c. Choice of the constraint modeling approach

These two approaches have been implemented to solve geometrical constraints and a short comparison is given here.

The first approach implemented was the constructive one, since there are fewer equations to solve than for the second one and thus the resolution appears faster. However, the solved positions of the geometrical elements strongly depend on their initial absolute positions. Moreover, the order of the constraints is important. For example, in the example of the finite cylinder, if the *C06 constraint* (parallelism constraint) is solved before the *C08 constraint*, the *C06 constraint* must be solved again, otherwise the initial *C06 constraint* will be not respected. For this reason, the TTRS constraints must be ranked by order of dependence: the constraint with the most dependences must be solved before the constraint with less dependences.

On the other hand, the degree of freedom analysis has been implemented in the demonstrator. This approach is longer to model, especially as trigonometric relations are not linear. However, this resolution method is interesting because it is flexible. It allows positioning different elements even if there are not enough constraints to solve it. After the resolution, not all the solved degrees of freedom are set with the 0 value. Thus the position and direction of component can be set by the users. The time needed for solving with this solving analysis can take a long time. To reduce it, the equations have been reordered as described in paragraph II.A.4.a However, even when ordering the equations, the solving time is still long. For example, solving the finite cylinder takes approximately 5 min with a common laptop. In order to reduce the calculation time, the constraint solving results are stored in a Python file including a unique identifier for each TTRS constraint. This TTRS file is browsed accordingly before solving the TTRS, in order to verify that it was not solved before.

The equation solving is performed by a Python library called "*Sympy*", which has the advantage, when performing a Degree of Freedom Analysis, of managing symbolic computation. The difficulty of this model is that the variables to be solved must be specified for each equation. It can be difficult to determine them, because these variables strongly depend on the other associated constraints. Thus, a variable to be solved is proposed for each TTRS equation.This variable can be changed according to the whole system of equations involved.

2. Model transformation

Once the TTRS constraints have been solved in Python, a model transformation is performed in order to create a 3D CAD model from the GERTRUDe model. The theory underlying the model transformation process is described in sections II.A and II.B . In practice, elements are first transformed from the GERTRUDe model to the Python model. The thermal 3D sketcher metamodel is composed of the GERTRUDe, the transformation and the FreeCAD metamodels.

The TTRS constraints are solved using the Sympy Python library. Then, the TTRS are transformed into faces using the method described in chapter 3 section III.B.1. Finally, each face is generated in the FreeCAD environment, and then the volume construction is performed thanks to half space concept explained in chapter 3 section II.A.2.a . All the assembly modifications are driven by Python-based developmentsdrive, so that if the 3D architects modify certain assembly constraints in the FreeCAD environment, these modification must be updated in the System model.

B. *SAMOS processing*

1. Definition of geometry

To create the geometry of a component directly in the 3D tool environment, a special workbench called 3D Sketcher has been developed in FreeCAD (Figure 95). It provides a library of simple components that the 3D architects or the designers can combine in order to create a new assembly. The 3D architects can then move components, and when the components have been modified and architecture parts completed, all these 3D CAD data are traced back to SysML. The GUI of this workbench is still under development, as it must still integrate icons for the different TTRS positioning constraints.

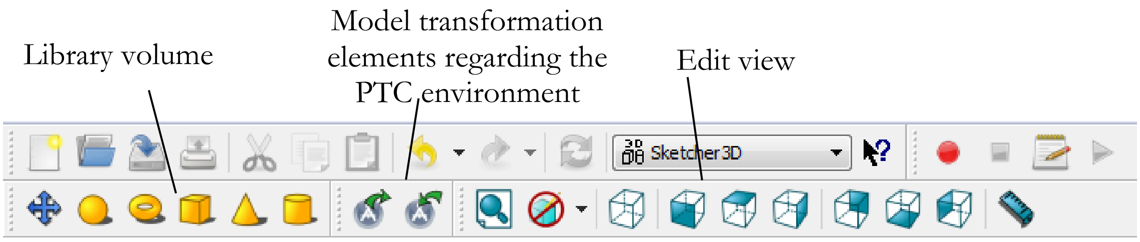


Figure 95 : GUI of the thermal 3D Sketcher workbench.

By using the Constructive Solid Geometry (CSG) approach, it is then possible to build more complex geometries with a sequential composition of different geometrical objects, performed through boolean operations. Moreover, for useful geometries that are not yet implemented in the library, it is possible to add them by using GERTRUDe GUI for developers and modifying the well-documented Python code for the corresponding transformation rules. On the contrary, an error message is displayed when complex geometries are generated in the FreeCAD tool that cannot yet be decomposed using GERTRUDe Library elements.

Finally, if the component considered has no geometry, i.e. any TTRS can be used for 3D part creation, a part is created with geometrical specifications by default. Then, we must create a 3D CAD "void part" which resembles an orange UML class in 3D (Figure 96).

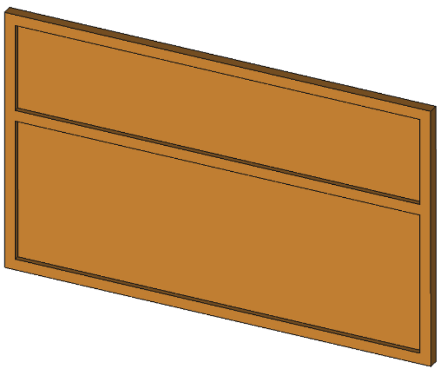


Figure 96: Transformation in the FreeCAD environment of a component initially defined in SysML without geometry.

2. Simulation setup

Once each geometrical element has been positioned, the simulation teams add the simulation specifications through the following steps:

- First, the material of each component can be added. A GUI has been developed in the FreeCAD environment to faciltate this step (Figure 97). As we address the conceptual design phase, the component materials may not yet be known. Regarding convection analysis, as the component temperature depends only on the component geometry, this is not a difficulty. By contrast, for conduction and radiation modeling, the thermal simulation results depend on the component materials. Therefore, we propose material classes with mean values or orders of magnitudes of their physical properties in a list box accessed in the Material Selection array of the Material selection tab (Figure 97).

- The second step concerns the verification of quantified requirements. Indeed, quantified requirements allow providing certain required thermal performance values. The required values can be minimal value, maximal value, or equal value. They will then be automatically compared to the simulation results, and the developed GUI will display a color code (green for fitted results and red for requirements not matched), matching how they fit with the requirements, both in the Requirement tab and in the 3D environment (color of unsuitable components).
- The third step addresses the definition of the boundary conditions. In the case of a solid component, the boundary condition is the constant temperature or the thermal power applied to the component. In the case of a fluid component, the boundary condition is the Inlet and Outlet (boundary conditions (pressure, or velocity and temperature of the inlet) and the face). Once Inlet and Outlet have been defined, the topological cutting of the fluid component can be performed, in order to prepare the Fourth step. This cutting is performed automatically in the framework developed, using an algorithm described further on.
- The fourth step refers to the definition of a thermal medium. As described in Chapter 4 V.A.2, a medium is a special component that contains the law of thermal propagation. A medium is connected between two or more components. It can describe the convection, conduction or radiation propagation modes, since different laws have been implemented, with different parameters.

Material definition

The GUI developed in the FreeCAD environment allows designers to prepare their physical (thermal) simulation related to the requirements specified by the System Architect.

The first task addresses the choice of material of the components. As explained previously, depending on the thermal propagation mode chosen, this choice is not compulsory. Also, when the material has not yet been clearly defined (as is commonly the case in the conceptual phase), the user can simply choose a material class (polymer, ceramic or metallic), with mean values for the physical properties. When the material is not necessary for a specific simulation, the simulation can be launched without any material specification. For example, for a static convection analysis, the heat exchanged between solid and fluid elements does not depend on the solid material (as described thanks to Nusselt correlations in chapter 4 IV.B.2.a)

The choice of material can be provided by a database developed in the csv format file. This database is connected to the demonstrator, and it is possible to modify it, in order to add new materials with properties, as described in Figure 97.

Since only convection and conduction modules have been developed in this demonstrator up to now, the material parameters do not include parameters necessary for the radiation analysis, such as emissivity.

This material allocation task can also be carried out by clicking on the geometrical representation of a component and then selecting the material chosen and clicking on the Apply button (Figure 97).

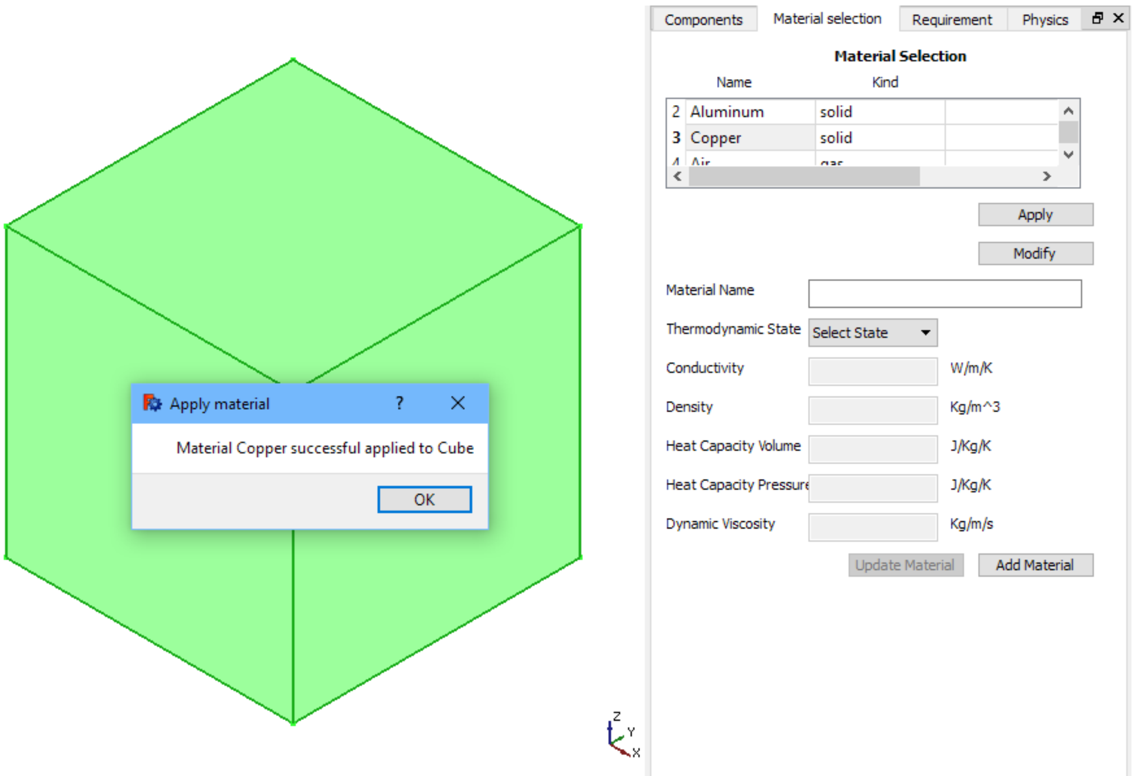


Figure 97: Example of material definition.

Quantified requirement definition

Once a material has been allocated, certain requirements can be added or modified. Requirements can be imported directly from the TheReSE model. Requirements are listed as described in Figure 98.

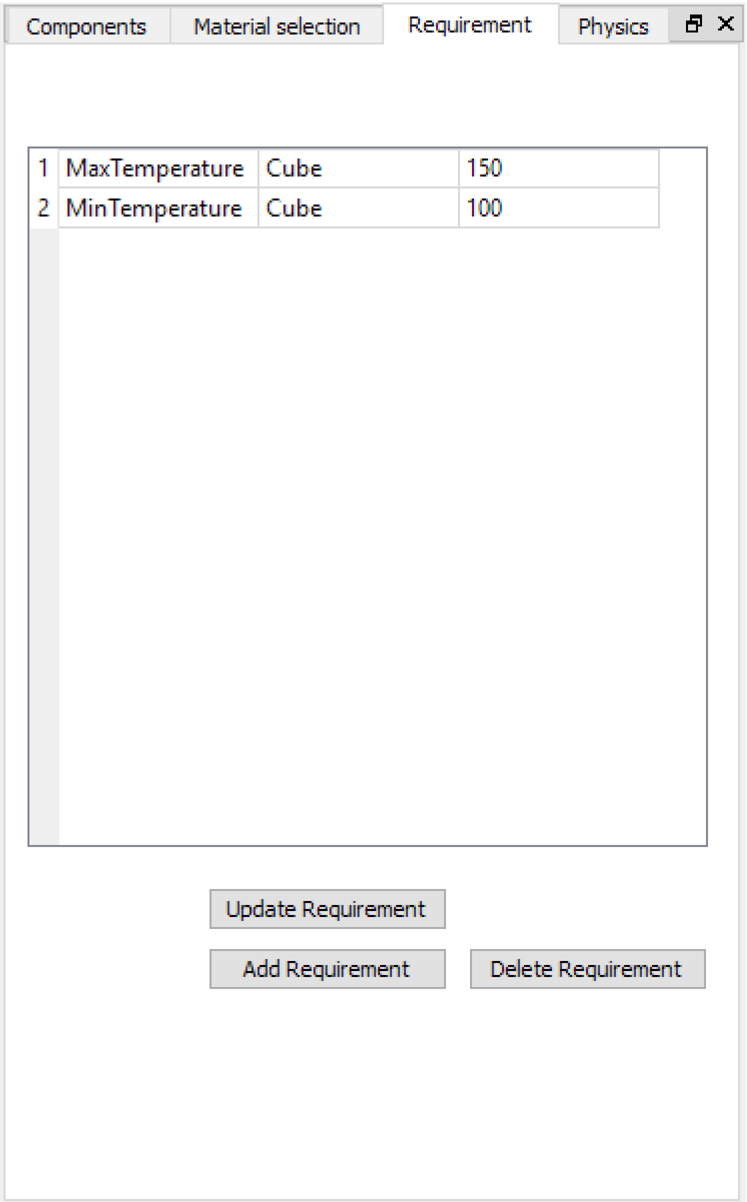


Figure 98: Requirement tab.

In this tab, it is possible to update the values of existing requirements or create new ones, for example, stemming from the business expertise simulation. Figure 99 illustrates how to create a new quantified requirement for the thermal simulation and the various fields which can be modified with the "Update Requirement" button. The first two lines are close to classical requirements: with a name and a text that describe it. Other values are specific to quantified requirements:

- The "Specification rules" list box proposes to define if the requirement is a minimum or maximum acceptable value: it specifies whether the simulation results have to be lower or higher than this specified value.

- The second field, “Required value”, is the required temperature value with units.
- The “Output for simulation” value corresponds to the method chosen for the simulation setup. Simulation can set various output values: the average, maximal or minimal temperature of a component.
- Finally the “Component tab” specifies the component(s) to this requirement has to be applied.

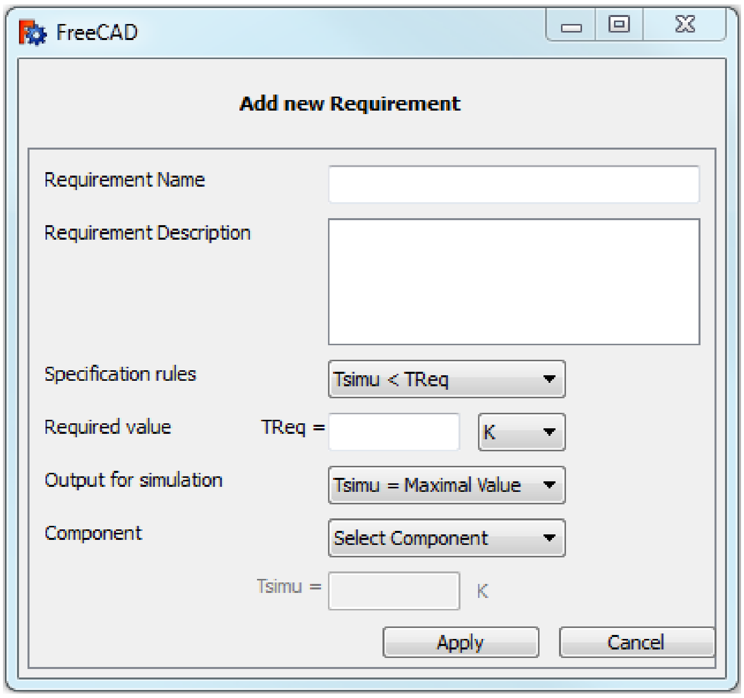


Figure 99: “Add new Requirement” window.

Boundary condition definition

Once a number of requirements have been added to several components the simulation teams must add the physical conditions necessary to perform their simulation.

The first physical elements to be added to the thermal simulation are the component's thermal boundary conditions. First, the Physics Tab (Figure 100) proposes to update the list of components (either originating from the *TheReSE import* (SysML-Python) script, or from new added components in the 3D CAD environment) or to select one. Then, it is possible to add the boundary conditions for each component. As the boundary conditions depend on the thermodynamic state of the component material, the boundary condition window differs as a function of whether the material of the component is solid, fluid/gaseous or none.

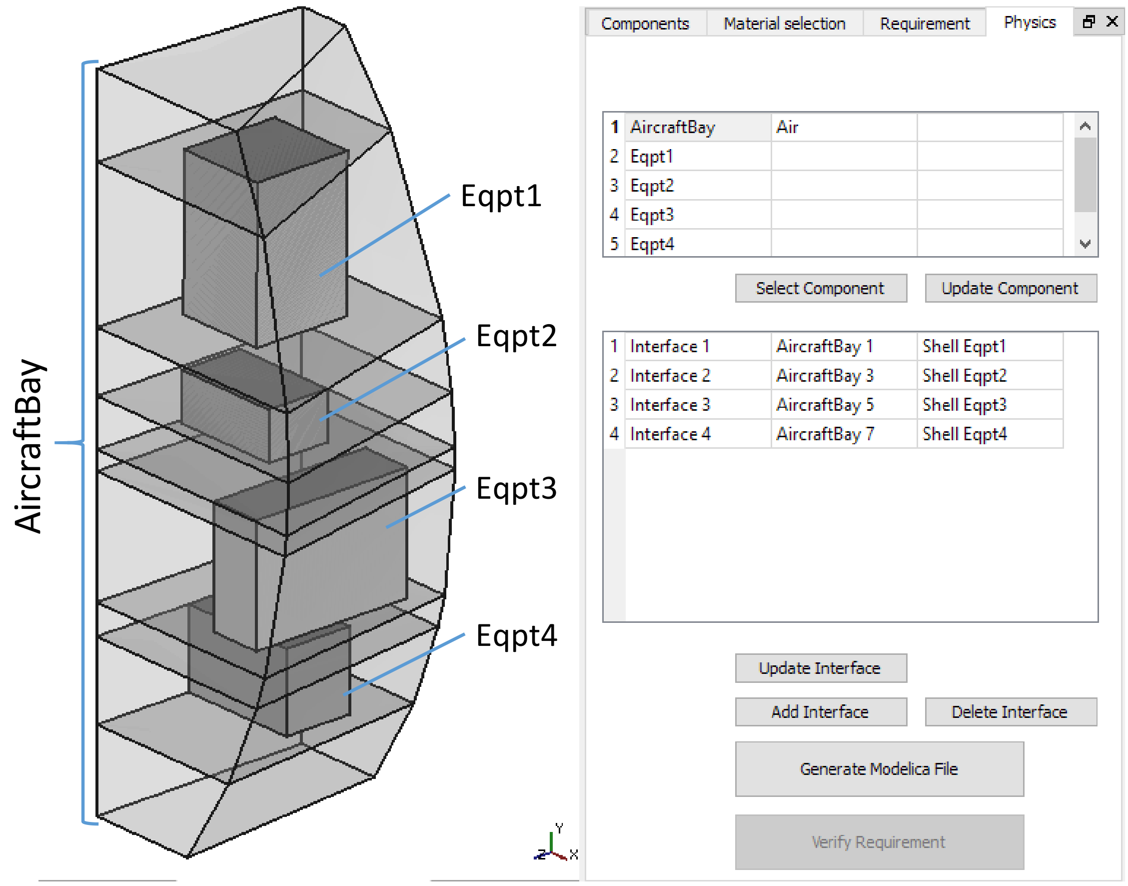


Figure 100: The "Physics Tab" description.

Since the model used for the thermal analysis is a rough model, each component will be considered as one node (Chapter 4 V.B).

The boundary condition of a solid material applied to the corresponding Interactive TTRS node can be a constant thermal dissipated power, a constant thermal dissipated power flux density, or a constant temperature on the external faces, as described in Figure 101.

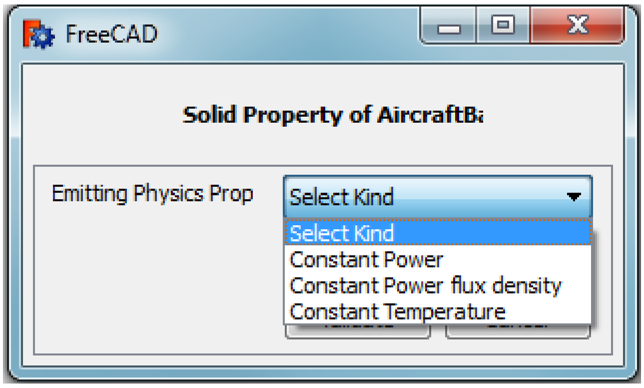


Figure 101: The "Add solid boundary conditions" window.

Concerning the fluid elements, the boundary conditions involve two geometrical elements: the first is the *Inlet* and the second is the *Outlet*. An *Inlet* element can be set as a constant pressure, a constant mass flow or a constant flow rate, and its temperature must be added. For the *Outlet*, the flow property to choose is the same as that of the *Inlet*, except for the temperature, which will result from the simulation (Figure 102). The *Inlet* and *Outlet* specifications require that a geometrical face must have been selected prior to applying conditions.

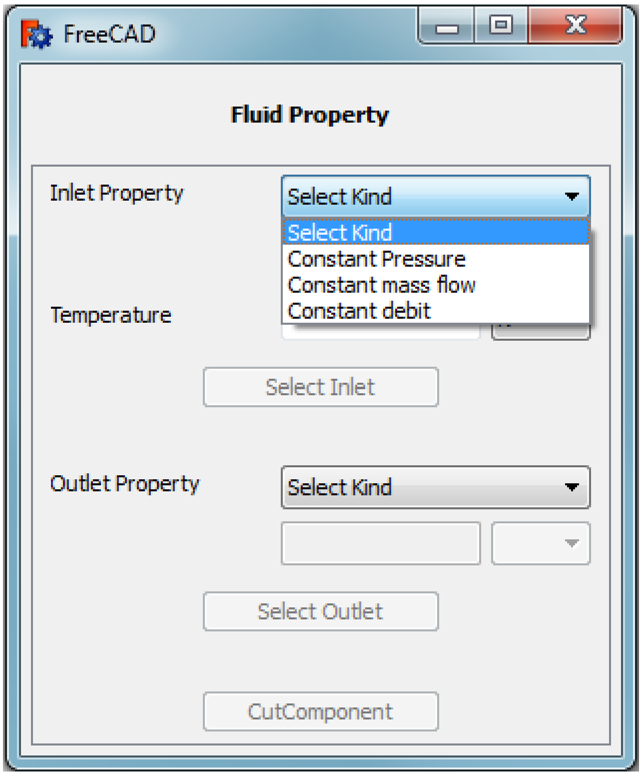


Figure 102: Adding boundary conditions for fluid component

Geometrical demarcation for topological modeling

Regarding the solid components, the geometry does not need to be topologically modified for the simulation since the demonstrator implemented considers each component only as a node. On the contrary, a fluid component will be adapted, as described in Chapter 4 V.B.2, to make the interacting faces stand out. Thus the "CutComponent" button in Figure 102 calls the algorithm "Automatic component cutting" (Algorithm 1).

Media definition

Once the boundary conditions have been added and the component geometry functionally cut, it is possible to proceed to the last step: the definition of each thermal equation to take into account through the definition of the Medium. The Medium was previously defined as a component that carries thermal equations. The definition of this medium must suit the simulation teams. Therefore, a GUI is proposed to the simulation teams to choose the thermal mode they want to simulate (Figure 103). At the moment, only the convection mode has been implemented. The Nusselt correlation equations will be filled directly by the simulation teams in this mode.

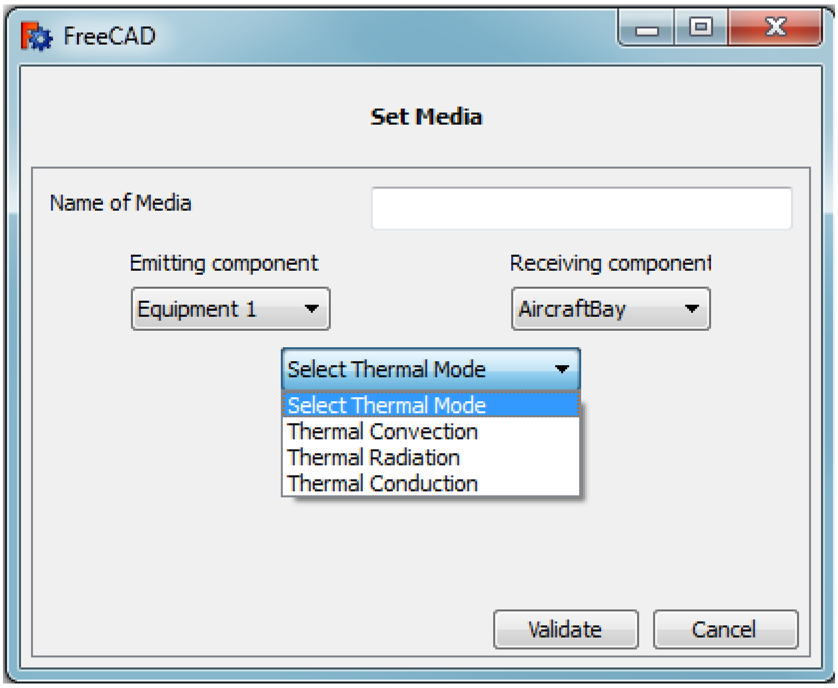


Figure 103: Thermal media definition.

V. Development in Modelica (OpenModelica environment)

Once all the thermal elements have been added to the 3D CAD GUI, a Modelica file can then be generated automatically. This file will take into account the topology of the thermal architecture, the physics and geometry parameters. This Modelica file is prefilled with different parameters: geometrical parameters, material parameters, boundary conditions parameters. The choice of the thermal mode model leads to the transmission of specific parameters. For example, if the only convection is chosen as the thermal model chosen, the material parameters of solid components are not transmitted, since they are not considered in the physical equations of this thermal mode. On the contrary, materials are very important for the radiation modeling, particularly for calculating the emissivity of each element. The parameters necessary also depend on the geometrical view and the thermal law considered. For example, parameters for a convection model based on thermal resistances are different from those necessary for a Nusselt correlation model (fluid parameter, and fluid velocity). Consequently, two types of Modelica model types have been proposed: the first is a new library developed using a coupling convection conduction in one dimension, in which the boundary conditions are fixed by the model; the second is a thermal model of the Modelica standard library using thermal resistances.

A. Development of a Modelica Library related to the shell view.

Usage of analytic calculation for conduction analysis is interesting, due to the precision of its simulation results. However, the projection basis used in the thermal quadrupole model strongly depends on the geometry. In addition, not all the TTRS classes fully support the choice of such a projection basis (as demonstrated in Chapter 4 VI.A.1.b). Finally, due to the previous reason and since the thermal resistance modelling approach Chapter4 III.B. It is more relevant for a hollow geometry, it has been chosen for the 3D thermal Sketcher implementation. Nevertheless, we will also present the Modelica library we have developed based on the component TTRS class to select the best adapted thermal model.

1. New ports added

A Modelica model has been developed from thermal modeling approach described in Chapter 4. First of all, a new port must be added. In the Modelica language, the inner variables of ports can be a flow or a potential. Flow variables will keep the same value in a loop, and Kirchoff's law will be applied at a junction. Potential variables keep the same value at a junction and can vary in a loop.

An example is given in Figure 104 in in which the links are shown in blue. The potential variables are black and called $Potential_{k.1}$ and $Potential_{k.2}$ (with k, being an integer between 1 and 5) and the flow variables are described in red and called $Flow_j$ (where j is an integer which belongs to $[\![1,3]\!]$)

According to this example, the potential variables are given by the following set of equations:

$$\begin{cases} Potential_{k.1} = Potential_{k+1.2} \; \forall \; k \in \; [\![1,4]\!] \\ Potential_{5.1} = Potential_{1.1} \\ Potential_{4.2} = Potential_{1.2} \end{cases} \tag{45}$$

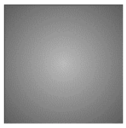
The flow variables are described by Kirchoff's law, with the following equation:

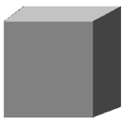
$$Flow_3 = Flow_1 + Flow_2 \tag{46}$$

Each variable of a port described in the table below is a potential variable.

Table 45: New ports for the geometry and material definition.

Icon	Port name	Function	variables

	InMater	Port which contains different material properties	• λ thermal conductivity of material,
			• ρ density of material,
			• Cp thermal mass capacity at constant pressure,
			• $State$ is an integer number, which corresponds to a state:
			  o 1 for solid
			  o 2 for liquid
			  o 3 for gaseous
			• μ material viscosity (only for liquid and gaseous).

	inGeometry	Port which corresponds to a geometrical parameter necessary for computation	• S is the area of a face which exchanges heat. • p is the perimeter corresponding to the prismatic TTRS class. • L is the length of the prismatic TTRS class. • Ri is the inside radius for the spherical and cylindrical TTRS classes. • Re is the outside radius for the spherical and cylindrical TTRS classes. • $CarDist$ is the characteristic distance used for the calculation of the Reynolds number.

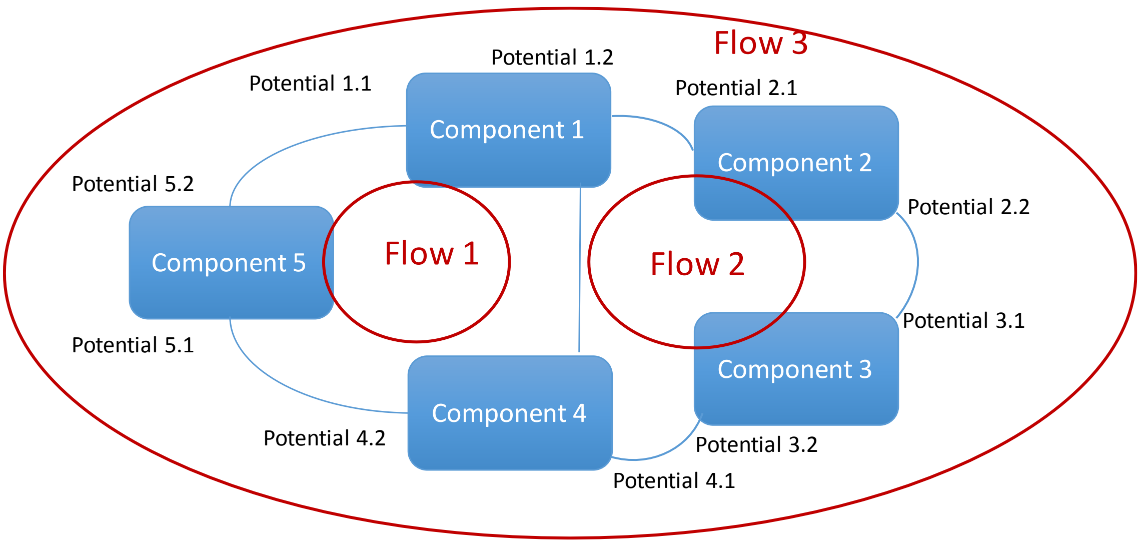
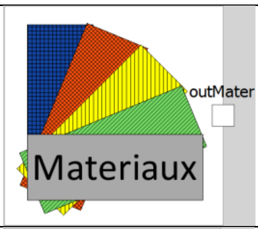
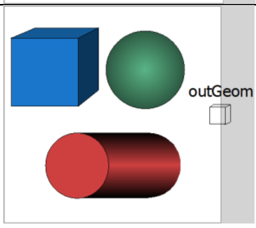
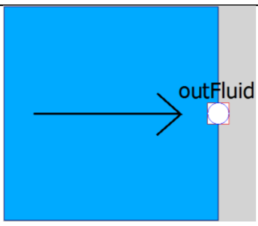
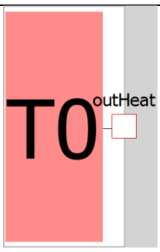
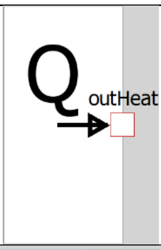
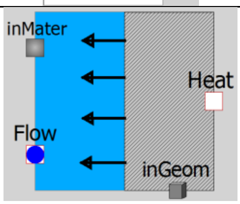


Figure 104: Differences between flow variables and potential variables.

2. Boundary conditions

Modelica models have been developed here in order to include the boundary conditions. The models described therefore include conditions like material, geometrical parameters and heat boundary conditions. They are described in the following table (Table 46: Modelica models developed to define simulation conditions.Table 46).

Table 46: Modelica models developed to define simulation conditions.

Icon	Function	Imposed parameters
	This Modelica model describes the material of the corresponding component.	$(\boldsymbol{\lambda}, \boldsymbol{\rho}, Cp, \boldsymbol{State}, \mu)$ are imposed. Bold variables are requisites in addition to materials
	This Modelica model describes the parameters necessary to define the component geometry.	• $(\boldsymbol{p}, \boldsymbol{L})$ for prismatic element $\begin{cases} CarDist = L \\ S = p * L \end{cases}$ • $(\boldsymbol{L}, \boldsymbol{Re}, Ri)$ for Cylindrical $\begin{cases} CarDist = max\ (L, 2 * Re) \\ S = 2 * \pi * L \end{cases}$ • $(\boldsymbol{Re}, Ri)$ for Cylindrical $\begin{cases} CarDist = 2 * Re \\ S = 4 * \pi * Re^2 \end{cases}$
	The boundary condition concerning the inlet is described here by using the following Modelica model. In this model it is possible to choose if the boundary condition imposed is flow rate (q) or pressure (P).	• (D, T) for a constant flow rate • (P, T) for a constant pressure
	The boundary condition of a component in a solid thermodynamic state can be imposed as a constant temperature.	T is imposed as a constant temperature
	or as a constant dissipated heat.	ϕ (Q) is imposed as a constant dissipated heat
	This Modelica model is uses a Nusselt correlation to add a boundary condition for the convection mode..	$\frac{\phi}{S} = h(T_s - T_f)$

Modeling the material elements and geometrical parameters is performed using specific ports. These ports have been created to manage the transmission of variables. It differs from the classical Modelica approach. The classical approach uses a pointer to a material. Thus the approach using ports appears to be more interesting, because the links between the material, geometry and components are immediately visible when the model is opened. However, the model generated is very complex. Consequently, a modification of the approach developed could be performed to fit with the classical Modelica approach.

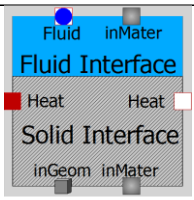
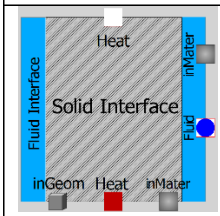
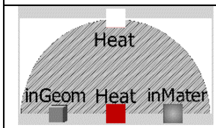
3. TTRS-based convection-coupled conduction modeling elements

It is important to develop the medium before modeling the boundary condition approach. The *Media* developed address the coupling between conduction and convection. For a simple TTRS class, it will represent a 1D conduction element coupled with a convection analysis. The differential equations are pre-calculated. All the models developed are described in the following table.

Then, for each *heat port*, the temperature and the heat flux are calculated according to the previous equations. A and B are constants to be solved, depending on the boundary condition.

(T_a, Q_a) designate variables of the red heat port and (T_b, Q_b) designate variables of the white heat port.

Table 47: Conduction Modelica models according to the component TTRS class involved.

Icon	Function	Equation
	Conduction element for the Prismatic TTRS class	We write $\alpha = \sqrt{\frac{h*p}{\lambda*S}}$ $\begin{cases} T_a = A + T_{fluid\ Int} \\ \frac{Q_a}{S} = -\lambda * B * \alpha \end{cases}$ $\begin{cases} T_b = A * \cosh(\alpha * L) + B * \sinh(\alpha * L) + T_{fluid\ Int} \\ \frac{\phi_b}{S} = \lambda * \alpha(B * \cosh(\alpha * L) + A * \sinh(\alpha * L)) \end{cases}$
	Conduction element for the Cylindrical TTRS class	We write out $\alpha = \sqrt{\frac{h}{2*\lambda*L}}$ $\begin{cases} T_a = A * I_0(\alpha * R_i) + B * K_0(\alpha * R_i) \\ \frac{\phi_a}{4 * \pi * R_i} = -\lambda * \left(A * I_0'(\alpha * R_i) + B * K_0'(\alpha * R_i \right. \end{cases}$ $\begin{cases} T_b = A * I_0(\alpha * R_e) + B * K_0(\alpha * R_e) \\ \frac{\phi_b}{S} = -\lambda * \left(A * I_0'(\alpha * R_e) + B * K_0'(\alpha * R_e)\right) \end{cases}$ Where I_0 and K_0 are respectively a 0 order first kind modified Bessel function, and a 0 order second kind modified Bessel function.
	Conduction element for the Spherical TTRS class	$\begin{cases} T_a = A + \frac{B}{R_i} \\ \frac{Q_a}{S} = \frac{\lambda * B}{{R_i}^2} \end{cases}$ $\begin{cases} T_b = A + \frac{B}{R_e} \\ \frac{Q_b}{S} = \frac{\lambda * B}{{R_e}^2} \end{cases}$

B. *Model transformation and perspectives*

The transformation from the thermal enriched 3D CAD model into the Modelica model (appendix 4) is performed using Python scripts. A predefined Modelica model is associated with each medium selected in the 3D CAD environment. In parallel, the top-level Modelica model, which is related to the whole physical architecture defined in the 3D CAD environment, is built automatically through the links (Interacting Faces) defined by the 3D CAD elements in contact with each thermal medium. For example, a physical contact between two elements implies heat and temperature continuities, and a link (IS) between these elements. The Modelica models associated with the geometry or the media must have been defined in the demonstrator beforehand. For example, a medium associated with convection will be associated with a defined Modelica convection model. Up to now, the simulation teams have not been able to choose the most suitable model from a list of available models. Nevertheless, future work could implement this enhancement, by checking that the proposed Modelica models are related to the selected thermal propagation mode, and that the media inputs/outputs, including boundary conditions, are compatible with the Modelica model. Otherwise the simulation teams must modify the input/output ports of the Modelica model chosen. In future, the TTRS-based convection-coupled conduction modeling elements developed previously (Chapter 4 VI.A.1) could be added in the model transformation rules, notably for sizing tubular radiators. Although, these elements have not yet been implemented due to a time constraint, these models are more accurate than those of the standard Modelica library.

Regarding model transformation management, the addition of other thermal models (whether from a standard library with different details, or that defined by the user) could be interesting for future work.

VI. Conclusions

In this chapter, we presented the Thermal 3D Sketcher, which is the tool developed to demonstrate the capacity of the SAMOS framework proposed to function in an industrial context. The first part of this chapter described the requirements related to selecting the tools required for its implementation. The tools selected were the PTC modeler for the SysML model, FreeCAD for the 3D CAD tool, and OpenModelica for the simulation. The algorithms and the model transformation rules between the different tools were provided. This was then followed by a description of the implementation of the two SysML extensions GERTRUDe and TheReSE in the SysML environment, and of the developments performed on the Graphical User Interface (GUI). The implementation of the thermal 3D Sketcher in the 3D CAD tool was detailed through the SAMOS process (from the import of the SysML architecture until the simulation setup), including the various model transformations (for data exchange between each tool used) and the GUI developed. Finally, the developments in Modelica for the Simulation step was presented for the shell view described in chapter 4.

After having described the mechanisms used for the implementation of the thermal 3D Sketcher, the SAMOS approach and its implementation through the thermal 3D sketcher will be verified in the next chapter.

Chapter 6 – Scenario-based validation

I. System conveyer 183

A. Description of the case study 183

B. Geometry definition in GERTRUDe 184

1. System architects' model 184

2. The developer viewpoint 184

C. Generation of the 3D CAD architecture 187

D. CAD2GERTRUDe transformation 188

II. Helicopter bay 189

A. Case study description 189

1. Geometrical requirements 189

2. Thermal requirements 190

B. Application of SAMOS approach 191

1. System modeling 191

2. Geometrical and thermal analysis 192

C. Modelica modeling 195

D. Verification 198

III. Airplane cab 200

A. Case study description 200

B. Geometrical requirements 201

1. Geometrical requirements modeling 201

2. Modeling related to changes of geometrical requirements 202

C. Thermal behavior traceability through TheReSE 204

IV. Electric Power Train 207

A. Geometry modeling 207

B. Thermal requirements 208

C. Geometrical modeling 209

D. Thermal modeling 212

V. SAMOS verification case study: electric bicycle 215

A. Case study description 215
1. Geometry requirements 215
2. Thermal specifications 216
B. Geometry modeling 217
C. Survey results 219
VI. Verification and validation of the SAMOS and thermal 3D sketcher 221
VII. Conclusion 222

The description of the thermal 3D sketcher including the model transformation and the development of the corresponding demonstrator (tool) has been developed in the previous chapters. The validation of the research issue addressed here is performed through the implementation of five case studies, including four industrial ones (system conveyer, helicopter bay, aircraft cabin, electric power train), and an academic scenario: an electric bicycle (developed to support an engineering training course. They focus on different parts of the thermal 3D Sketcher. The System conveyer case study is aimed at verifying the geometrical model. The aircraft cabin case study presents a description of the TheReSE model. The helicopter bay and the electrical power train for an electrical bus serve to verify the thermal requirements for several alternative 3D architectures. The last case study consists in testing the Thermal 3D sketcher during an engineering training course to help students select the best physical architecture for an electric bicycle, based on the thermal simulation results.

I. System conveyer

A. Description of the case study

The first approach was experimented in an industrial case study, i.e. a modular conveyor system. The aim of this conveyor is to move parts that have just been hot molded. The geometrical requirements are the following (Figure 105): the length of the rollers must be equal to 3.5 m; the diameter of the rollers and the distance between them must be dimensioned to carry a parallelepiped block measuring 400*400*2500 mm^3 composed of glowing hot steel block.

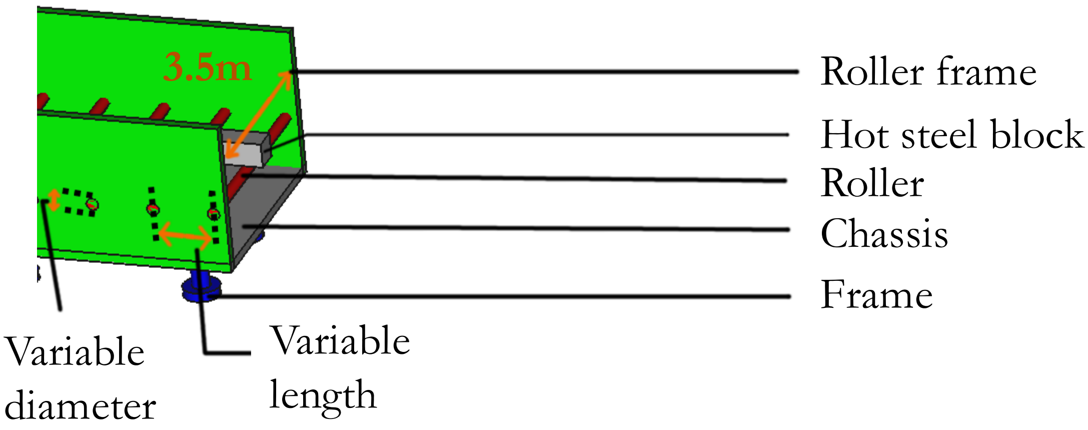


Figure 105 : Conveyor case study description

B. *Geometry definition in GERTRUDe*

1. System architects' model

First, the System architects have to specify the geometry of the conveyor with approximate dimensions, provided, for example, by a previous design experiment or specific requirements, but they do not need to specify the radius and the number of rollers.

Starting from a physical architecture defined in a block definition diagram in SysML, where the names and the multiplicity of the conveyor elements are specified (Figure 106), the System architects add the geometrical specifications of the components through GERTRUDe.

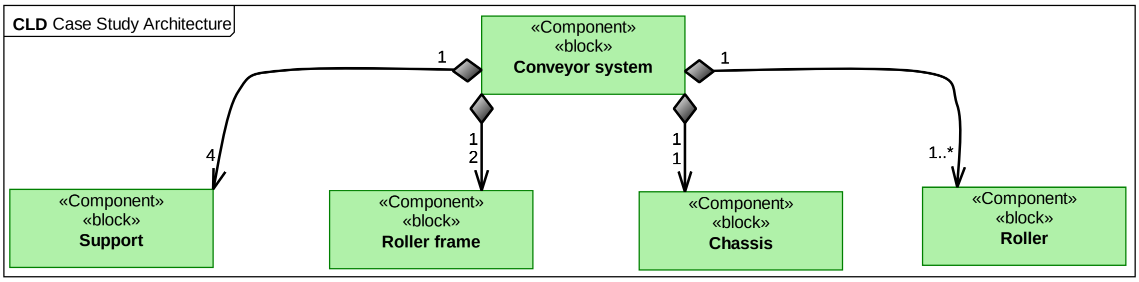


Figure 106 : Initial physical architecture of the conveyor

Roller will be approximated as a finite cylinder. *Roller support* and *Chassis* are rectangle parallelepipeds. Support can be either approximated as a finite cylinder, or composed of two finite cylinders. A library of simple components was developed in GERTRUDe profile, in order to aid the System architects to quickly assign a geometry to the components (Figure 107).

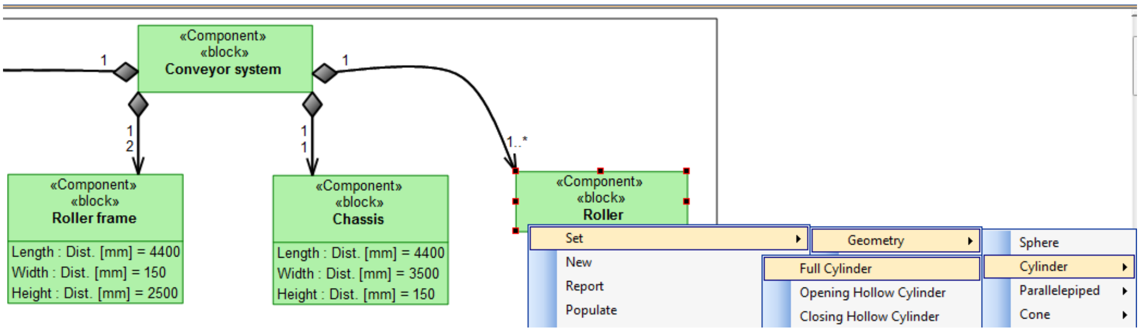


Figure 107 : Geometrical enrichment of physical architecture by GERTRUDe

2. The developer viewpoint

However, if the System architects have to specify a more complex geometry, for example, a double cylinder geometry for the frame, GERTRUDe makes it possible to build it by the composition of existing TTRSs, as described hereafter.

Assuming that the geometry of the *Frame* is defined by the association of two finite cylinder TTRSs, Head Cylinder TTRS and Foot Cylinder TTRS (Figure 108), these resulting TTRS belong to the revolute TTRS class and have been defined in the GERTRUDe Geometry library.

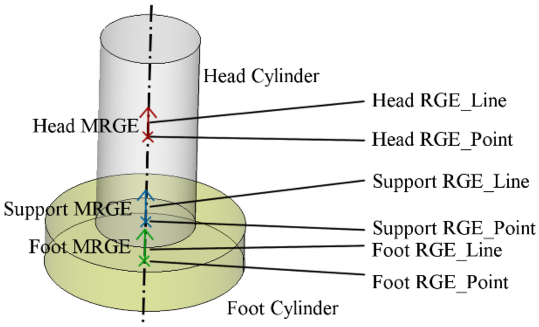


Figure 108: Construction of the double cylinder support geometry based on TTRS modeling.

To assemble these two cylinders, they have to be relatively positioned through constraints between their MRGEs (for a finite cylinder TTRS, its MRGE is composed of a line for the extrusion (length) direction and a point at half the cylinder's length). The first constraint is a coincidence between both cylinder RGE_lines. The second is the distance between the two cylinder RGE_points.

Finally, the placement of the MGRE of the whole double cylinder support is ensured by the two similar TTRS constraints (C11 coincidence constraint, and C02 distance constraint) applied on its revolute MRGE. Figure 109 explains the relations in the construction of such a new geometry.

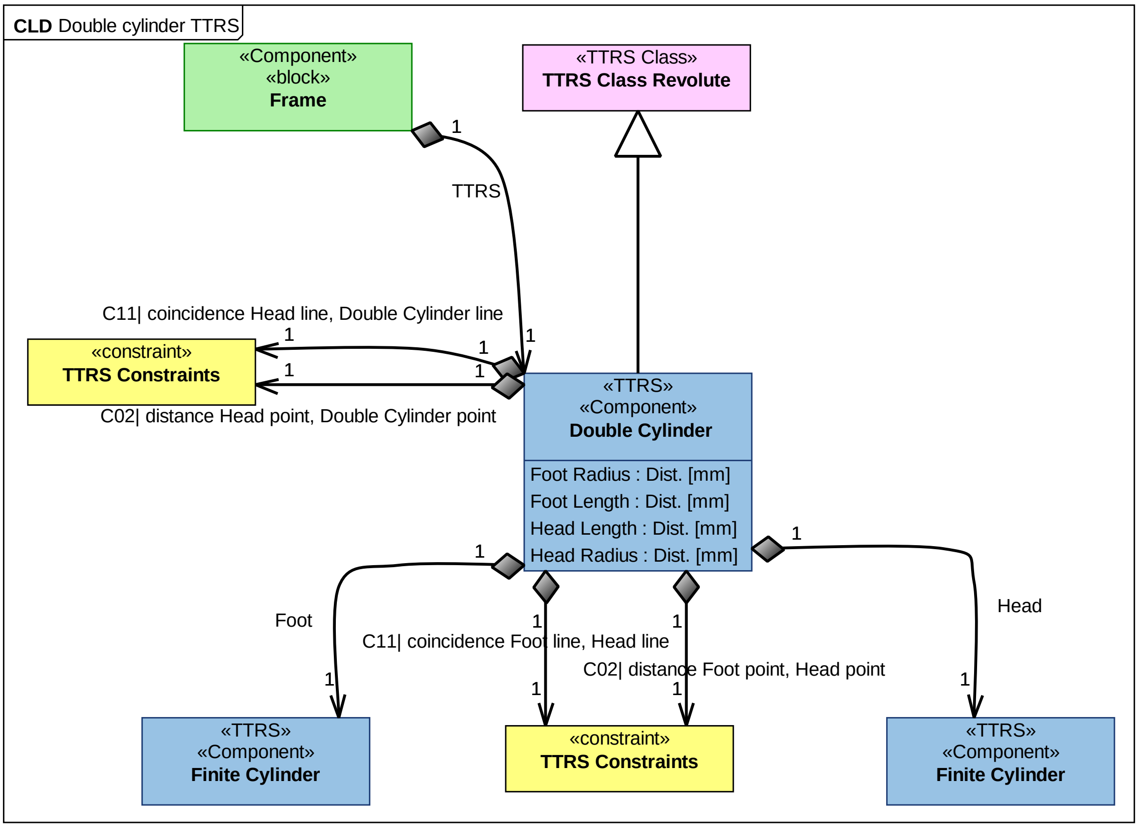


Figure 109: Specification of relative positioning constraints to build the support double cylinder TTRS.

The detailed parametric diagram of the implementation of these constraints is presented in an AssemblyParam diagram (Figure 110). Equality constraints have been added to ensure equality between intrinsic parameters (like radius or length) received from the different entities.

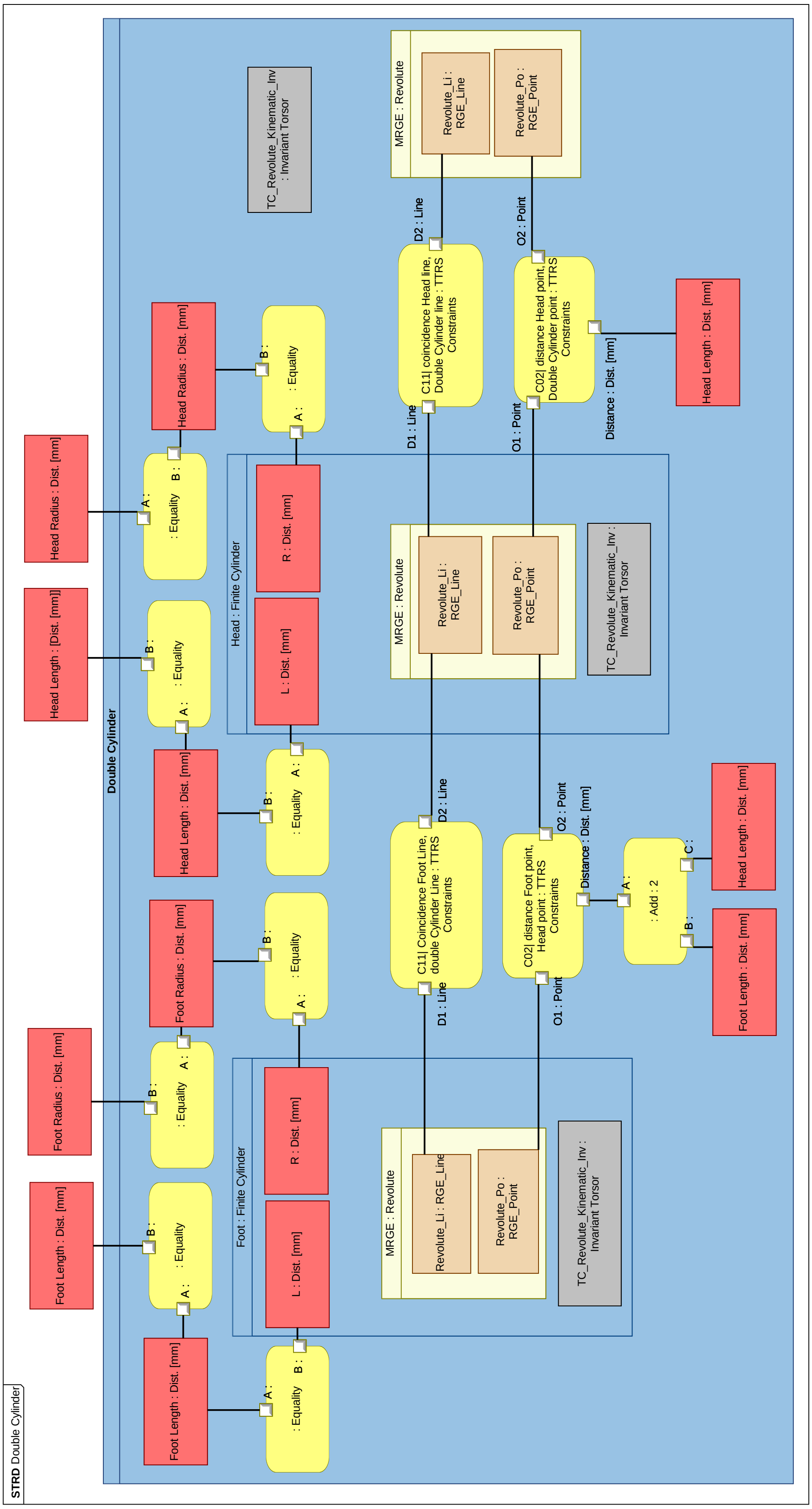


Figure 110: AssemblyParam diagram describing the TTRS model of the double cylinder.

The previous modeling task requires good spatial vision and notions of the TTRS theory, and will not usually be performed by the System architects; however, trained engineers could easily develop any new geometry based on the proposed SysML extension. The System architects will rather use GERTRUDe to allocate predefined geometries, their dimensional parameters and geometrical constraints to the components when they need to specify them.

C. Generation of the 3D CAD architecture

The GERTRUDe SysML-FreeCAD transformation can be launched from the FreeCAD environment.

As in this example, the radius of the *Roller* component is not specified, a default parameter value equal to 1 is set and the color of the corresponding 3D component generated will be orange, whereas the color of the components with all their completed values will be gray (Figure 111).

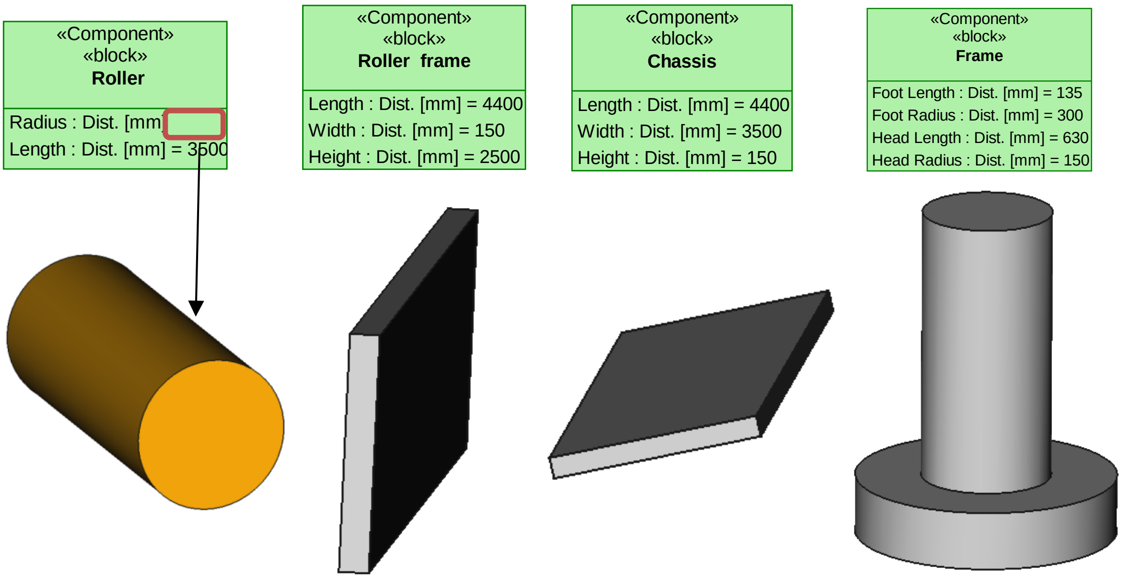


Figure 111: Generation of 3D conveyor components from GERTRUDe components.

Then, the 3D architects can modify and assemble these components and add missing data or constraints in the FreeCAD tool environment, in order to propose a 3D spatial architecture.

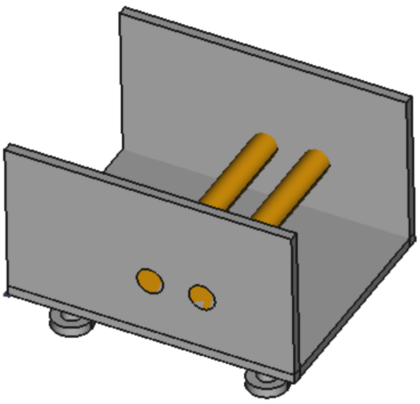


Figure 112: Initial 3D spatial architecture of the conveyor system.

For example, the conveyor has been defined with 5 rollers with a 100mm radius, and a new component has been added (Figure 113), compared to the initial architecture (Figure 112): a top casing is required for safety considerations.

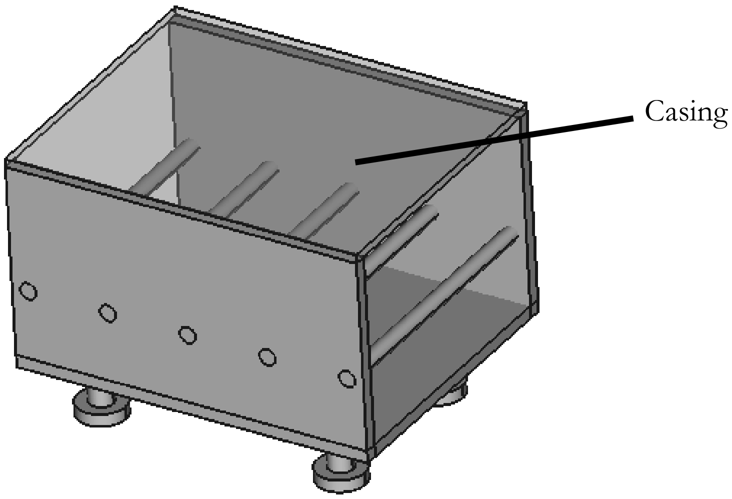


Figure 113: Proposed 3D spatial architecture in FreeCAD tool.

D. *CAD2GERTRUDe transformation*

Although this transformation has not yet been implemented in the demonstrator, this case study proposes an example of a modification of new CAD items traced back to GERTRUDe. The expected result of the CAD2GERTRUDe transformation is presented in Figure 114. The modified or added components are displayed in yellow, in order to help the system architects visualize which components have been modified by the 3D architects. In this example, the radius and the number of the Roller components, as well as the Casing component have been added.

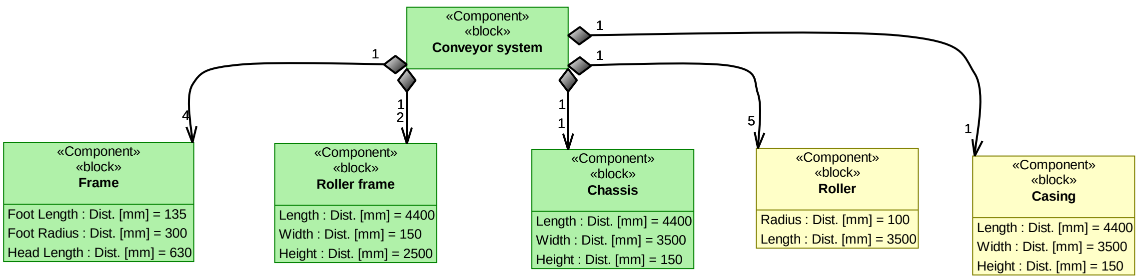


Figure 114: Expected updated physical architecture traced back in the GERTRUDe extension.

Finally, the whole architecture is traced back in the GERTRUDe-enriched SysML model thanks to the model transformation. This architecture also contains information on the geometrical constraints and the position of the components. For each assembly of parts, a stereotyped AssemblyParam (STRD) diagram represents the geometrical constraints between these components. Figure 115 illustrates the contact constraint between the right roller support and roller 2. This contact constraint is represented by a distance constraint, with the parameter distance equal to "null".

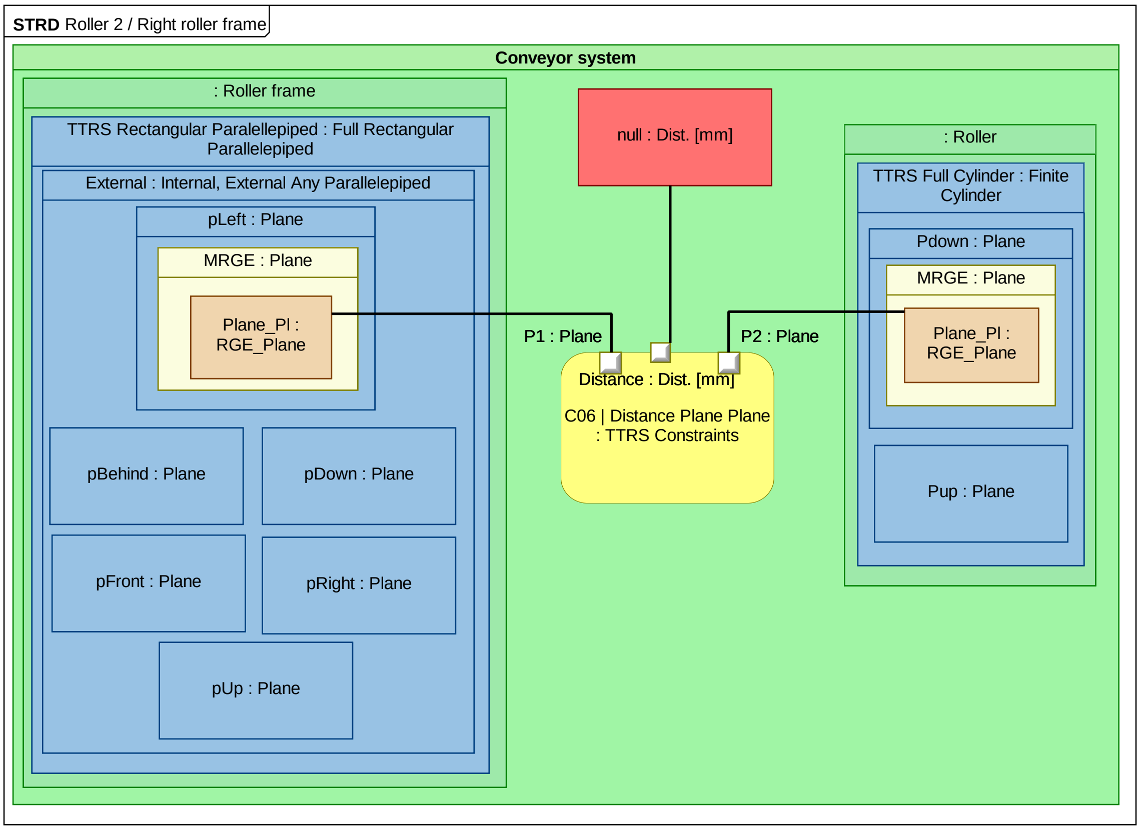


Figure 115: Example of the contact constraint between the right roller support and roller 2.

II. Helicopter bay

The case study of the helicopter bay is a complete industrial case study, extracted from the TOICA (Thermal Overall Integrated Conception of Aircraft) H2020 project. It includes system, geometry and thermal modeling. The results of the thermal analysis will be compared to a finite element analysis in order to estimate the error between the simulation results provided by the thermal analysis in the thermal 3D sketcher. The helicopter bay may be referred to as aircraft bay or avionic bay in the following paragraphs.

A. Case study description

The goal of this case study is to evaluate the thermal behavior of an avionic bay architecture.

1. Geometrical requirements

Four devices are positioned at different altitudes. Each item of equipment is considered as a rectangular parallelepiped. The 3D architecture of the avionic bay is considered as given in Figure 116.

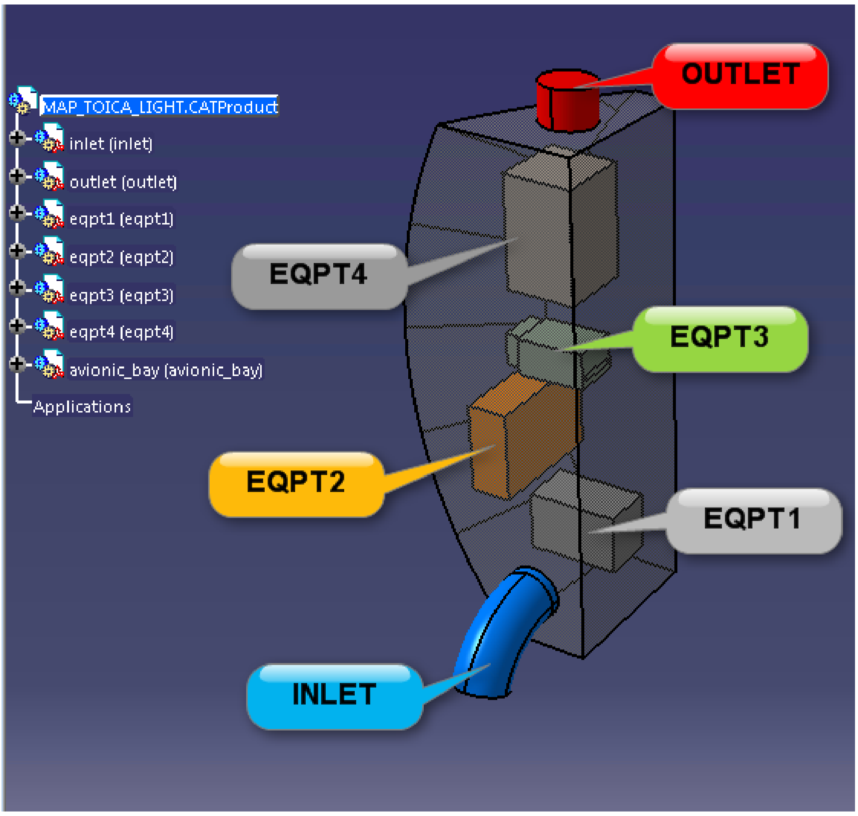


Figure 116: Avionic bay 3D architecture.

2. Thermal requirements

The main thermal mode addressed is convection, as conduction and radiation have been considered as secondary. The equipment dissipates heat flows as described in Table 48.

Table 48: Equipment alternative boundary conditions

Item of equipment	Thermal dissipation Heat flux (W.m^{-2})	Ambient temperature specification(*)
EQPT1	320	70°C continuous
EQPT2	329	70°C continuous
EQPT3	1260	70°C continuous
EQPT4	302	70°C continuous

* Ambient air temperature considered around the equipment.

Assuming that an item of equipment dissipates a constant power, we have considered that the heat flux (in W/m2) is uniformly dissipated through the faces of the equipment.

Concerning the ventilation modeling, the following configuration is considered:

- Inlet : Air temperature = 55°C / P=Patm
- Outlet : mass flow rate = 100 g.s-1

Finally, the helicopter bay is assumed to be adiabatic.

B. *Application of SAMOS approach*

1. System modeling

The system modeling in SysML has been performed using the GERTRUDe and TheReSE extensions. Figure 117 illustrates the geometry allocation to each architecture of an item of equipment through the GERTRUDe GUI. This GUI was developed in order to facilitate using a geometry library for the Systems architects. On this example, a rectangular parallelepiped geometry is applied to the 4th item of equipment.

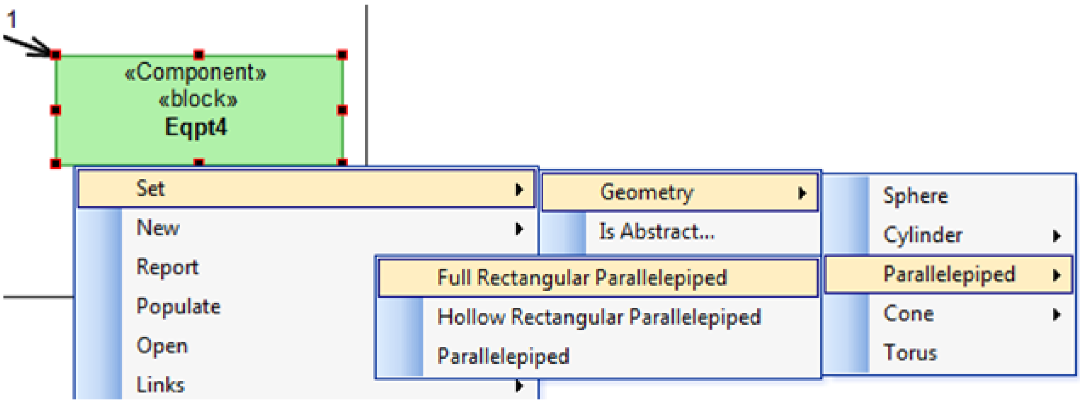


Figure 117: Definition of the geometry of each component with GERTRUDe GUI.

The final architecture model is provided in Figure 118, where each item of equipment has a TTRS-based geometry and its associated intrinsic dimension parameters. The fluid medium will be defined in the 3D CAD environment.

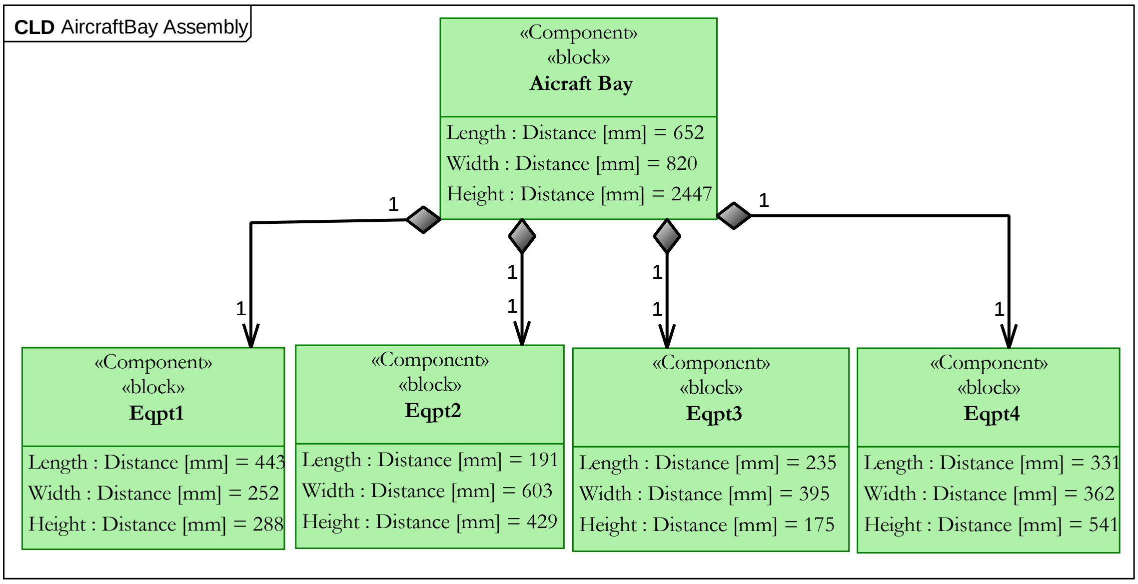


Figure 118: Results of geometrical definition of architecture components with GERTRUDe.

Once done, it is then possible to perform the GERTRUDE2CAD model transformation, in order to transform the architecture provided by the System Architects into a 3D model in the FreeCAD environment, enabling the further geometrical and thermal analysis steps.

2. Geometrical and thermal analysis

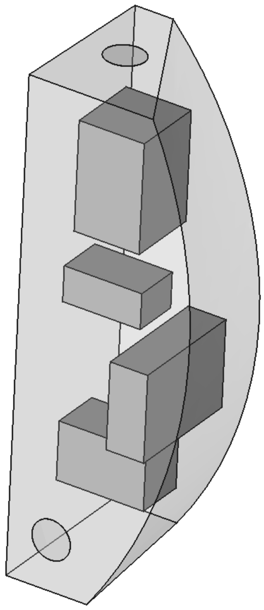


Figure 119: 3D architecture definition.

First the 3D architects (i.e. space allocator) have to complete the 3D architecture for the Aircraft bay. Although the geometry can be complex, it has to be simplified in order to support the thermal analysis. Once the geometry of the bay has been created, the 3D architects position each component in relation to each other either with assembly constraints (not yet developed) or through its coordinates, either defined manually or selected by mouse click. Figure 119 shows the final 3D architecture of the helicopter bay.

Thermal specifications are then added by the thermal experts. They first select a material for each component. As described in chapter 5 IV.B.2, it is not compulsory to specify the material of the components used for the solid elements when considering them as simple components (and not media) for convection analysis with a shell view. However, for fluid elements, the specification of the material is required (for the Nusselt correlation). Figure 120 illustrates the application of the "air" material to the helicopter bay fluid element, as all the equipment is located in air.

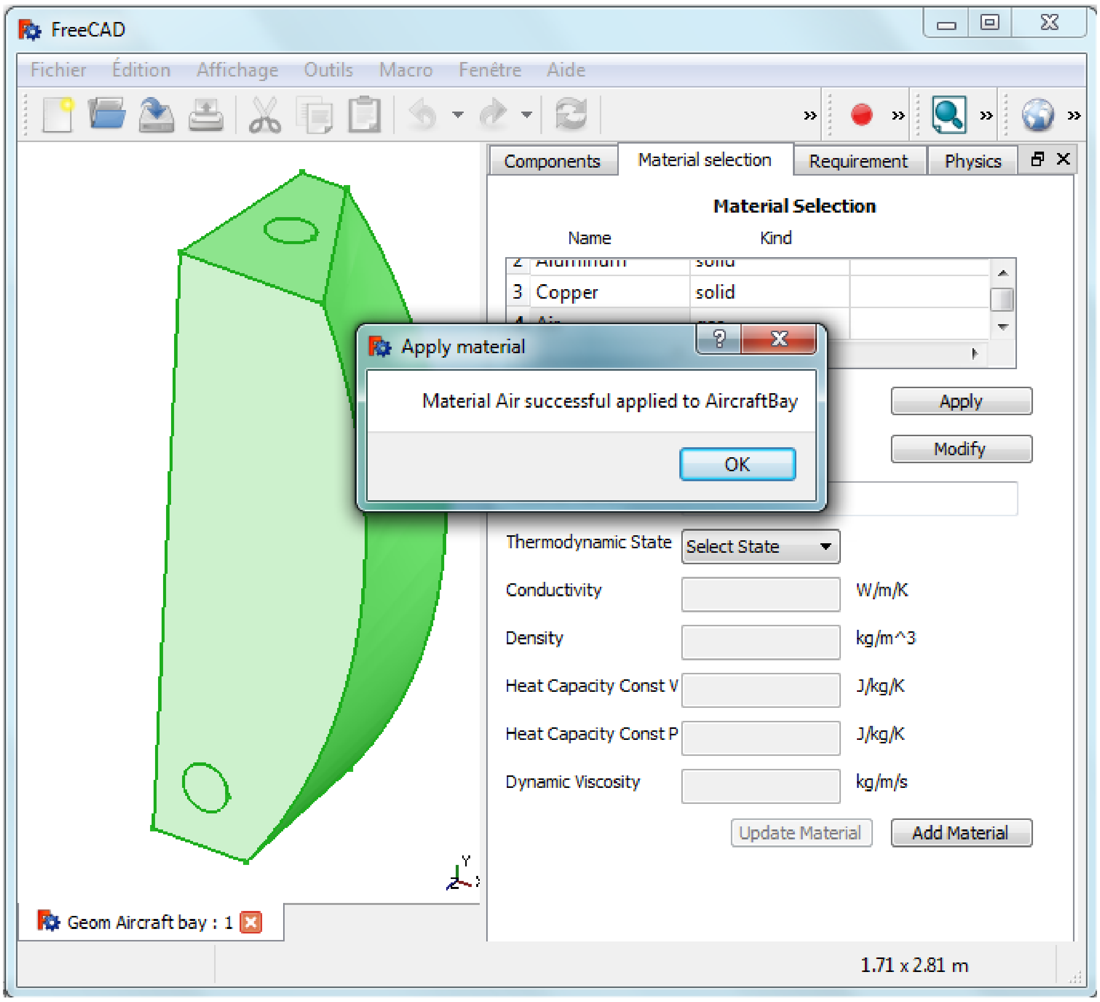


Figure 120: Material selection for each component.

If the requirements provided by the System architects have to be updated, the thermal experts can add or modify them. In our example, a new requirement related to the maximum temperature of the air (which must be below 70°C) has been added as a quantified requirement, as described in Figure 121.

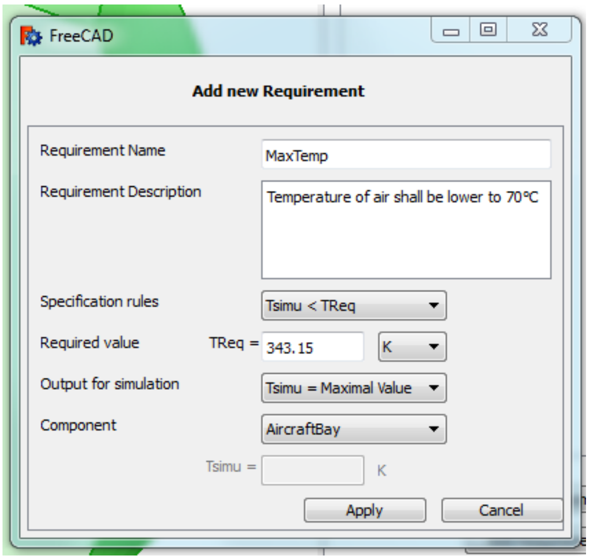


Figure 121: Adding a new requirement.

Next, the simulation teams can also add the boundary conditions to each item of equipment. For the fluid element, the associated boundary conditions are described in Figure 122. After setting the Inlet, the Outlet and the boundary conditions, the fluid element is cut into slices. Finally, each fluid slice generates a Modelica class through a model transformation.

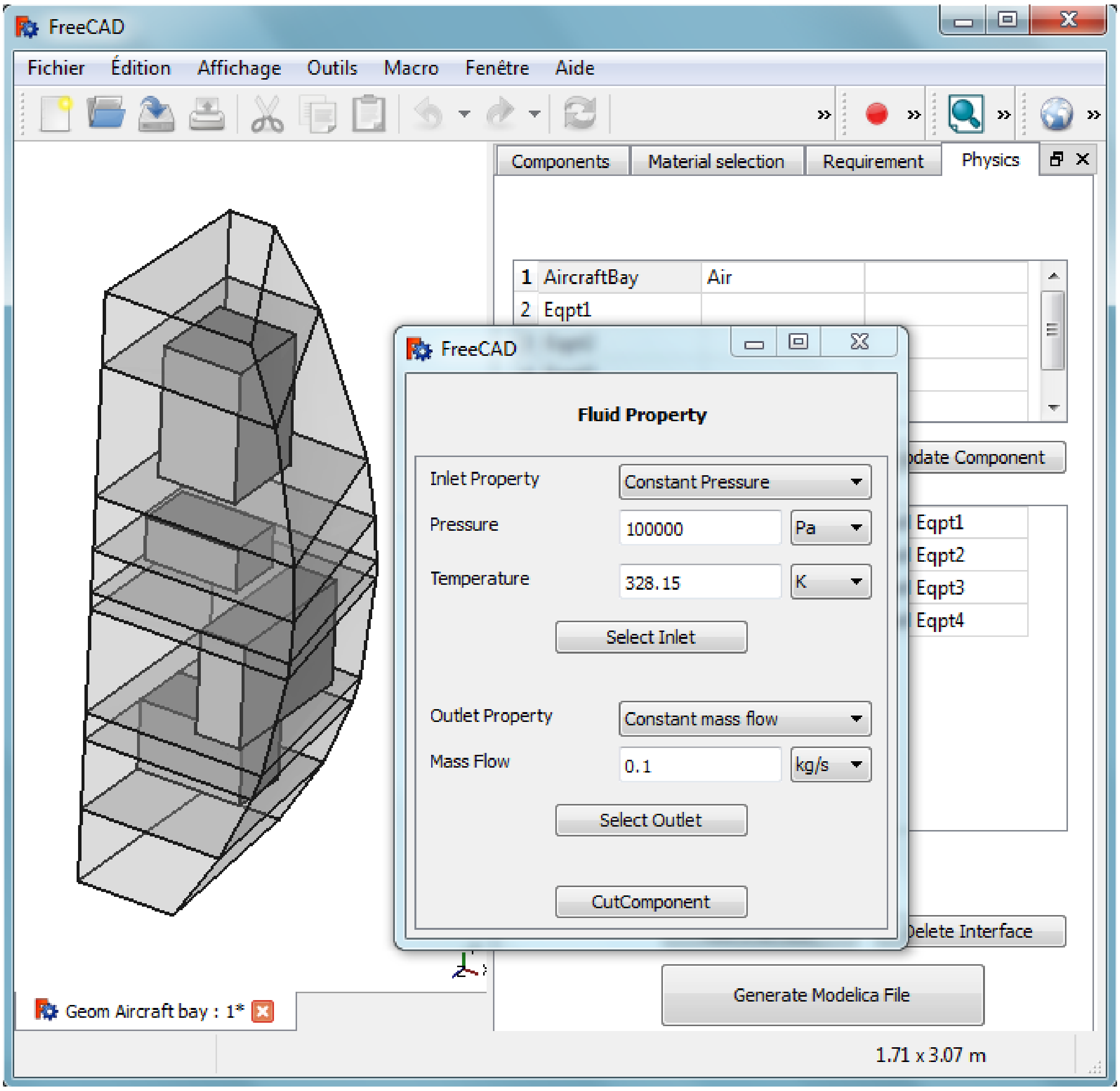


Figure 122: Adding boundary conditions

The last task of the simulation teams is to define the thermal interactions between the components. For example, in the case study, all the media considered are the convection media, as we assume the radiation effect and the conduction behavior of the racks to be negligible. Figure 123 shows the FreeCAD window that the thermal experts have to fill when a convection medium is added when using Nusselt correlation analysis.

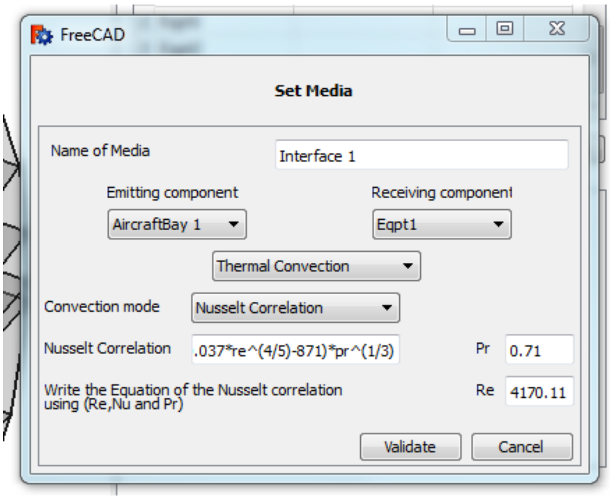


Figure 123: Media definition.

Once all the media have been added, it is possible to perform the model transformation in order to generate the Modelica model from the 3D geometrical model enriched with thermal specifications.

C. Modelica modeling

The Modelica model generated by the transformation is provided in Figure 124. The links between the components and the media have been added manually to improve readability. The automatic representation of these links would require a routing algorithm that has not been implemented in the demonstrator.

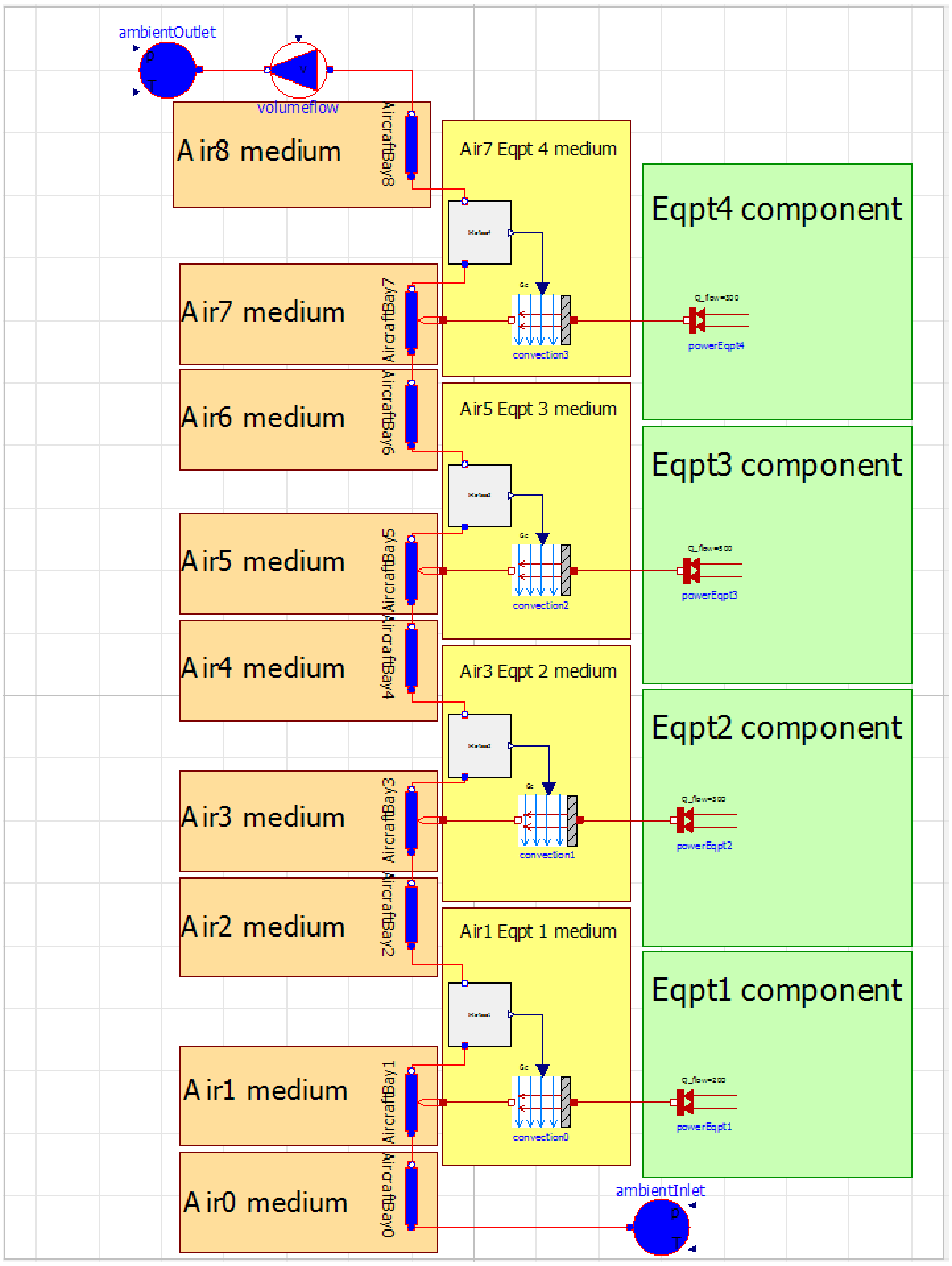


Figure 124: The resulting Modelica model generated.

In this model, all the equipment has been modeled with the *FixedHeatFlow* Modelica model, as their materials are not specified (equipment considered as component (not medium)). All the cut fluid parts have been modeled with *HeatedPipe* and *IsolatedPipe* models. The convection modeling is ensured by the *Nusselt* and *Convection* models as described in chapter 4 IV.B.2.a .

The simulation is then launched automatically using the OMPython library, and an automatic comparison of the simulation results regarding the requirements is performed.

Fulfilled requirements appear in green, otherwise in red if the results do not fit the thermal requirements.

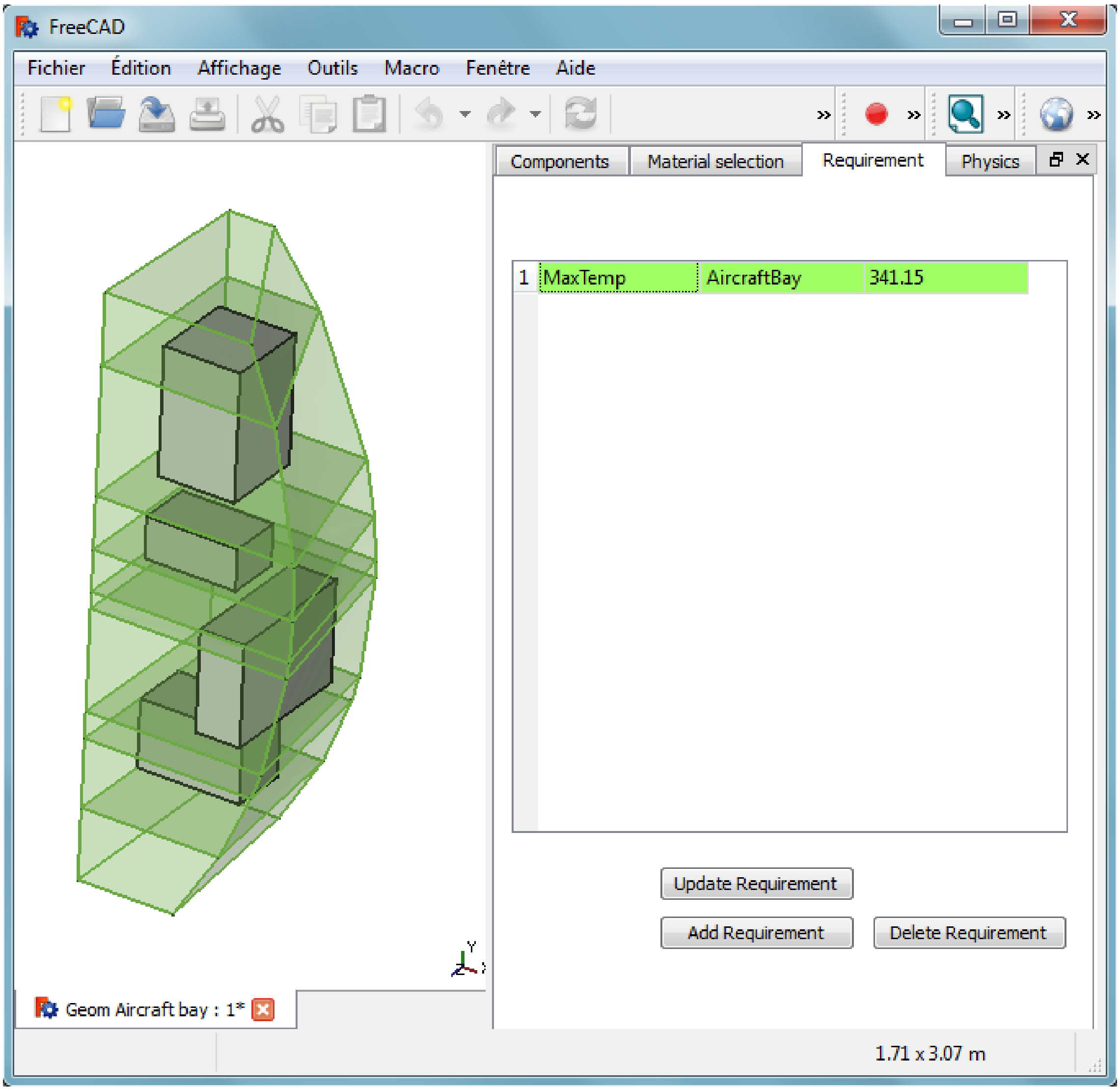


Figure 125: Automatic verification of simulation results regarding the requirements.

Next, it is also possible to visualize the components or fluid temperature curves as a function of time (Figure 126). For example, we can see the curve of the maximum temperature of the fluid (air).

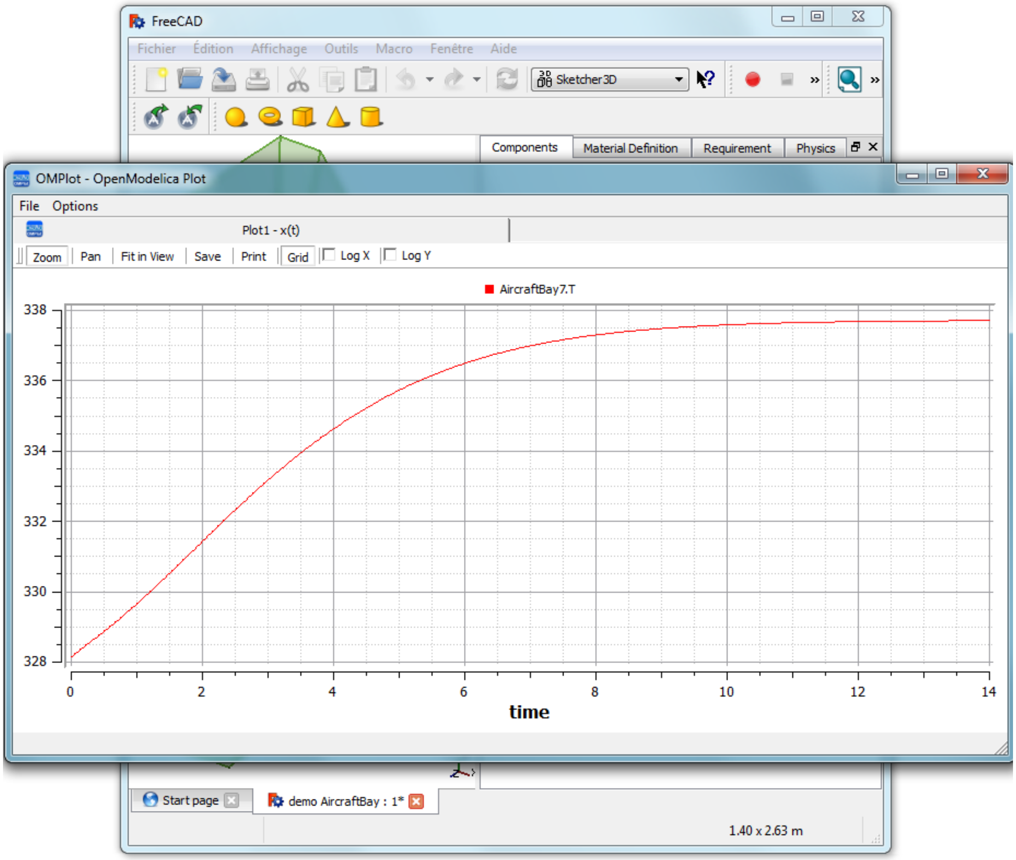


Figure 126: Resulting plotted curve of the maximum outlet air temperature.

D. Verification

In order to verify the simulation results, we compare them with a Finite Element Method model using the ANSYS workbench software and the CFX calculator. The corresponding results are displayed in Figure 127. The fluid analysis shows the fluid displacement in the helicopter bay, whereas the thermal analysis presents the fluid temperature map.

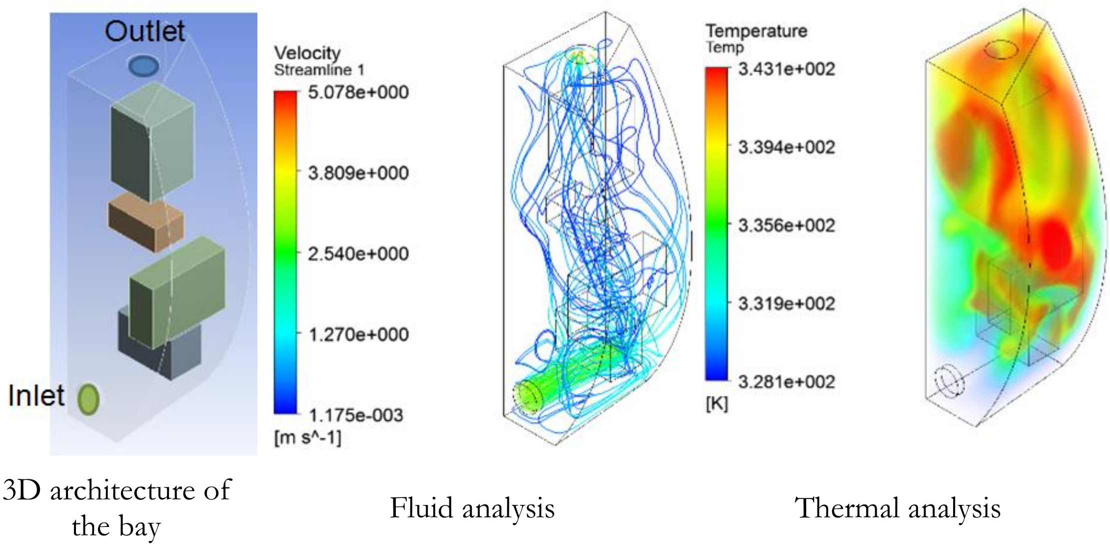


Figure 127: ANSYS CFX simulation results on the 3D architecture of the bay.

In order to compare relevant information between the Modelica model and ANSYS CFX model, we cut the ANSYS fluid part into different altitudes (corresponding to the Modelica sections). The average temperature in each cut section has been calculated. This result is provided in Table 49 and the corresponding curve according to the altitude considered is plotted in Figure 128.

Table 49: Comparison of temperatures according to the section altitude.

y (m) (altitude of the cut plane)	Average Temperature (K) ANSYS CFX Model	Average Temperature (K) Modelica Model
0.60	334.55	330.1
1.14	344.75	333.19
1.38	344.27	338.14
2.15	343.69	341.19

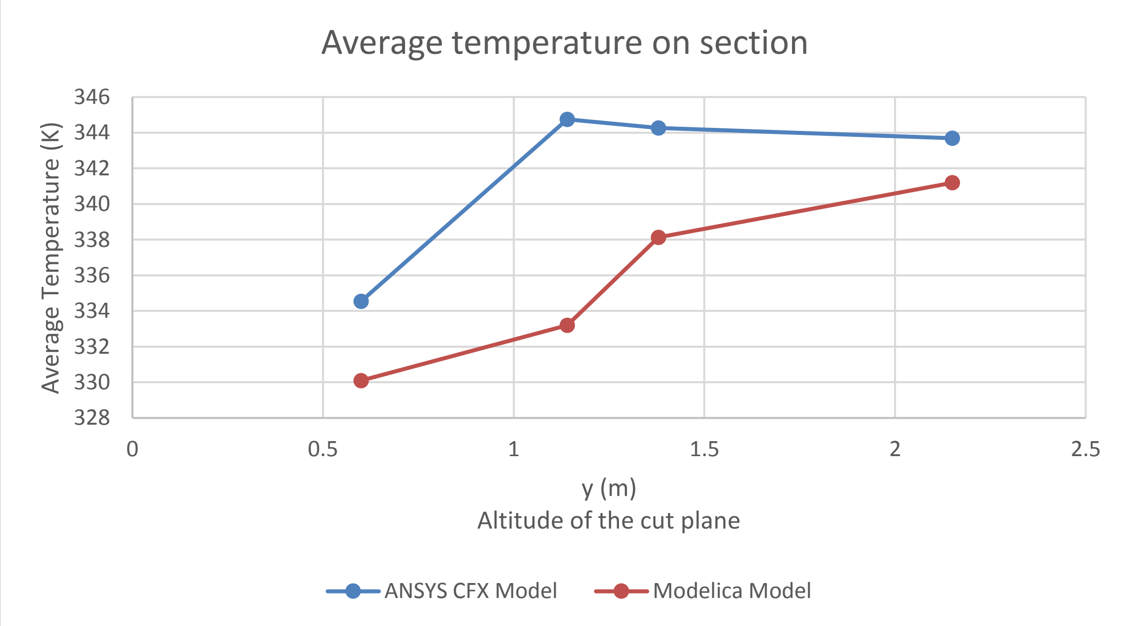


Figure 128: Comparison results of the temperature according to the cut plane altitude.

Then, although the Modelica and ANSYS CFX results are different, the maximum gap (12K) can be considered acceptable for the conceptual design phase we addressed. However, the thermal model chosen could be improved in order to obtain better matching. Moreover, an uncertainties analysis should complete the simulation results, in order to obtain a confidence interval. The simulation results should be traced back to the SysML model (not yet implemented in the demonstrator).

III. Airplane cab

In order to describe the traceability of the enriched 3D thermal data in the System model (SysML model), and particularly with TheReSE, another industrial case study was designed. The objective of this Aircraft cab case study is to describe a thermal model with TheReSE.

A. Case study description

The goal of this case study is to verify the positioning of airplane calculators under thermal constraints in the case where a new calculator is required. In order to perform this study, we have considered the most thermally critical section of an A320 Airbus airplane of an upmarket airline company, i.e. the first class section. This section (i.e. the first row of seats) is composed of four passenger seats with integrated tablet screens. Calculators one and two, used for general airplane management, are already located under the floor of the aircraft (Figure 129). In order to display information on these screens, an additional calculator is required, calculator 3. In this example, we have considered only the thermal convection propagation mode, not including the floor, for which we have also considered conduction. The radiation is negligible, since the component temperatures must be lower than 50°C. Aluminum has been imposed as the calculator material.

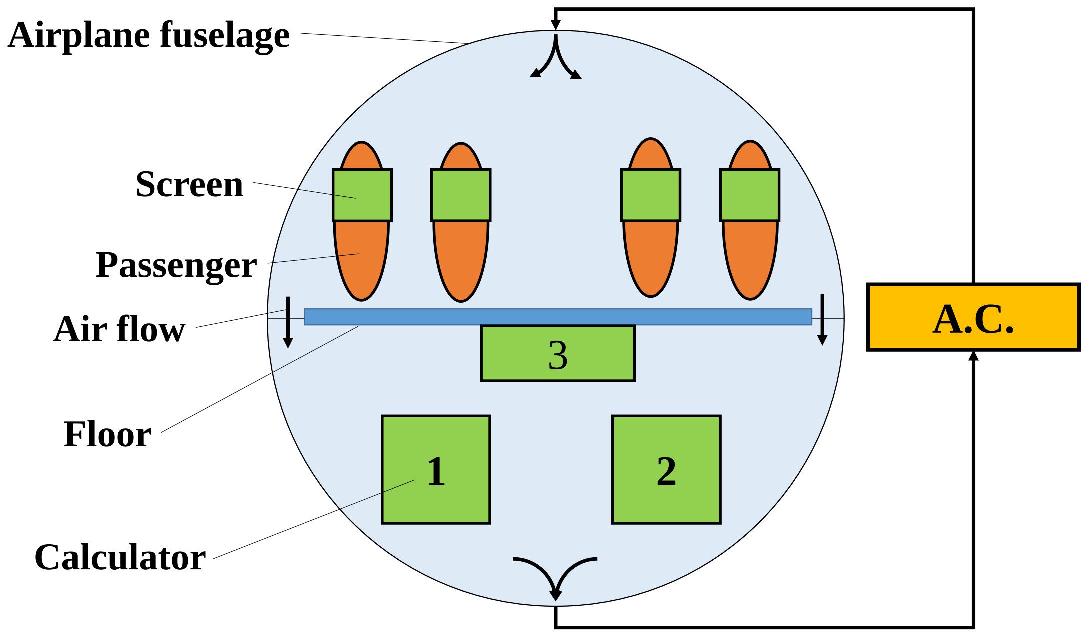


Figure 129: Air Conditioning system (A.C.) model in the airplane section considered.

The description of the thermal related requirements is given in Table 50. The initial temperature of both the air and all the components is 20°C, except for the passengers that have a constant temperature of 21°C. The components with adiabatic behavior have been considered for thermal simulation since their geometry will influence the convection flows (they have been considered as simple components (with a shell view) and not as a medium).

Table 50: Description of component requirements.

Components	**Approximated geometry**	**Thermal behavior**
Airplane fuselage	Hollow cylinder	Adiabatic
Screen	Rectangle parallelepiped	Adiabatic
Passenger	Ellipsoid	Dressed human, apparent constant temperature: 21°C
Floor	Rectangular parallelepiped	Additional conduction effect
Calculators 1,2	Rectangular parallelepiped	Rate of heat flow emitted: 500 W
Calculator 3	Rectangular parallelepiped	Rate of heat flow emitted: 450 W

B. *Geometrical requirements*

1. Geometrical requirements modeling

This case study presents how a quantified requirement can be modeled in SysML. We focus on a geometrical requirement in the Airplane cabin case study. Indeed, although the System model of the aircraft cab is finished, it can be enriched with geometrical requirements (through GERTRUDe). A quantified requirement can be declared thanks to the new requirement stereotype called "*Quantified requirement*". Figure 130 gives an example of such a requirement by specifying that the distance between the calculator and the floor must not exceed 250 mm.

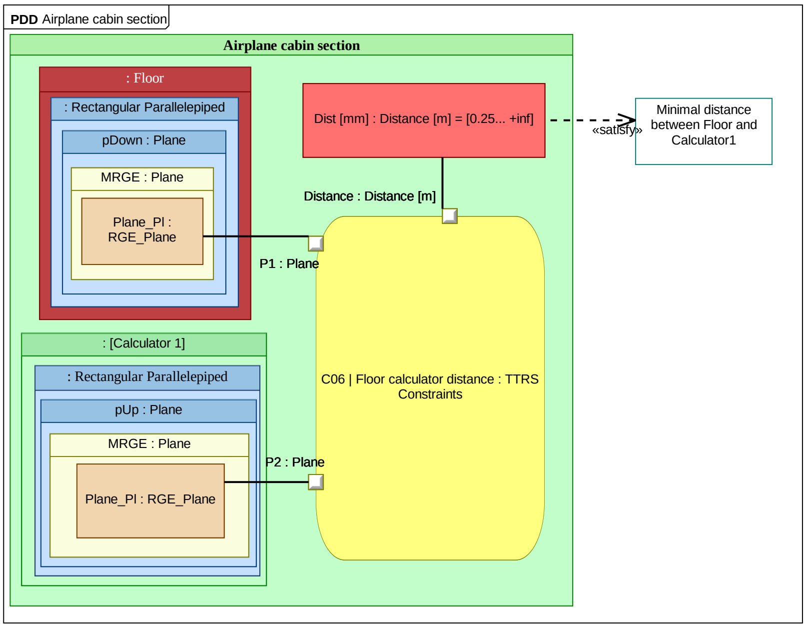


Figure 130: Model of the C06 positioning constraint between the Floor and Calculator 1 with GERTRUDe.

Now that several 3D specifications have been created, the allocation of geometry can be performed.

2. Modeling related to changes of geometrical requirements

The capitalization of both the system and simulation models achieved by this MBSE approach makes it easy to modify requirements and evaluate their impact on the final results. Likewise, if necessary, the model can be refined to take into account more components (like the luggage storage in our example, Figure 131).

Let us consider, for example, that for the previous example of the airplane section model, a low-cost company asks for more compact seat architecture, i.e. an economy class architecture composed of six passengers and only two calculators. The 3D architecture changes are presented in Figure 131. Moreover, in this figure, we can see that the 3D architects have also modified the 3D architecture, by adding new luggage storages.

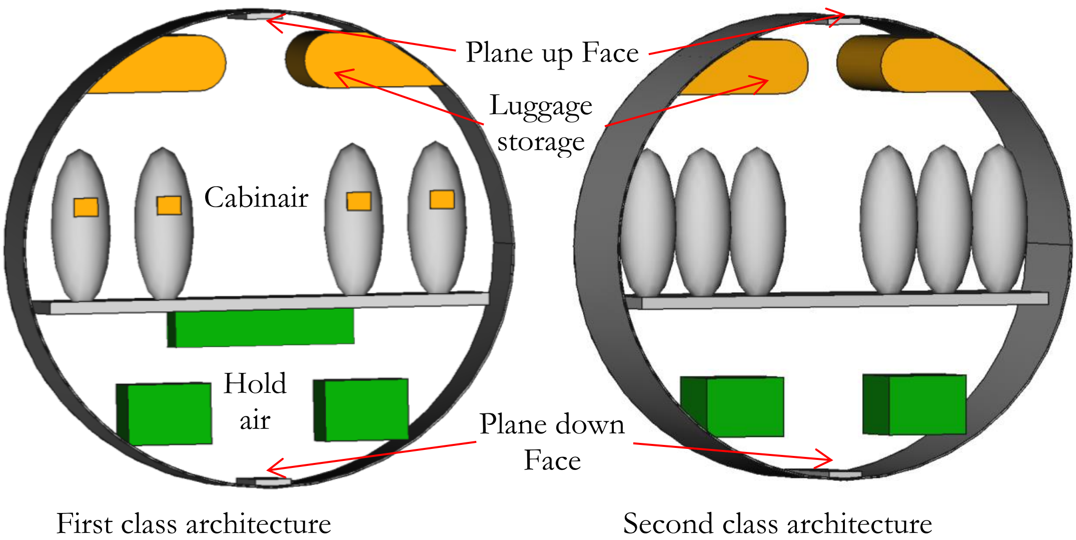


Figure 131: Evolution of 3D architecture according to change of requirements and addition.

This 3D view is undoubtedly more convenient to the 3D architects and simulation teams to make their design changes. In parallel, the traceability of these changes in the System model will ensure the global consistency of the whole conceptual design phase (Figure 132). For reasons of confidentiality, no dimension is provided.

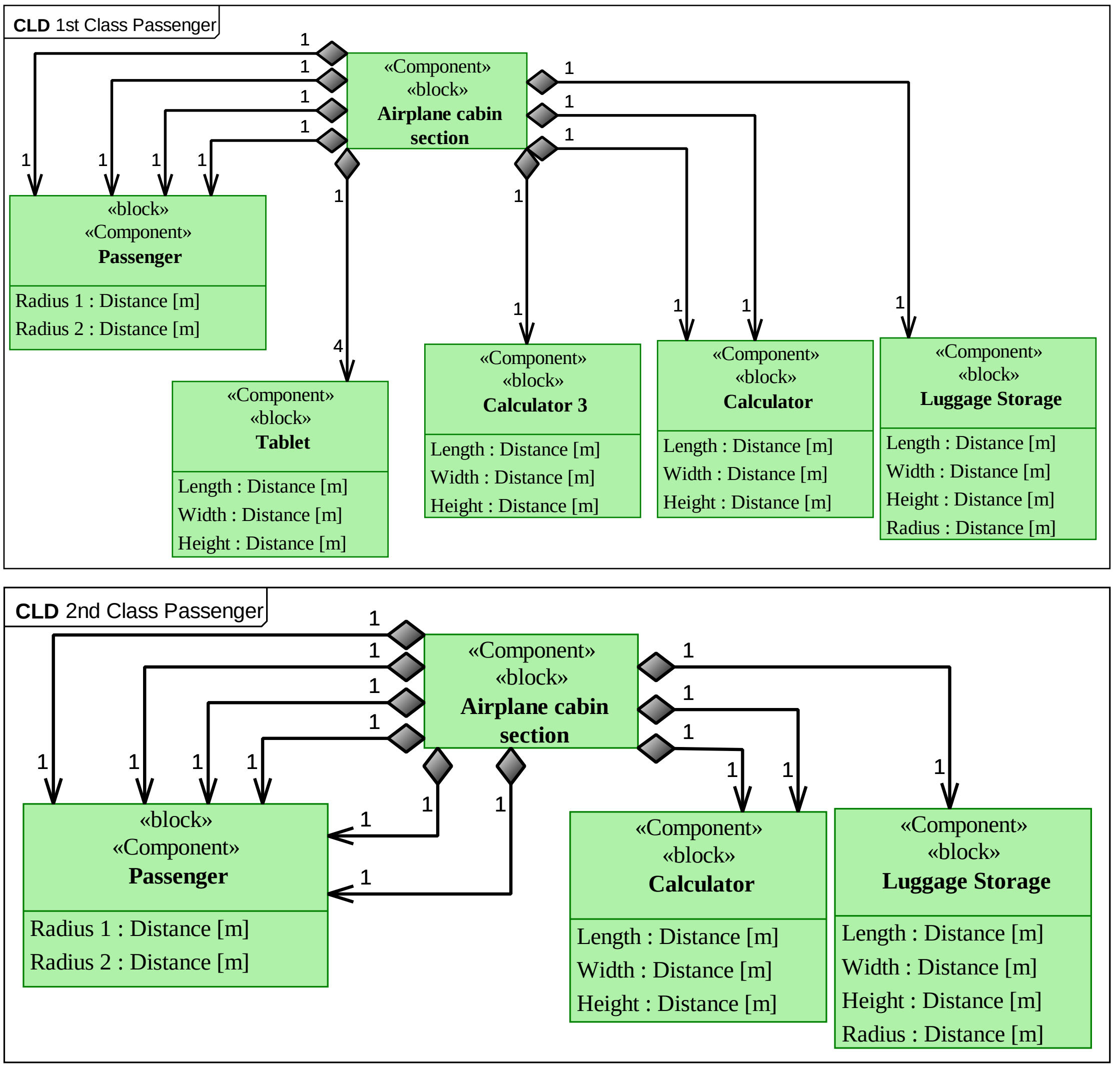


Figure 132 Assembly diagram of first and second classes.

C. Thermal behavior traceability through TheReSE

Once the thermal analysis has been performed, all the information has to be traced back in the System model in SysML.

When starting with the requirements diagram in our example, two additional requirements (defined in the FreeCAD environment of the thermal 3D sketcher) address the acceptable temperature range of the components and the air respectively (Figure 133). They have to be added to the initial requirement from which they derive.

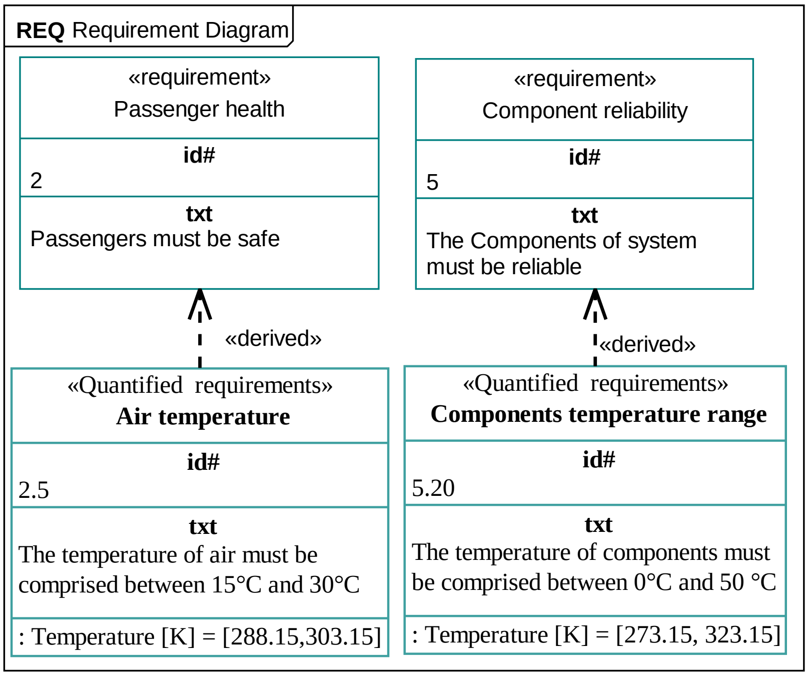


Figure 133: Extract of some additional thermal requirements traceability.

In addition, information on thermal properties (from the "*Physical properties Table*" stereotype) and thermal simulation conditions (such as boundary condition) (from the "*Physical solicitations*" stereotype) can either be specified (from quantified requirements) or traced (from simulation results). The example of the thermal convection model of Calculator 1 is given in Figure 134. This diagram is built on the basis of a *Physics Diagram*, Chapter 5 III.B.1 . In this example, we consider that the simulation has already been performed, and all the results have to be traced back in TheReSE. As described in the conveyer system model, although this aspect has not yet been implemented in the thermal 3D Sketcher, the description of the information expected to be traced back in TheReSE is given for the example of Calculator 1 in Figure 134. For reasons of confidentiality, no dimension is provided

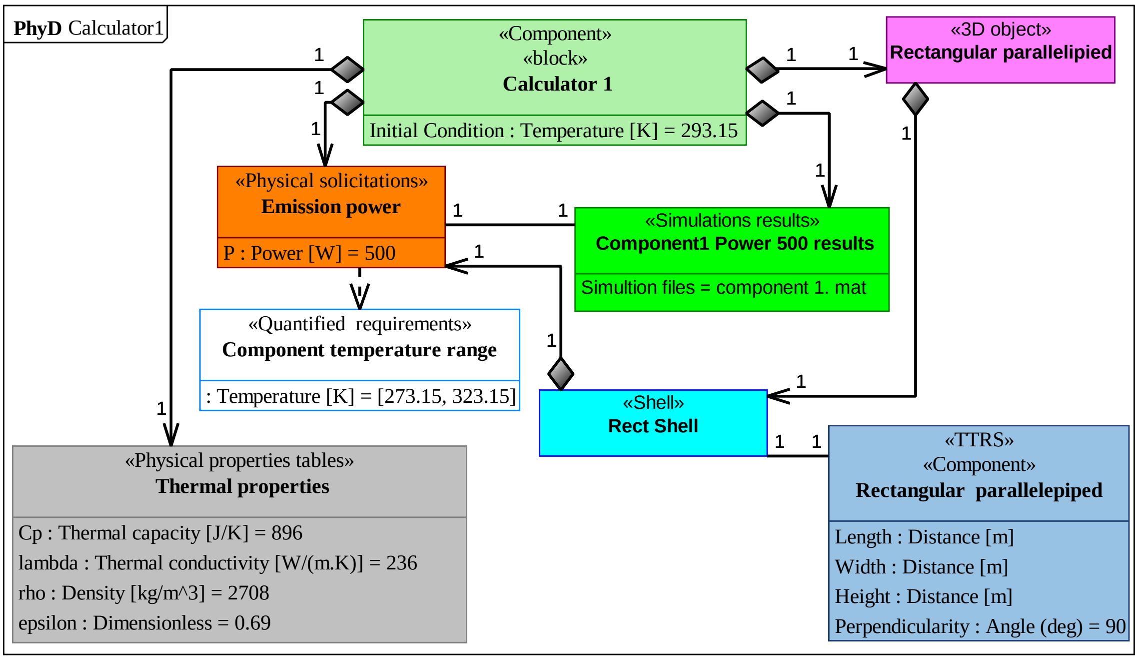


Figure 134: Example of traceability of the thermal convection model of Calculator 1.

Moreover, TheReSE can also be used to formalize the thermal interactions defined in the FreeCAD environment of the thermal 3D sketcher. These interactions modeled using the instances of *Medium* specified between the components, are based on physical and relative positioning constraints. This representation is given using the custom diagram called *Interaction Diagram* (stereotyped diagram derived from the composite structure diagram). The simulated thermal behavior takes into account conduction and convection media for the airplane cab example described in Figure 135. The red and blue ports represent the faces (fs) and the Shell, respectively, related to the physical stresses as a function of the corresponding thermal propagation mode of the medium. The connections indicate the Interacting Faces involved in the medium's behavior.

Since the geometry of all the components has to be taken into account for the convection behavior model (whether they exchange thermal heat by convection or not, i.e. whether they are components or media), and in order to simplify the representation of the convection interactions in the *Interaction Diagram*, these interactions between the media are not displayed in the diagram. Figure 135 shows the *Interaction Diagrams* for the first class and second class architectures.

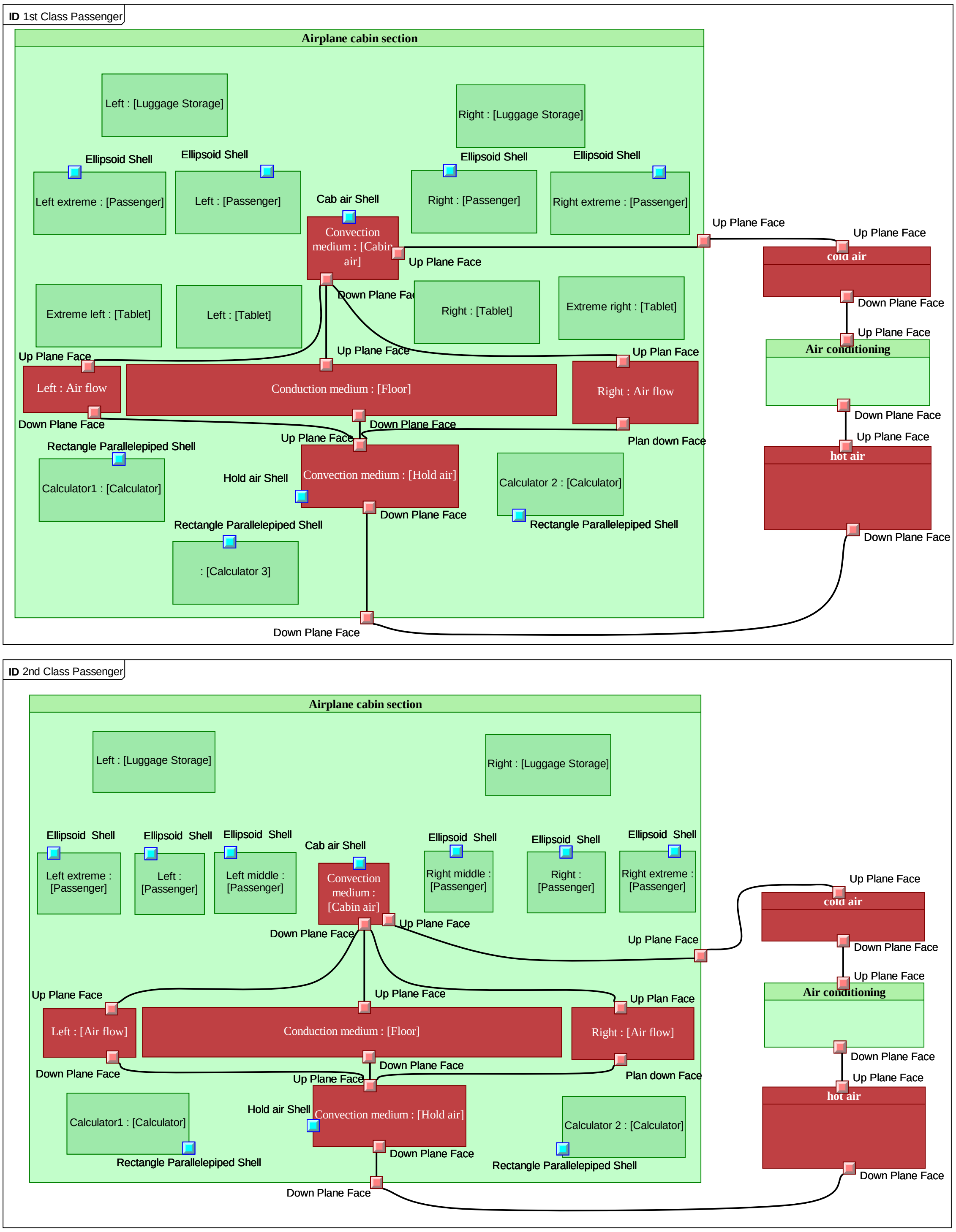


Figure 135: Interaction Diagram of the air conditioning system for the first (top) and second class section (bottom).

These thermal interaction models will usually be generated from the traceability process of the 3D simulations in the FreeCAD/OM environment, managed by the simulation teams, since they may need to complete missing information or add new elements of geometry and thermics to perform the 3D architecture thermal simulation. TheReSE and GERTRUDe extensions allow tracing this data back in the SysML model easily and consistently, using the stereotypes developed previously.

IV. Electric Power Train

The Electric Power Train (EPT) of an electric bus is an academic case study performed in order to demonstrate the ability of the sketcher to model convection behavior with pipes (thermal behavior enrichment provided by the simulation teams) and to compare two alternative architectures.

A. Geometry modeling

Two EPT architectures of an electric bus have been studied (Figure 136). The case study has been built from the data of the TM4 electrodynamic system (TM4, 2017) .

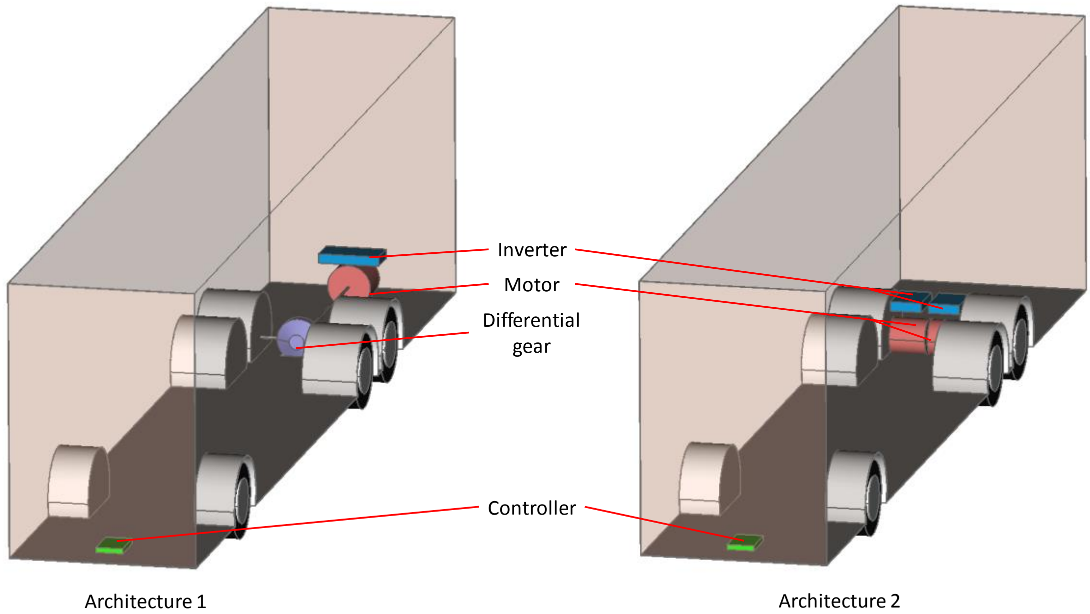


Figure 136: Description of two architectures of the electric bus scenario.

- The first architecture consists of one geared motor, one inverter, one electronic control unit, one differential gear and a chassis.

Component	Geometry	Dimension
Inverter TM4 CM300	Rectangular parallelepiped	• L = 801 mm • W = 414 mm • H = 125 mm
Motor TM4 SUMO HD	Cylinder	• R = 286 mm • L = 505 mm
Control electronic unit TM4Neuro	Rectangular parallelepiped	• L = 274 mm • W = 178 mm • H = 67 mm
Differential Gear Dana 80 Rear	Cylinder	• R = 286 mm • L =168 mm

- For the second architecture, instead of a single motor and inverter, there are two motors and two inverters (one per wheel), but it no longer requires the differential gear.

Component	Geometry	Dimensions
Inverter TM4 CM200	Rectangular parallelepiped	• L = 801 mm • W = 414 mm • H = 125 mm
Motor TM4 SUMO MD	Cylinder	• R = 238 mm • L = 478 mm
Control electronic unit TM4Neuro	Rectangle parallelepiped	• L = 274 mm • W = 178 mm • H = 67 mm

B. *Thermal requirements*

The thermal requirements have been extracted from data sheets of components and will be used as physical solicitations for simulation boundary conditions.

Considering the two architectures, as the performance yield is 95% for the motor and the inverter, we can approximate that all the power losses (5% of the maximum power) are transformed into thermal dissipation, thus the heat power for each component (motor and inverter) will be considered at 5% of the maximum rate of the heat flow. Regarding the temperature requirements, the data sheets consider only the maximum temperature of the coolant (water). These requirements are summarized in Table 51 and Table 52 for the first and the second architecture, respectively.

Table 51: Thermal requirements for the first architecture.

Component	Thermal dissipation	Maximum coolant temperature
Inverter TM4 CM300	Rate of heat flow$12.5kW$	338.15 K
Motor TM4 SUMO HD	Rate of heat flow $12.5kW$	338.15 K
Control electronic unit TM4 Neuro	Adiabatic	N.C.
Differential Gear Dana 80 Rear	Adiabatic	N. C.

Table 52: Thermal requirements for the second architecture.

Component	**Thermal dissipation**	**Maximum coolant temperature**
Inverter TM4 CM200	Rate of heat flow $10kW$	338.15 K
Motor TM4 SUMO MD	Rate of heat flow $10kW$	338.15 K
Control electronic unit TM4Neuro	Adiabatic	N.C.

As there is no information concerning the fan used, we propose to use the following values.

Table 53 Thermal requirements for fans

Component	**Parameter**	**Architecture 1**	**Architecture 2**
Air fan	Inlet temperature	$193.15K$	
	Inlet volumetric flow rate	$0.35m^3.s^{-1}$	$0.25m^3.s^{-1}$
	Outlet pressure	$P = Patm$	
Water pump	Volumetric flow rate	$0.01\ m^3.s^{-1}$	

C. *Geometrical modeling*

In Figure 137, both architectures have been modeled in SysML, using the GERTRUDe extension developed (Chapter 3 IV.A). The architecture components are represented by stereotyped "Component" blocks: each block is enriched with a simplified geometry and its corresponding dimensions can be specified in the predefined unit (in meters here). The relative positioning (position and orientation) of the components can be specified by the TTRS constraints, between their respective Minimum Reference Geometrical Elements (MGRE).

Architecture 1

Architecture 2

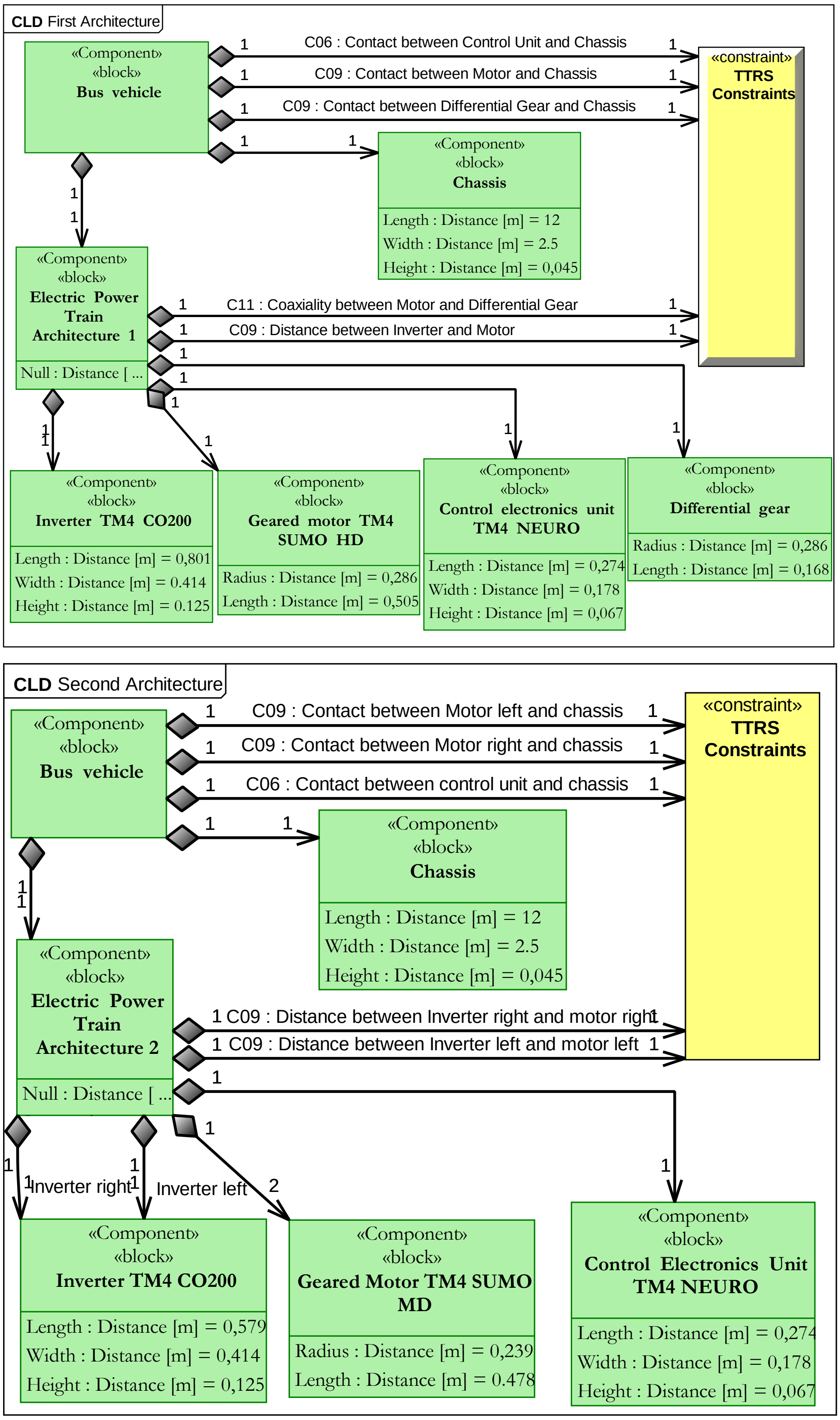


Figure 137: Alternative EPT architectures with GERTRUDe.

Then, the 3D modeling is automatically generated in FreeCAD (Figure 138), using the SAMOS platform developed based on the GERTRUDe2CAD model transformation. For each architecture the 3D model generated from the GERTRUDe model has been enriched to meet thermal specifications (notably for simulation boundary conditions): a heat exchanger (purple color) and a cooling system (orange color) have been added to each architecture.

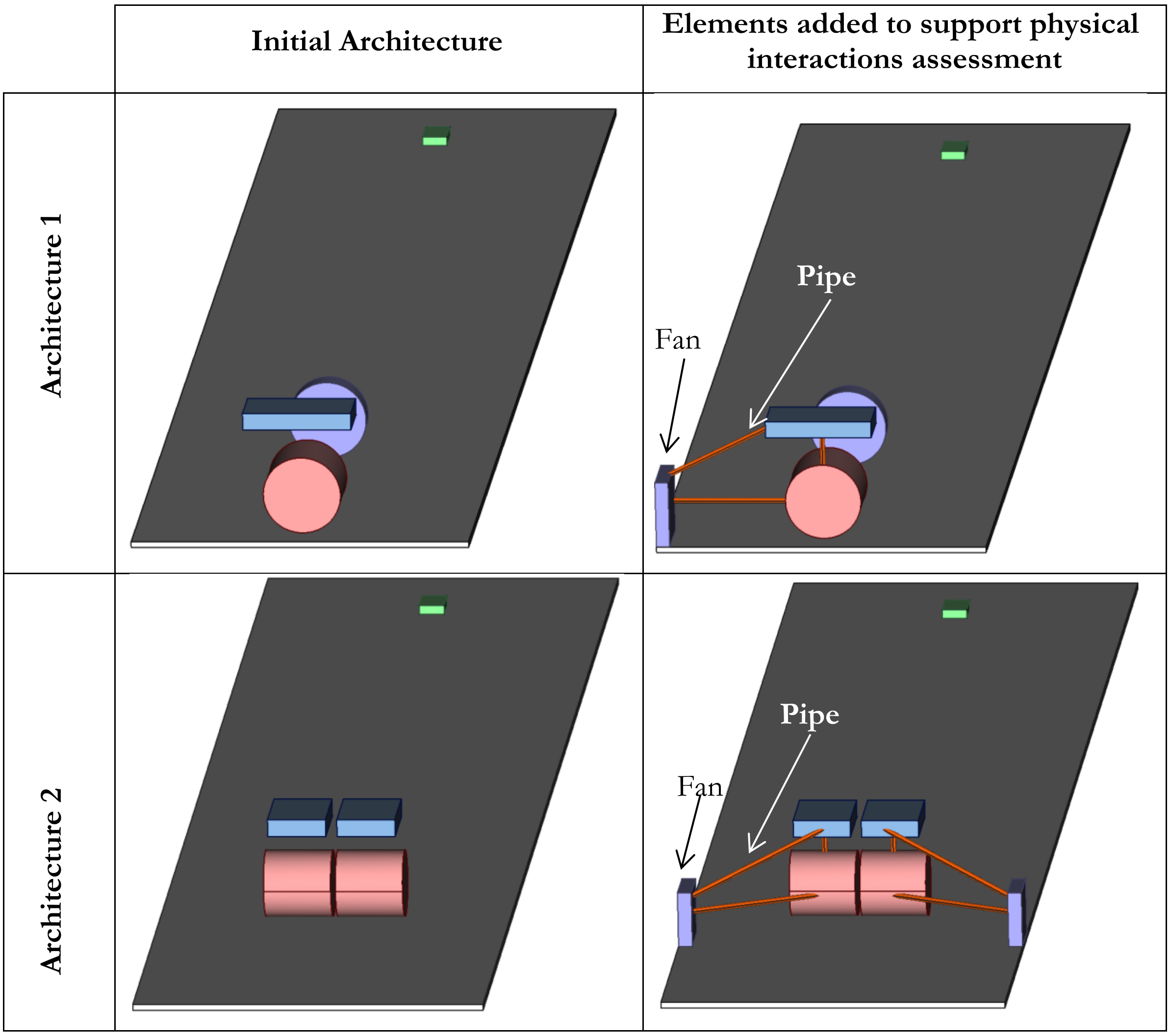


Figure 138: Generated and enriched 3D modeling of the two architectures.

Once the 3D model has been finished, the thermal specifications are added using the thermal 3D sketcher GUI, as described in the second case study “Helicopter bay” (section II). The automatic generation of a Modelica file for a closed fluid circuit has not yet been implemented in the demonstrator.

D. Thermal modeling

For each architecture a Modelica file has been generated manually according to the geometrical parameters and thermal specifications. The Modelica model of the first architecture is presented in Figure 139.

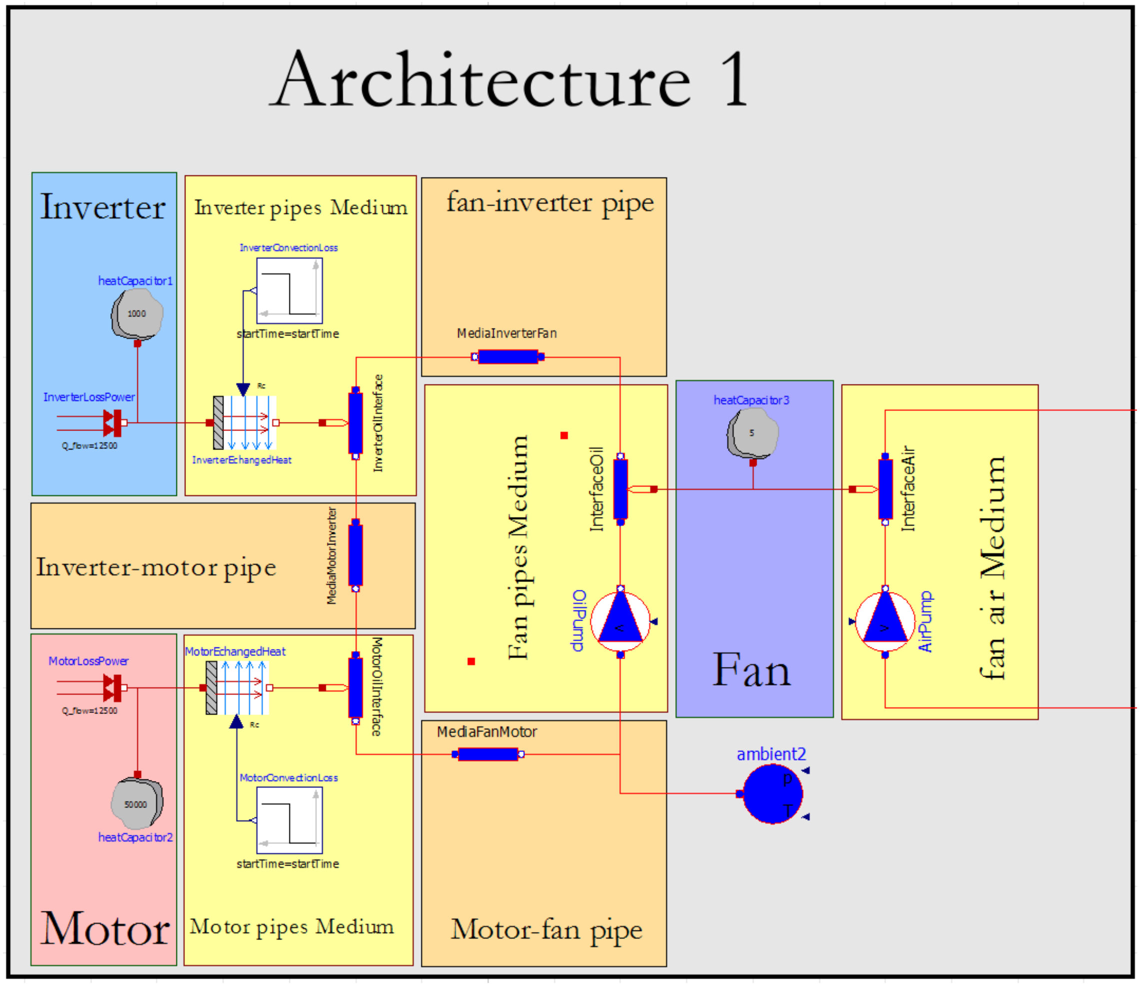


Figure 139: The Modelica model of architecture 1 generated.

The Modelica model includes the cooling system composed of the pipes and the associated media, described below:

- the fan air medium concerns the inlet of air and the heat exchange with the fan;
- the fan pipes medium represents the heat exchange between the cooling system and the fan;
- the inverted pipes medium concerns the heat exchange between the cooling system and the inverter;
- finally, the motor pipes medium allows the heat exchange between the motor and the cooling system.

A geometry has been associated with each medium. The geometries of the cooling system and of the previous media are presented in Figure 140.

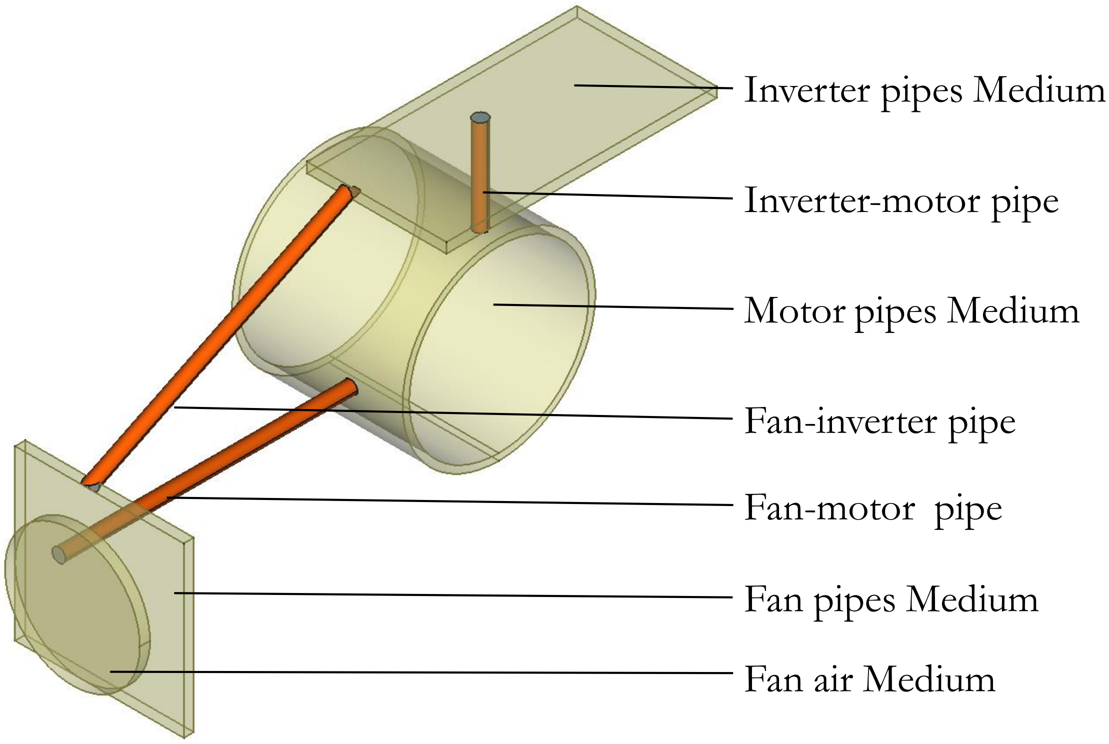


Figure 140: Geometrical view of media and pipes.

The second architecture is more complex from the thermal viewpoint. It is composed of the architecture 1 model duplicated to make two identical architectures side-by-side. these two sub-models can be distinguished in Figure 141 by the suffix "L." for the left part, and the suffix "R." for the right part. These two sub-models are connected through a conduction medium between the left and right motors.

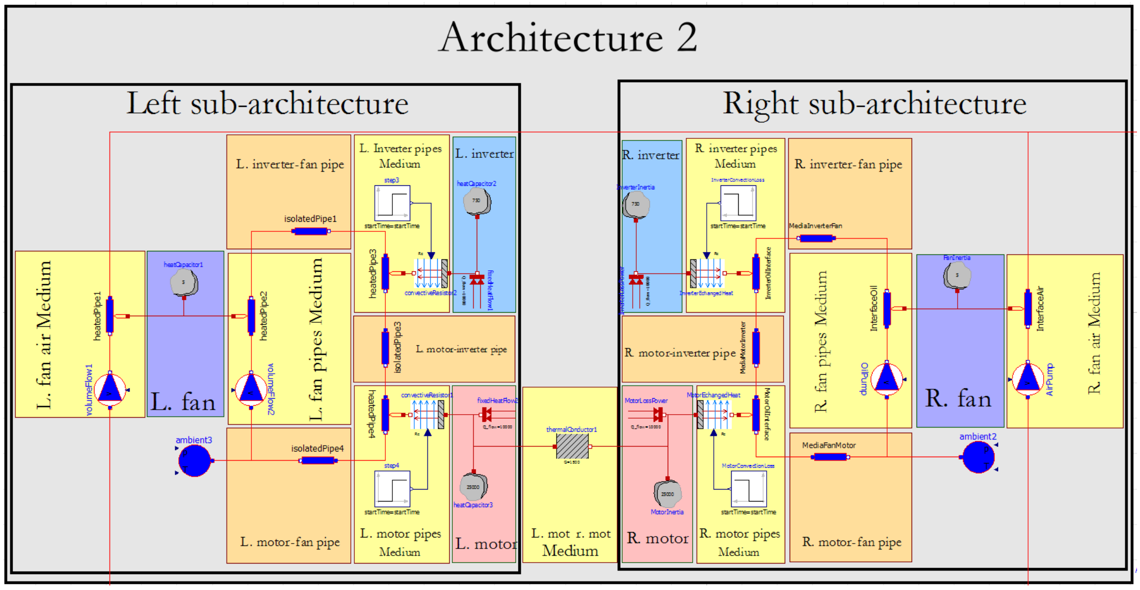


Figure 141: Modelica model of Architecture 2.

The Modelica simulation results are provided in Figure 142. This figure describes the evolution of the temperature of the cooling liquid for both the architectures. The datasheet requirement (338.15 K) is not satisfied by any architecture. However, these architectures can be improved by the addition of a condenser in the cooling system. This condenser will improve the heat exchanged between the air and the cooling system.

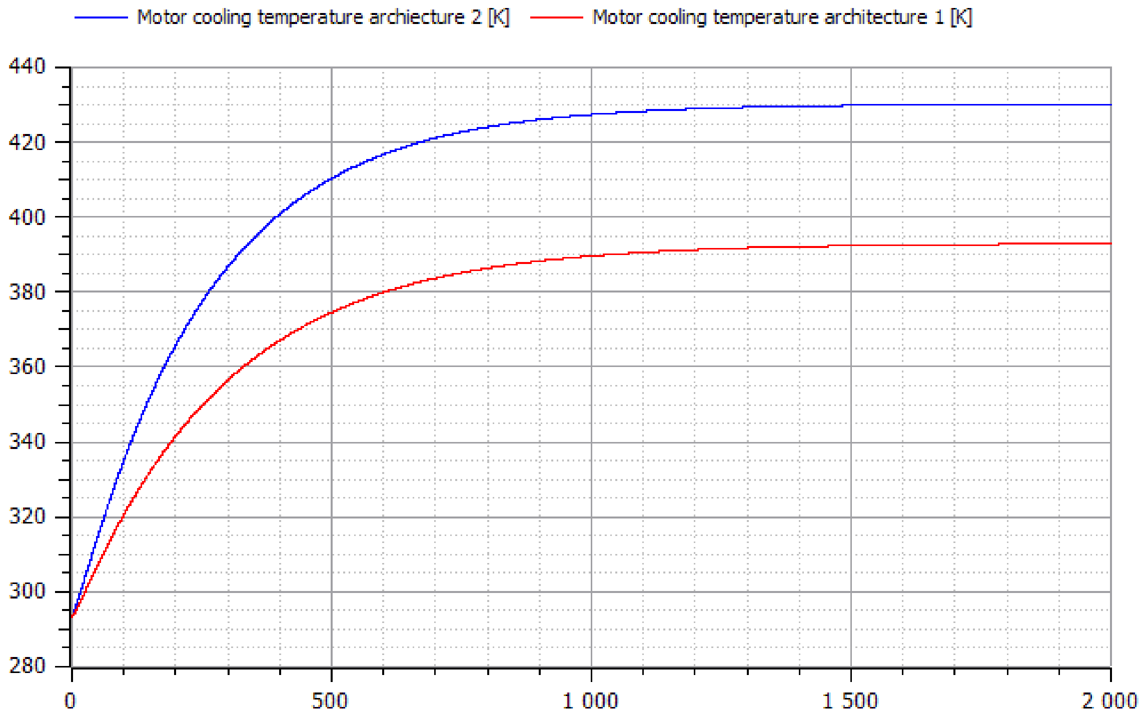


Figure 142: Comparison of the cooling liquid temperature near the motor for architecture 1 and architecture 2.

Architecture 1 comes closest to meeting the thermal requirements. Thus this architecture could be selected, while requiring several enhancements. Both architecture models can be traced back to the SysML System model (Figure 143).

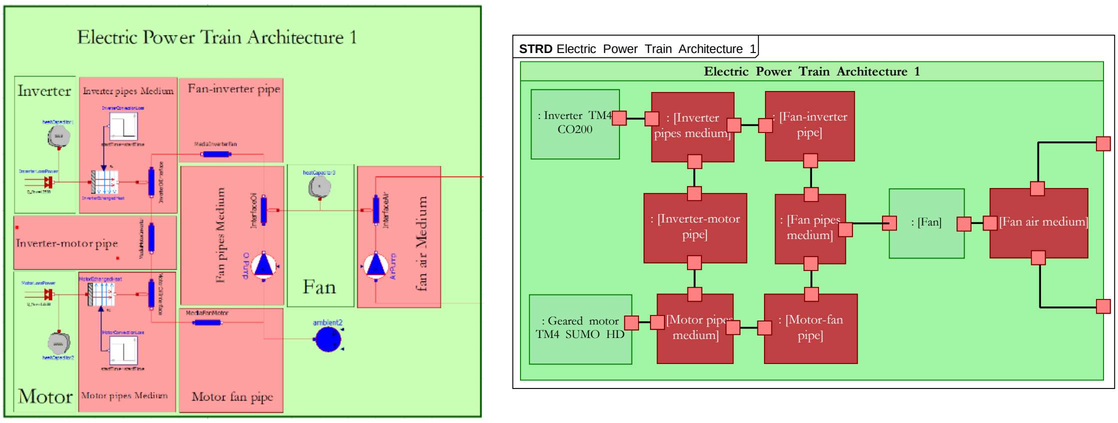


Figure 143: Traceability management for a 3D thermal architecture 1 modeled in Modelica (left), transformed into the TheReSE enriched SysML model (right).

V. SAMOS verification case study: electric bicycle

This case study aims at validating the objective of reducing design time through the SAMOS framework.

The electric bicycle case study was proposed to Supmeca engineering students during the VVIS (system engineering validation, and verification) course, which they attend during their last year of training. The goal of this academic scenario was to analyze the time saved by using the thermal 3D sketcher tool during the conceptual design process. This scenario also allowed testing whether the future engineers might be interested in such a tool. Therefore each student was surveyed in order to collect the deficiencies observed and the corresponding evolutions expected from the thermal 3D sketcher.

A. Case study description

The goal of the case study was to compare three physical architectures that include the installation of a motor and a battery in an ordinary bicycle.

Based on the system model description of the bicycle, and the addition of the electrical systems (battery, and motor) intended for the electric bicycle provided by the teacher (in the role of the System architect), the students had to play the roles of the 3D architects and the simulation teams given the mission of testing the three spatial architectures, add the corresponding requirements including thermal specifications, and thermally evaluate the demonstrator of the thermal 3D sketcher. The future engineers responded to a survey in which they were asked to estimate the time needed to build an architecture and assess the advantages and ergonomics of such a tool.

1. Geometry requirements

The three electric bicycle architectures are composed of a bicycle and two additional components: a motor and a battery whose dimensions are given in Table 54. The crankset is driven by the motor via a chain. All the architectures have the motor set in front of the crankset. The three architectures differ from each other with respect to the position of the battery (Figure 144). For the first architecture, the battery is fixed to the oblique tube of the bicycle frame. For the second architecture, the battery is fixed to the base. Finally, the battery of the third architecture is fixed to the saddle tube.

Table 54: Description of the additional component geometries.

Component	Geometry	Dimensions
Motor	Rectangular parallelepiped	• L = 80 mm • W = 40 mm • H = 100 mm
Battery	Rectangular parallelepiped	• L = 80 mm • W = 85 mm • H = 390 mm

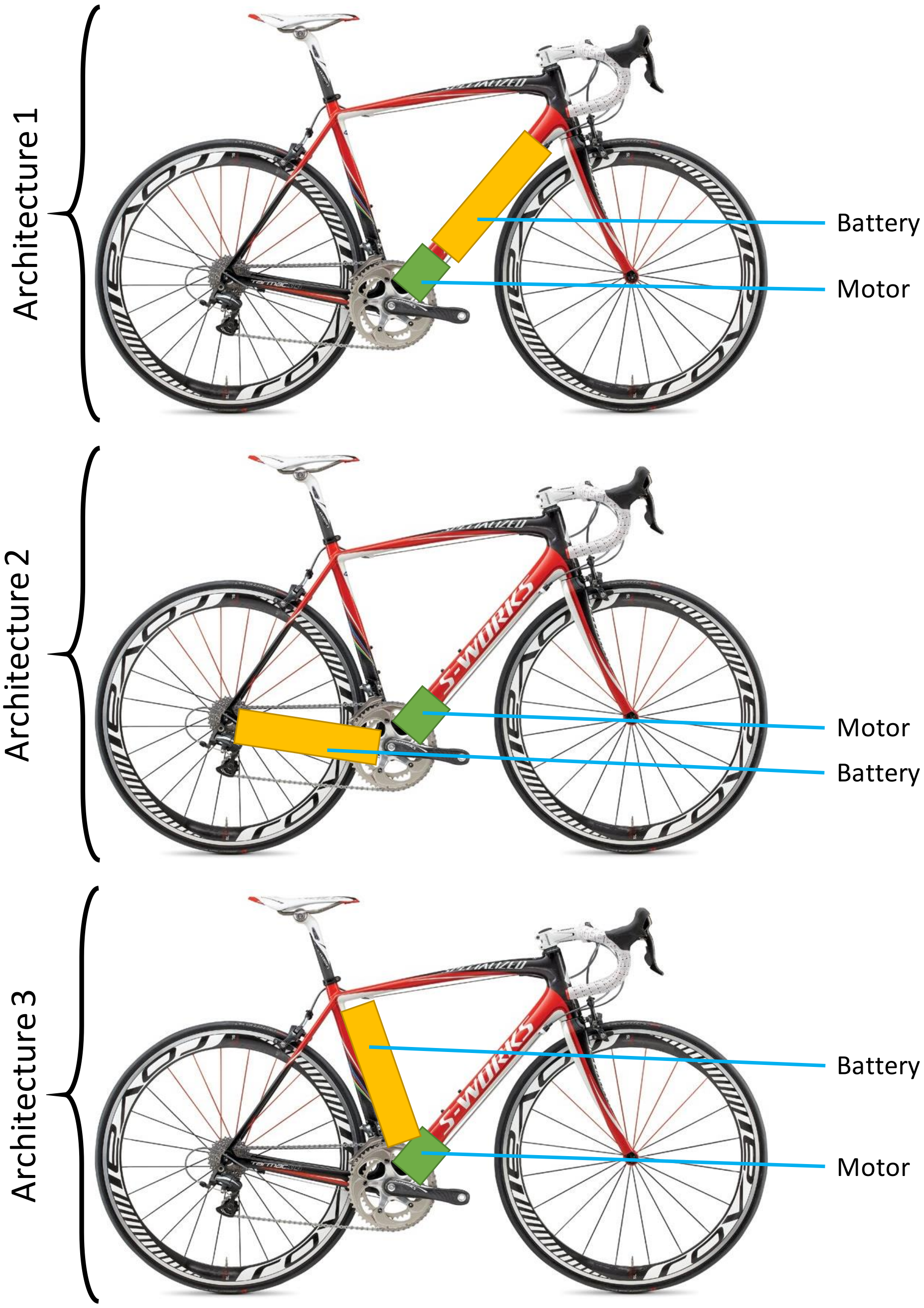


Figure 144: Comparison of the different 3D architectures.

2. Thermal specifications

The thermal specifications are chosen to justify the thermal analysis for such a case study, i.e. between the two additional components. Therefore we consider a new kind of lithium battery with a high power over volume ratio that can generate a 50W heat flow when the motor works as a generator. The maximum temperature for each item of equipment is $30°C$ $(303{,}15\ K)$ and we consider only the convection phenomena between components.

Table 55: Thermal specifications of the electrical bicycle components

Component	Thermal dissipation	Maximum equipment temperature
Motor	Heat flow $30W$	303.15 K
Battery	Heat flow $50W$	303.15 K

The fluid (air) convection is given by the motion of the bicycle. Then, the velocity of the fluid is set at $15km.h^{-1}$. The pressure is atmospheric and approximated at $1bar$

B. Geometry modeling

The first step is to enrich the System architecture used for the thermal analysis with geometrical information. An extract of the corresponding assembly diagram using GERTRUDe is given in Figure 145.

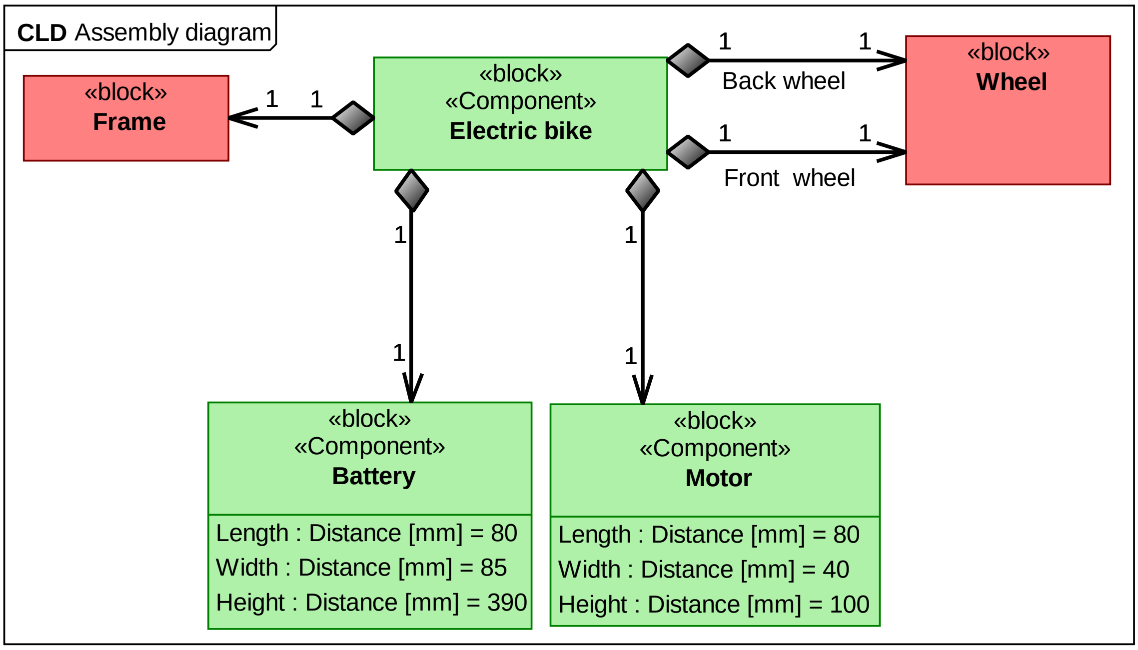


Figure 145: Extract of the electric bike architecture enriched with geometry modeling.

The geometrical components are then transferred through the thermal 3D sketcher into the FreeCAD environment and the thermal requirements are added through the specifications given previously. The convection analysis is performed based on the fluid slice generation process (Figure 146).

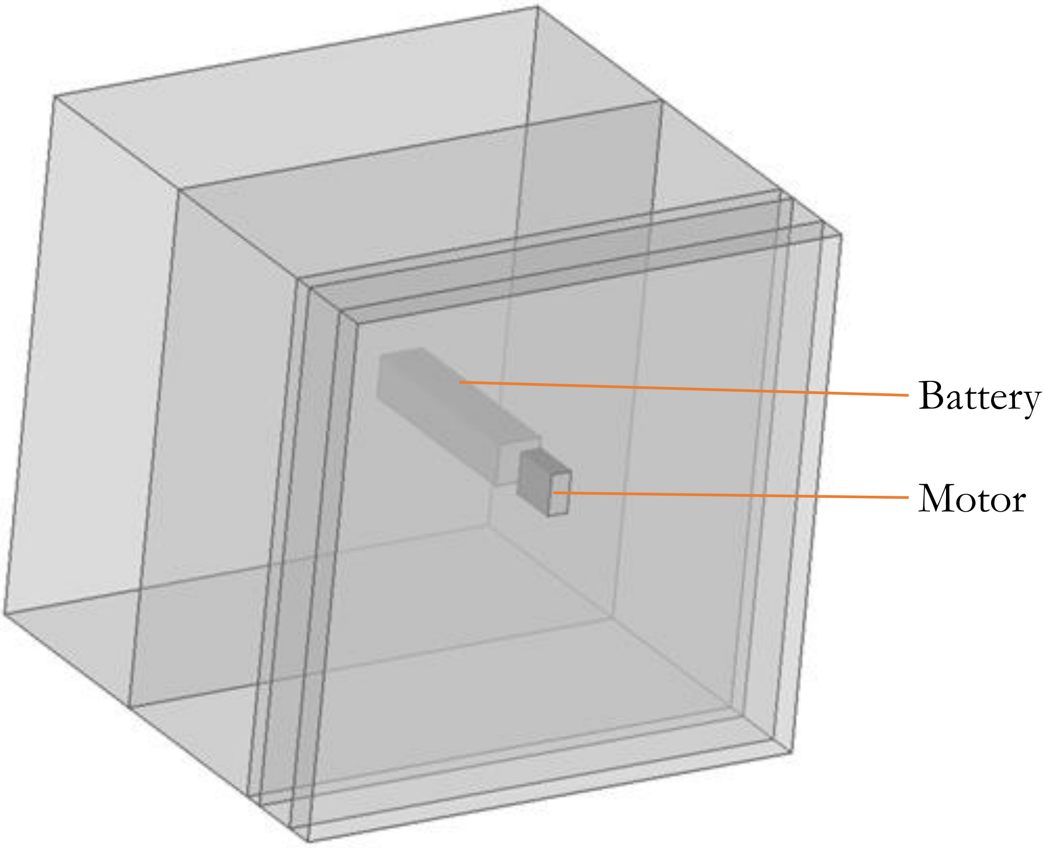


Figure 146: Fluid cutting modeling for the convection analysis of the 3D architecture.

Then the corresponding Modelica model is generated (Figure 147).

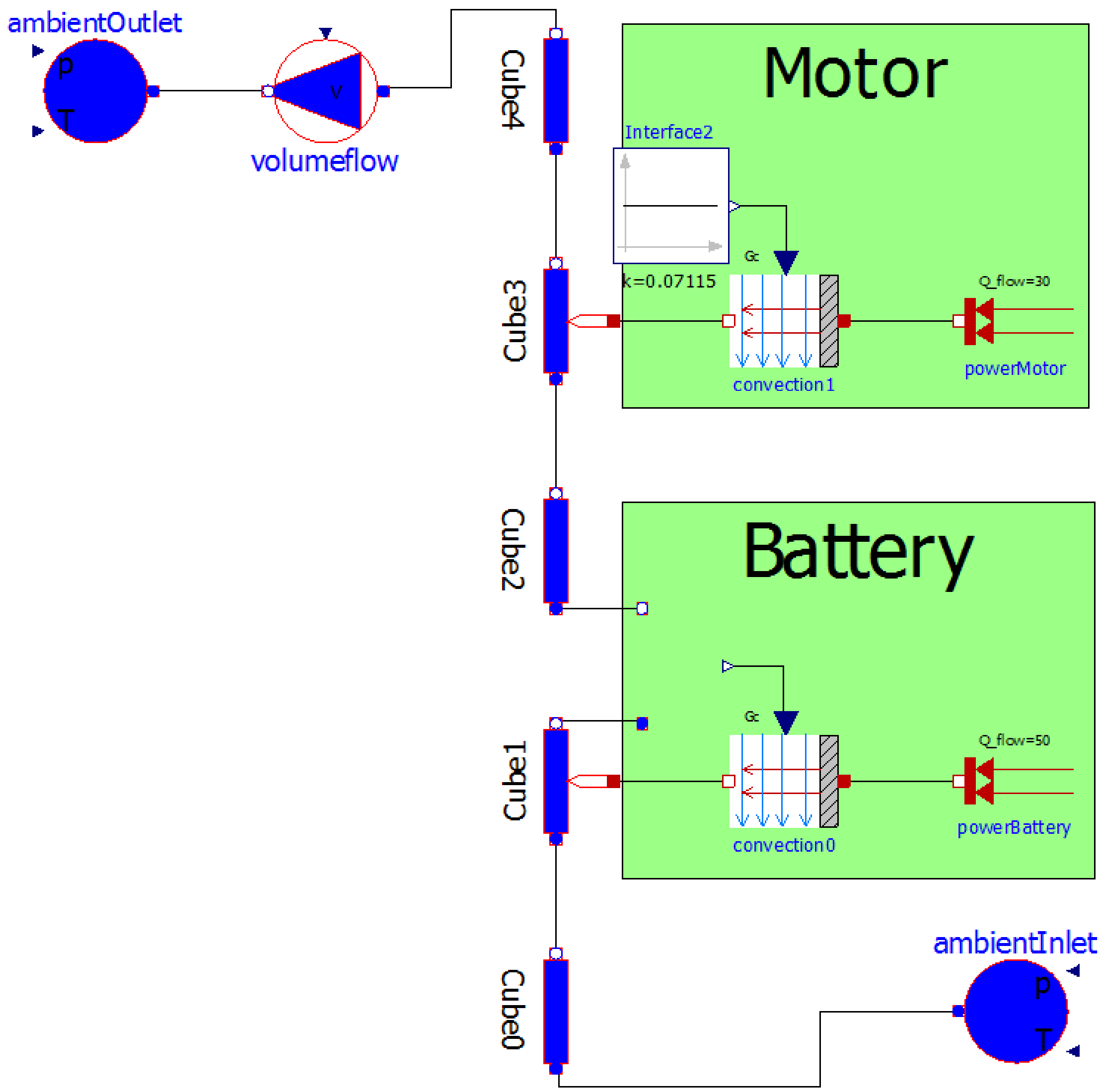


Figure 147: The Modelica convection model generated.

C. Survey results

The 20 engineering students finished the case study in less than 3 hours, whereas a simple Modelica model takes approximately 3h for a student to develop (from my experience). The results of the survey are summarized in Table 56.

Table 56: Summary of the survey results given by students.

Time to simulate architecture 1	Time to simulate architecture 2	Time to simulate architecture 3	Utility	Intuitive
82.5 min	13.5 min	10.5 min	3.92/5	3.2/5

Concerning the thermal 3D Sketcher utility, the students were used to building a system model with SysML and their main reproach was the poor capacity of SysML environments to evaluate architecture. They were obviously pleased to be able to perform it with the thermal 3D sketcher. During this course, my feeling was that the students were happy to transfer the architecture to different tools automatically. Thus the implementation of model transformation proved beneficial for them. Regarding intuitiveness, the evaluation of 3.2 was principally due to several bugs encountered by the students when they modeled the case study.

Finally, regarding the modeling time taken by the students, it comprised the geometrical modeling (including the positioning of different geometrical elements), the thermal specification modeling (including the choice of the thermal laws to be used), and the thermal evaluation of the 3D architecture. For the first architecture, it took an average of 1h 30, but the progression of the curve shown in Figure 148 demonstrates the ease of learning how to use the tool. It also shows that architecture modification with the thermal 3D sketcher does not require repeating all the steps and thus takes less time.

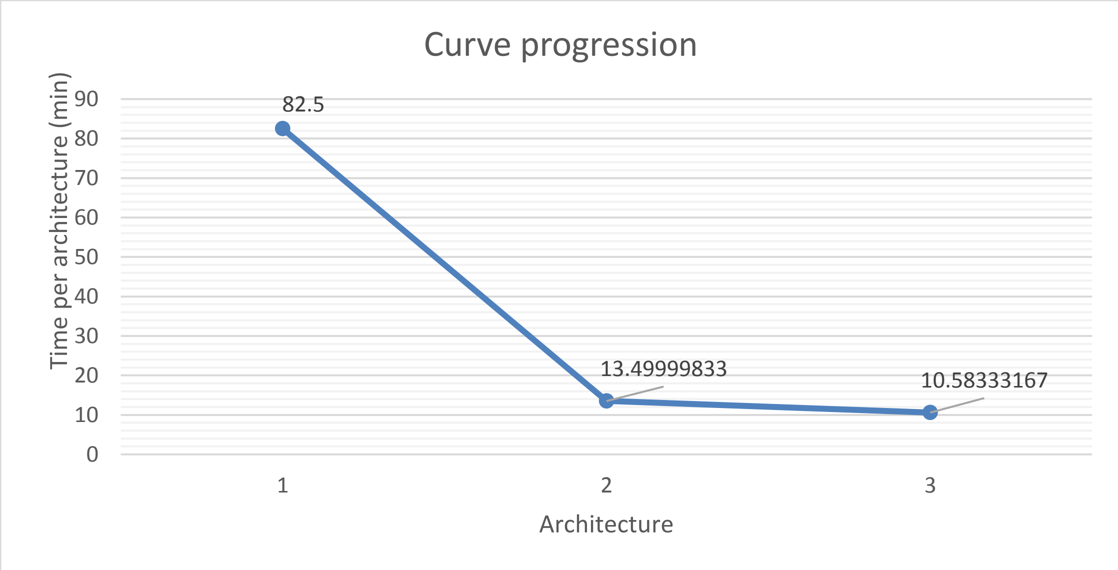


Figure 148: Evolution of design time as a function of the number of changes to the architecture.

The curve shows that the progression is fast. When the first architecture was provided, each student completed the second architecture in less than 20min. Finally, this case study shows that an architecture can be evaluated quickly once the capitalization process has been adopted.

VI. Verification and validation of the SAMOS and thermal 3D sketcher

The verification of the thermal 3D sketcher and the SAMOS approach was performed through the previous case studies.

Regarding their validation, we evaluated whether all the requirements were met by the SAMOS approach and the thermal 3D sketcher. The main requirements are summarized in Table 57.

Table 57: Validation of SAMOS and the thermal 3D Sketcher.

Requirements	#ID	SAMOS	Thermal 3D Sketcher
Data interactions	1.1.1	✓	✓
Actor interactions	1.1.2	✓	✓
Assess 3D architecture under thermal constraints	1.2	✓	✓
Conceptual design phase	1.3	✓	✓
Industrial context constraints	2	✓	✓
Thesis constraints	3	✓	✓

The main initial requirements are validated both by SAMOS and by the thermal 3D sketcher.

- The data interactions requirement is ensured by the M2M transformations, as presented in the different case studies.
- The actor interactions requirement is validated by the SAMOS approach and the thermal 3D Sketcher, which was validated by the engineering students.
- The assessment of 3D architecture under thermal constraints is ensured by both the SAMOS process and the thermal 3D sketcher
- Regarding the conceptual design phase, the SAMOS process is adapted to this phase as many simulations can be performed within a limited time. The demonstrator of the thermal 3D sketcher verifies this requirement by using the languages adapted to this phase, by providing a simple geometry library, and by the speed of the analytical thermal simulation.
- The industrial context requirement was satisfied through my active participation in the industrial case studies with the IRT partners during the PhD thesis, and particularly in the TOICA H2020 project regarding the thermal developments.
- Finally, concerning the thesis constraints, although the duration of the thesis was extended, because of the long writing process (in English), the implementation of the demonstrator and the formalization of the SAMOS process was performed within the period allotted to the PhD.

VII. Conclusions

This chapter described the verification of the approaches proposed through five case studies. The first case study was a conveyer system that allowed verifying the transformation of the geometrical model. Thus, any new geometry added in the GERTRUDe library can be generated in the FreeCAD environment.

The second case study was a helicopter bay. This case study demonstrated the management of thermal information in FreeCAD, the generation of the Modelica model, and the management of the simulation. A comparison of the simulation results with a finite element model showed that the fluid model could be adapted to support a finer analysis, even if the error rate was wholly consistent with the conceptual design phase.

The third case study dealt with an aircraft cab. This case study focused on the representation of the thermal model using a TheReSE extension. The model transformation had not yet been implemented. The TheReSE model is a prerequisite for managing traceability issues.

The fourth case study provided a comparison of two electric power train architectures for an electric bus, based on a thermal model including a cooling circuit. However, model transformation from the 3D environment into Modelica to support the automatic generation of fluid displacement in closed fluid circuit requires further development.

The fifth case study addressed a bike electrification scenario. This simple academic case study was proposed to a group of engineering students to verify the benefit of such a tool, its ergonomics and the design time reduction requirement, particularly in the case of the thermal assessment of an existing architecture that has to be modified.

Finally, the validation of the SAMOS approach and of the thermal 3D Sketcher was performed.

Conclusion and future work

I. Observation 224
II. Need 224
III. Contributions 225
IV. Perspectives 228

I. Observation

On the basis of the expression of a need, conceptual design consists in analyzing it and then in identifying the candidate architectures that have to be evaluated in order to select the architecture(s) to be developed that best meet(s) the initial need. Each step of the design process allows specifying and refining the system to be designed so that its description becomes more precise and detailed. During each stage, the various actors are led to making a succession of choices, to determine an optimal solution. However, during the conceptual design phase, the system architects in charge of validating successive architectures (functional, logical and physical), must make difficult choices, because at the beginning of design few elements are available to assess the different architectures available, or to match them with the multiple requirements that the system must satisfy.

At present, system architects choose the concept architecture on the basis of their expertise and knowledge. Then, the performance requirements are verified, since the embodiment design and detailed design phases will verify the multi-physical requirements through detailed simulation. However, the increasing complexity of systems and limitations due to environmental issues and regulations has led to profound changes in such systems. This increasing complexity implies a large number of interactions between different disciplines that have to be taken into account to ensure the consistency and traceability of data and models, leading companies to introduce the MBSE (Model-Based Systems Engineering) approach to meet these objectives while reducing costs and design time.

II. Need

In order to avoid the considerable increase in design costs and time usually occurring during the development or detailed design phases, due to numerous and long iterations between the different disciplines in various technical departments, one solution would be to evaluate the alternative concept architectures under geometrical and physical constraints during the upstream design phases. Moreover, since all physical behaviors are impacted by the geometry and topology of the architecture, it is necessary to evaluate the latter in 3D.

A process and a platform tool capable of ensuring the homogeneity of the data, whatever the discipline of the different actors, is required to underpin their collaboration. The actors of this phase are the following:

- the system architects, who build physical architecture alternatives from the requirements and specify the associated system requirements;
- the 3D architects, who associate a volume of space and an initial position in space for each component of the architecture;
- and the simulation teams that add multi-physical constraints to the 3D architecture.

III. Contributions

To tackle these issues, we propose an architecture assessment process called SAMOS (Spatial Architecture based on Multi-Physics and Organization of Systems) (Figure 1) that enables these actors to exchange information while limiting the risk of inconsistencies and misunderstandings.

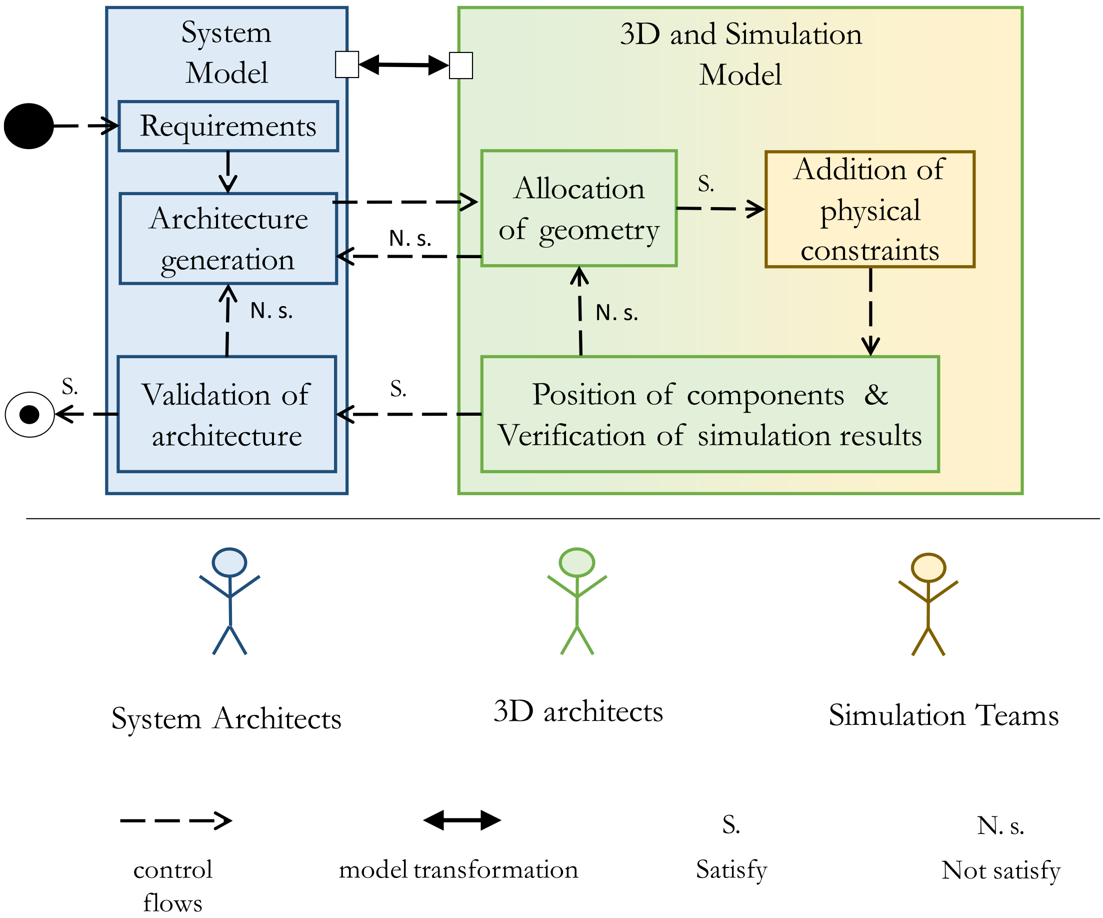


Figure 149: SAMOS framework.

Our first contribution focused on interaction management during the conceptual design phase with the analysis of the roles and the corresponding collaboration needed between the actors involved, in order to propose an approach for managing their (human and data) exchanges. Indeed, since conceptual design is a phase in which many architectures have to be explored, the roles of the system architects, the 3D architects, and the simulation team must be clearly defined. Therefore, it is necessary to verify the corresponding 3D architecture in order to help the system architects to choose a suitable concept architecture based on physical constraints. The definition of the corresponding 3D architecture is performed by 3D architects based on a component architecture provided by the system architects: for each component, the 3D architects assign a simplified geometry and an initial position. Up to now, multi-physical verifications have not usually been performed during the conceptual design phase. This assessment implies that the simulation teams will add multi-physical constraints and launch physical simulations. In order to avoid misunderstandings between these different actors which could lead to delays and increase development time, we proposed a single 3D environment, in which the physical architecture to be assessed will generate a 3D architecture that the 3D architect can complete, and to which the simulation teams will add multi-physical constraints (in this case thermal), and perform the simulation.

The choice of architecture will then be made by system architects based on the simulation results and matching them with the initial requirements. Different technical solutions were identified in the state-of-the art to manage these exchanges. Then, using an MBSE approach (with the SysML language), we selected and implemented the model transformation to support data consistency and traceability and the corresponding automatic processes.

Then, focusing on thermics, we developed a demonstrator tool called "3D thermal sketcher", relying on a 3D environment and two SysML extensions (Figure 150).

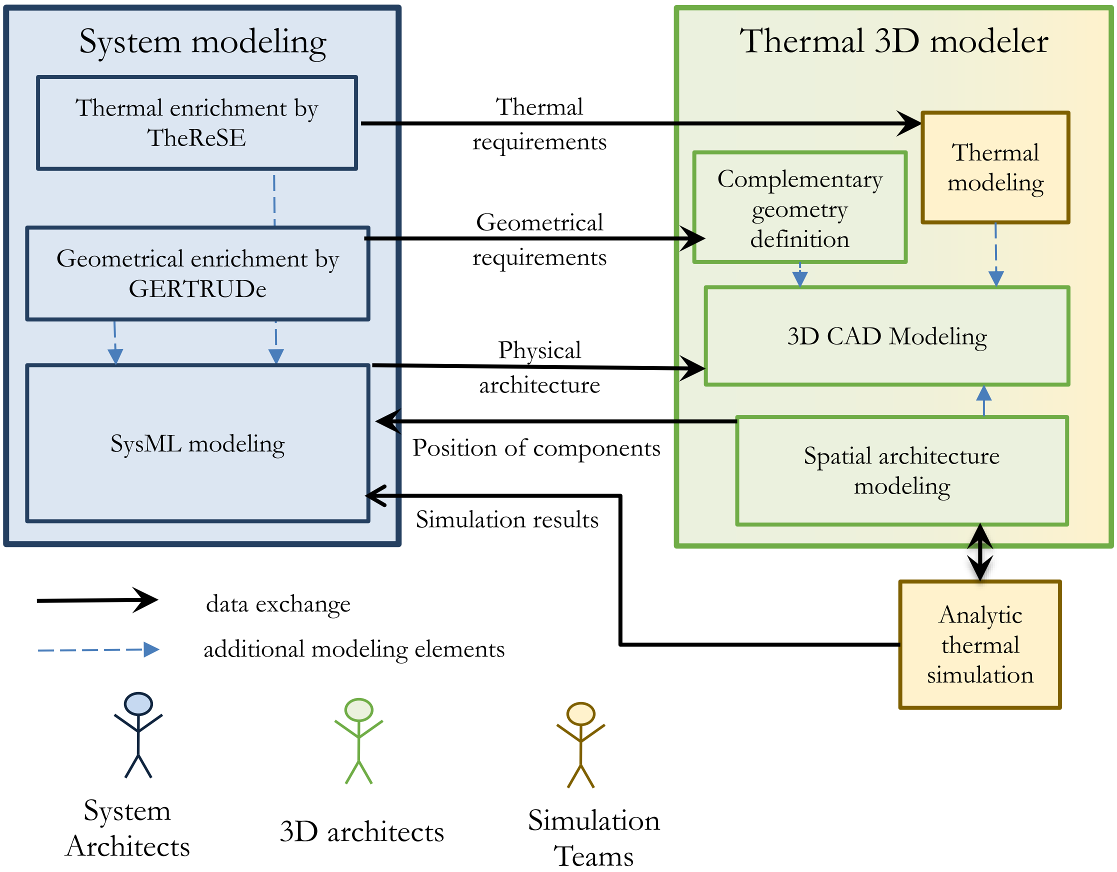


Figure 150: Thermal 3D sketcher structure.

Our second contribution addressed the integration of geometrical and physical modeling in an MBSE context for the conceptual design, through the development of two SysML extensions, in order to enrich the "systems view" models by adding geometrical data (simplified associated volumes with dimensions and positions), thermal data and the corresponding constraints. This made it possible for the simulation teams to share useful constraints such as bulk volume, distances, thermal boundary conditions etc. with multidisciplinary teams in order to begin their pre-sizing work. This common information shared between all the multi-disciplinary teams will be helpful to ensure the global consistency of the system.

In order to take into account geometrical considerations in the early stages of mechatronic design, we proposed a SysML extension for the geometry: GERTRUDe. This profile and the HMI developed not only greatly facilitated the specification of geometrical requirements but also allowed modeling the components of physical architectures with simplified geometries and specifying geometrical constraints, based on the TTRS (Technologically and Topologically Related Surfaces) theory, in order to position them in relation to each other. The GERTRUDe profile allows generating a geometry-enriched physical architecture. Its main added value is:

• to provide a means to System Architect to specify geometrical requirements for enriching physical architectures;

• to take into account geometrical constraints (component positioning) and to facilitate the work of preliminary design teams, by prepositioning the components before evaluating their corresponding physical interactions;

• by relying on the model transformation developed, to trace back a 3D architecture, including the modeled shell and face positions, from a 3D environment, using the TTRS theory.

In order to integrate thermal modeling in the system model, we developed a second SysML extension called TheReSE: Thermics Related SysML Extension. TheReSE allows enriching SysML models with thermal information, to enable the system architects to specify and automatically verify thermal requirements. This SysML extension, based on the GERTRUDe extension, will help systems architects to validate the concept physical architectures under geometrical and thermal constraints. It satisfies the two following objectives:

- system architects must be able to enrich physical architectures with thermal considerations (the physical and geometrical properties of components and thermal and geometrical constraints), in order to facilitate the generation of a 3D architecture that meets the system's physical requirements;

- they also need to retrieve the thermal simulation results of the pre-validated architecture provided by the simulation teams from the thermal architecture model, and the corresponding 3D architecture (component geometry and positioning).

The third contribution addressed the implementation of TTRS theory for thermal modeling, including the three thermal modes (conduction, convection, radiation). Solving analytical thermal laws relating to conduction can be simplified by viewing the components' geometry through their TTRS class. Using TTRS also allows performing thermal conduction analysis on more complex geometries. Regarding radiation, we proposed to calculate the view factor between two components regarding their TTRS and associated positioning constraints, in order to determine the equivalent Interacting Face.

Our fourth contribution dealt with 3D architecture assessment under thermal constraints, while bridging the gap between the "System" team and its model, and the technical domain field teams and their physical behavior models. Model transformation rules were applied to couple in a single platform (Thermal 3D Sketcher) system models in SysML, and 3D architecture and simulation models in Modelica language. This platform allows automatically (and rapidly) generating models and updating from one model to another, depending on whether the model transformation considered is intended for specification or traceability, in order to meet the requirement of reducing design time.

Finally, a whole MBSE approach was performed in the framework of this PhD thesis, from the initial requirements (including scientific issues, industrial and thesis constraints) to the validation of the SAMOS framework and the thermal 3D Sketcher demonstrator, after verifying its potential through five academic and industrial case-studies.

IV. Perspectives

In addition to the fact that we demonstrate the feasibility of the SAMOS concept for thermal modeling with simple geometry through the thermal 3D sketcher, this work brings to light numerous perspectives.

- Exploitation of GERTRUDe

The integration of geometrical specifications in the early stages of design, and notably during the emergence of different physical architectures in SysML could also facilitate geometrical metrics, and tackle physical integration issues in mechatronics systems. Indeed, Warniez & al. (Warniez, et al., 2014) proposed to compare the different candidate spatial architectures of a mechatronic system with geometrical metrics, which require component positioning information, in order to assess their corresponding compactness on the basis of their apparent density, component accessibility and assembly compactness (Figure 12). Finally, coupling between the thermal 3D Sketcher and metrics can help system architects to choose the best-adapted architecture, since using metrics also facilitates decision-making.

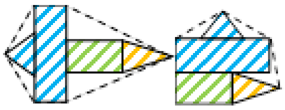


Figure 151 Convex hull of assembly

Based on the TTRS theory, the GERTRUDe profile could also be used to evaluate the tolerance between the functional surfaces of spatial architectures. Indeed, the advantage of TTRS theory is that it has already been used to manage geometry tolerancing analysis developed in other studies (Clement, et al., 1996).

- Multi-physical 3D Sketcher

Furthermore, the present work represents an initial step in taking into account thermal interactions related to geometry as early as possible in the design life cycle, in order to validate whether candidate spatial architectures with physical constraints meet the performance requirements and thermal behavior expected from them. New research perspectives will focus on the implementation of the SAMOS framework for other domains of physics like electromagnetism and dynamic vibrations. It will provide a Multi-physical 3D Sketcher to specify multi-physical constraints in the system model in SysML. It will also verify, through physical simulations, that 3D architectures of concepts satisfy requirements, while facilitating collaboration between multi-domain simulation teams. Eventually, its could be envisaged for managing multi-physical coupling.

- Improving thermal analysis

Concerning thermal analysis, the works presented in chapter 4 showed that it is not possible to associate a simplified thermal conduction law to each component TTRS class. It might be interesting to increase the number of symmetry classes to be considered in order to easily manage thermal conduction behavior for more complex geometries.

Moreover, the helicopter bay case study showed that the hypothesis chosen for the displacement of the fluid, considered irrotational, was not verified in this case, implying a large error in the simulation results. Thus, a combination of finite element analysis with analytic modeling could improve the thermal model: the avionic bay could first be simulated with a finite volume model, after which the corresponding velocity profile would be applied to manage heat transfer in thermal analytic equations.

The addition of other thermal models, whether from the Modelica standard library with different levels of detail, or from libraries defined by the user, as well as reduced models, experimental mapping, empirical laws, etc., could be interesting for future work.

These models could be parsed, analyzed and compared through a "cost function" based on a graph grammar to be developed, to measure the distance/gap between two models. The closer two models are to each other, the shorter the distance between them. This cost function has not yet been defined precisely (since not yet implemented), but its parameters could be the following:

- physical nature of ports;
- number of required input/output ports;
- model parameters involved.

Then a script could estimate and calculate the distance/gap between the Modelica models proposed in our existing model transformation and other Modelica models. Then, these Modelica models would be sorted according to their calculated distances. Simulation teams could analyze the different Modelica models, and choose the most suitable model to introduce in their architecture model. A similar approach (with a cost function) was taken by Fontaine et al. (Fontaine & Hammami, 2016) to compare two pseudo-MICs (Model Identity Cards). The comparison between two MICs can be naturally adapted to a comparison between two Modelica models. Indeed, a pseudo-MIC contains predefined information on a model, like ports, parameters, and other information. Fontaine et al. created a generator of pseudo-MICs from any Modelica model (MOtoMIC), which can be used to build pseudo-MICs from Modelica files. The CompMic comparator developed, which analyzes the distance between two MICs and proposes a list of compatible Modelica models so the user can choose the most suitable one, could help simulation teams to choose the thermal model to be implemented in the thermal 3D Sketcher

Other future works could include proposing to simulation teams that they replace the thermal equation within a medium with a pseudo-MIC model, in the case where they do yet know the specific behavior model they want to apply. Moreover, since a MIC is only a model specification, it does not specify any language or tool for the corresponding model, thus it could be used as a "black box" model to validate a thermal simulation architecture, before having real thermal behavior models.

- Automatic data exchange

Regarding model transformation management, since the M2M platform developed is a PSM-PSM (Platform-Specific Model), strongly dependent on the three tools chosen (PTC, OpenModelica and FreeCAD tools), a future solution would be to transpose this approach to a neutral format (XMI-STEP), so that it can be deployed for industrial use.

Another future work could concern the implementation of model transformation to other simulation tools, to improve the ability of the sketcher to support models according to various levels of accuracy.

- Model quality assessment

In the future, it would be interesting to introduce an uncertainty/model quality management approach in this platform to quantify the thermal simulation errors resulting from geometry simplification and assess the reduction of the corresponding simulation time compared to a FEM simulation.

List of personal publications

R. Barbedienne, O. Penas, J. Y. Choley, A. Rivière, A. Warniez and F. Della Monica, "Introduction of geometrical contraints modeling in SysML for mechatronic design," 2014 10th France-Japan/ 8th Europe-Asia Congress on Mecatronics (MECATRONICS2014-Tokyo), Tokyo, 2014, pp. 145-150.

R. Barbedienne, O. Penas, J. Y. Choley and L. Gasser, "TheReSE: SysML extension for thermal modeling," 2015 Annual IEEE Systems Conference (SysCon) Proceedings, Vancouver, BC, 2015, pp. 301-308.

R. Barbedienne, Y. Ben Messaoud, J. Y. Choley, O. Penas, A. Ouslimani and A. Rivière, "SAMOS for Spatial Architecture based on Multi-physics and Organisation of Systems in conceptual design," 2015 IEEE International Symposium on Systems Engineering (ISSE), Rome, 2015, pp. 135-141.

A. Warniez, O. Penas, R. Plateaux and R. Barbedienne, "SysML geometrical profile for integration of mechatronic systems," 2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, Besancon, 2014, pp. 709-714.

R. Plateaux, O. Penas, R. Barbedienne, P. Hehenberger, J.-Y. Choley, & A. Warniez, "Use of Technologically and Topologically Related Surfaces (TTRS) geometrical theory for Mechatronic Ontology" In CAD'16 2016, pp. 183-188.

R. Plateaux, O. Penas, R. Barbedienne, P. Hehenberger, J.-Y. Choley, & A. Warniez, "Use of technologically and topologically related surfaces (TTRS) geometrical theory for mechatronic design ontology" Computer-Aided Design and Applications (2017), 1-15.

Appendix 1 – Résumé étendu

La forte concurrence actuelle entre les entreprises requiert des systèmes de plus en plus complexes, à des coûts et durée de conception réduits. Cette complexité croissante des systèmes implique un grand nombre d'interactions entre les différentes disciplines qui doivent être prises en compte pour assurer la cohérence et la traçabilité des données et des modèles. C'est la raison pour laquelle l'approche MBSE (Model-Based Systems Engineering, Ingénierie des Systèmes basée sur les modèles) a été introduite dans le cycle de vie de conception des entreprises.

Par ailleurs, l'augmentation considérable des coûts et du temps de conception intervient généralement pendant les phases de développement ou de conception détaillée, à cause de nombreuses et longues itérations entre les différentes disciplines dans divers services techniques. En effet, leurs simulations respectives ne sont pas toujours cohérentes, en raison d'une part du manque d'homogénéité des données et de la difficulté de collaborer pendant la conception, et d'autre part parce que leurs outils diffèrent d'une discipline à l'autre et sont généralement non-interopérables.

Pour faire face à ces problèmes, cette thèse propose un processus et une plateforme outillée pour évaluer, dans les phases de conception amont, des architectures de concept sous contraintes géométriques et physiques et ainsi réduire le risque de modifications tardives pendant les phases de conception ultérieures impactant le temps et le coût de conception.

Pour répondre au besoin de collaborations entre les acteurs de cette phase, à savoir :

- l'architecte système, qui élabore des alternatives d'architecture physique à partir des exigences et spécifie les exigences systèmes associées ;
- l'architecte 3D qui associe un volume d'encombrement et une position initiale dans l'espace pour chaque composant de l'architecture, et
- les équipes de simulation qui ajoutent les contraintes multi-physiques sur l'architecture 3D pour réaliser leurs simulations;

nous proposons un processus nommé SAMOS (Spatial Architecture based on Multi-physics and Organization of Systems, Architecture spatiale basée sur la multi-physique et l'organisation des systèmes) (Figure 152) permettant à ces acteurs d'échanger des informations en limitant le risque d'incohérences et d'incompréhension. Puis, en nous focalisant sur la thermique, nous avons développé une plateforme outillée « Skecher 3D thermique », s'appuyant sur un environnement 3D, un outil de simulation et deux extensions SysML.

En s'appuyant sur l'état de l'art qui présentait trois structures permettant l'échange d'informations entre différents acteurs et environnements de modélisation : le modèle unique, la transformation de modèle et la fédération des modèles, nous avons retenu la transformation de modèles qui répondait le mieux à nos exigences. En effet, cette approche permet de faciliter les échanges automatiques d'informations et de données entre l'architecte système, l'architecte 3D et les équipes de simulation, et ainsi diminuer la redondance, les incohérences et donc le temps de conception.

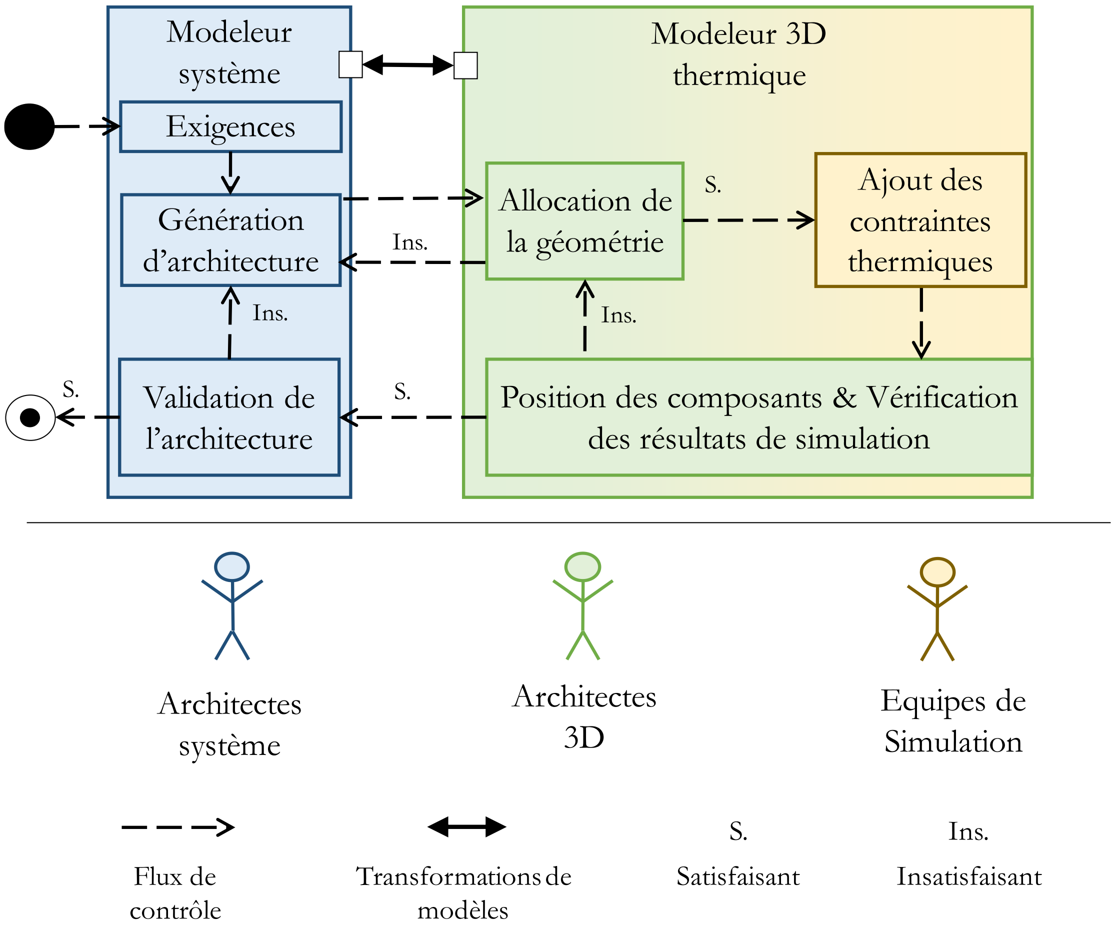


Figure 152 : Processus SAMOS.

Pour permettre la spécification de la géométrie et du positionnement spatial de chaque équipement, une extension SysML GERTRUDe permet d'enrichir le modèle SysML et l'architecture physique proposée par l'architecte système avec : des formes géométriques simples des composants, des dimensions et des contraintes de positionnement. GERTRUDe s'appuie sur le modèle géométrique des SATT enrichi avec des paramètres intrinsèques (pour prendre en compte à la fois les informations topologiques et géométriques). De la même façon, l'extension SysML TheReSE, basée sur GERTRUDe, permet la gestion des exigences thermiques : propriétés thermiques de composants et spécification des interactions thermiques susceptibles d'intervenir entre les composants de l'architecture. Enfin, ces deux extensions ont été développées de manière à pouvoir supporter la traçabilité des données dans le modèle SysML, après la simulation d'une architecture dans l'environnement 3D.

Les règles de transformation de modèles sont explicitées et permettent de générer automatiquement une architecture définie en SysML par l'architecte système, pouvant inclure des contraintes géométriques et thermiques, dans un environnement 3D. Réciproquement, une architecture 3D enrichie par un comportement thermique peut être tracée de l'environnement 3D vers SysML pour faciliter la tâche de vérification de l'architecte système et son choix d'architecture. L'environnement 3D nommé Modeleur 3D thermique a en effet été enrichi, afin de supporter l'ajout de contraintes thermiques et de pouvoir transformer ces données en modèles de simulation. Le principe de la plateforme développée est décrit dans la Figure 153.

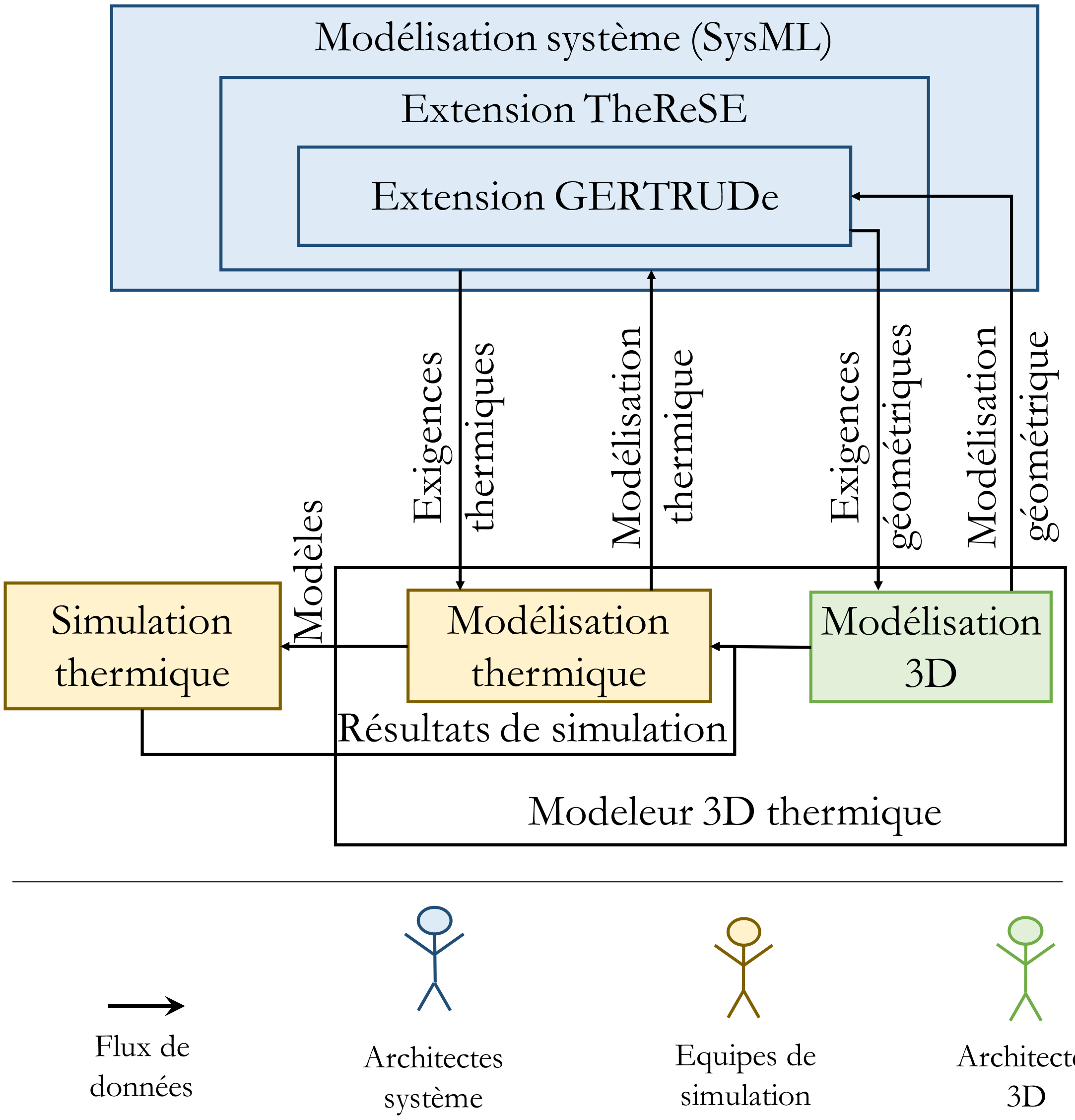


Figure 153 : Plateforme outillée de SAMOS : Modeleur 3D thermique.

Ainsi pour faciliter la collaboration et la flexibilité (pour s'adapter aux différents produits des entreprises), les acteurs peuvent interagir librement sur l'outil proposé, en se focalisant sur le modèle (SysML, 3D ou thermique) qui le concerne. Ainsi, par exemple, les interactions thermiques entre les différents équipements peuvent être soit spécifiées en SysML par l'architecte système (contraintes thermiques), soit être ajoutées dans l'environnement 3D par les experts thermiques.

Ces informations permettent de définir alors un réseau d'interactions thermiques, qui intègre à la fois les informations géométriques et thermiques. Ce réseau est alors transformé en un modèle thermique implémenté en Modelica, qui permet par simulation d'évaluer la température des composants (faces) et des fluides.

L'approche proposée a été implémentée et vérifiée dans un démonstrateur sur plusieurs cas d'études industriels et académiques, et a été validée, au regard des attentes scientifiques et industrielles.

Appendix 2 – The SIM process

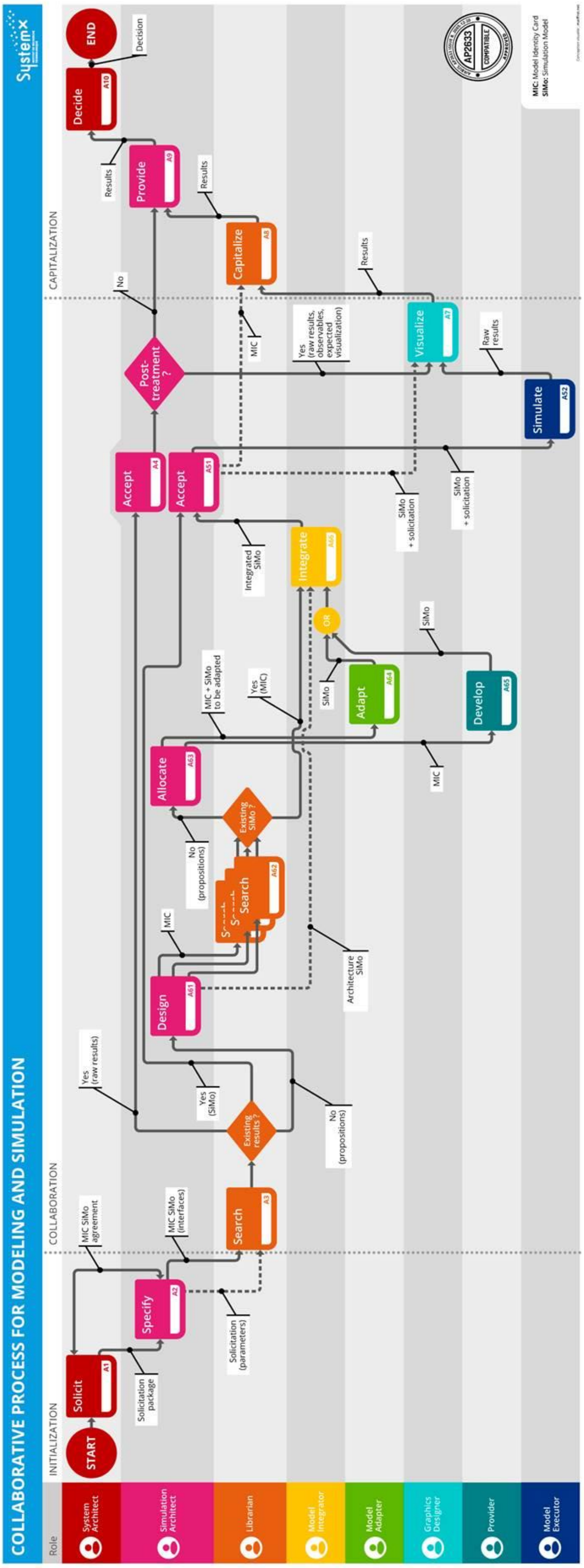


Figure 154: Description of the SIM process

Appendix 3 – System Modeling Language/ Unified Modeling Language

I. Overview of the various diagrams 237

II. Description of the diagrams used 239

A. Requirement diagram 239

1. Object descriptions 239
2. Different kind of links 239

B. Block definition diagram & Class diagram 239

1. Objects description 239
2. Different kind of links 240

C. Internal block diagram & Composite structure diagram 240

1. Object description 240
2. Different kind of links 240

D. Activity diagram 241

1. Objects description 241
2. Different kind of links 241

E. State machine diagram 241

1. Objects description 241
2. Different kind of links 242

F. Sequence diagram 242

The System modeling language (SysML) is a profile of the Unified Modeling Language (UML), a formal object language normalized in 1997 by the OMG (Object Management Group) and which is defined by a metamodel (specification of object oriented programming language) that gives it the major advantage of not having to depend on a programming language. UML is a language which allows software developers to describe the hierarchy within a program. UML was extended for Systems Engineering applications, with the emergence of SysML (SYStem Modeling Language), initiated by the International Council on Systems Engineering (INCOSE) to support MBSE (Model-Based Systems Engineering) and has been subject to an OMG specification since 2006. This shared design language addresses the design of complex multi-domain systems. It improves common model understanding in the collaborative engineering design of a product, and ensures the traceability of the design process. Finally, these languages were developed to capitalize development information, optimize development, and facilitate communication between different developers/designers. This appendix proposes a description of the mechanism used in SysML and the differences with the UML mechanisms.

I. Overview of the various diagrams

Different diagrams used for UML and SysML are described in Figure 155.

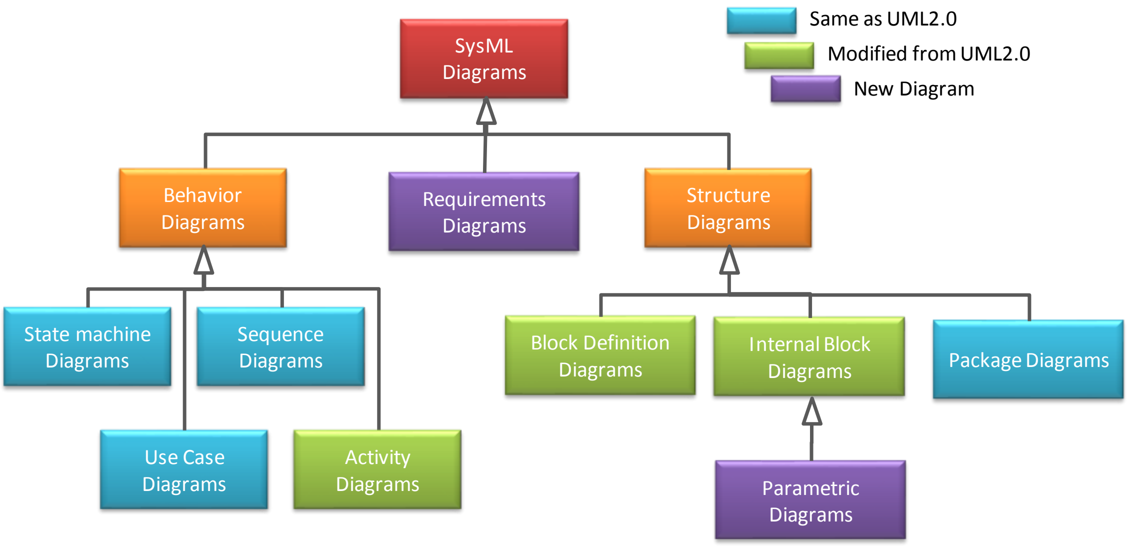


Figure 155 : SysML-UML diagrams

Table 58 briefly describes the different diagrams in SysML that are stereotyped from UML ones.

Table 58: Description of the different diagrams

SysML diagram	UML diagram	Description
Block Definition Diagram (BDD)	Class diagram (CLD)	The Block Definition Diagram and Class diagram propose to structure blocks and classes respectively.
Internal Block Diagram (IBD)	Composite Structure diagram (STRD)	The Internal Block Diagram and composite structure diagram detail the data exchanged between different elements via different ports. They also structure the elements.
Parametric Diagram (PARAM)	N/E*	The parametric diagram has no equivalence in UML. It allows describing the different relations between the parameters of a block. For example, calculating the global mass of a system according to the mass of each subsystem.
Use Case Diagram (UCD)	Use Case Diagram (UCD)	The Use Case Diagram represents the users' interactions and describes the different use cases in which users are involved.
Sequence diagram (SEQ)	Sequence diagram (SEQ)	The Sequence diagram displays how objects interact with each other. It contains the execution sequence of the different messages/signals between components or external actors.
Activity diagram (ACT)	Activity diagram (ACT)	The Activity diagram models the activities of the system and the corresponding activity input and output flows.
State Machine diagram (STM)	State machine diagram (STM)	The State machine diagram displays the different states in which the system can be. It also describes the events that can move the system from one state to another.
Requirement diagram (REQ)	N/E*	The requirement diagram is the most significant additional diagram from UML: it allows representing the hierarchy and links between different requirements of the system (for specification), and also with other modeling elements (for traceability).

* Non Equivalence

This annex focuses on the diagrams used in the manuscript, which are:

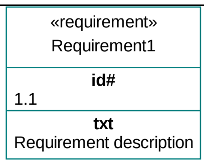
- Requirement diagram & Class diagram
- Block definition diagram & Composite structure diagram
- Internal block diagram
- Activity diagram
- State machine diagram
- Sequence diagram

II. Description of the diagrams used

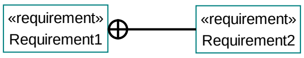
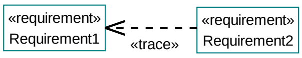
The description of the modeling elements is not exhaustive. It contains only language elements used in this manuscript.

A. Requirement diagram

1. Object descriptions

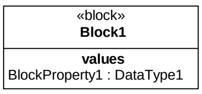
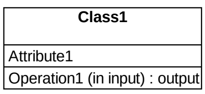
Representation	Name	Description
	Requirement	A requirement is a specification or condition that must (or should) be satisfied. SysML provides modeling constructs to represent text-based requirements and link them to other modeling elements

2. Different kind of links

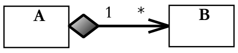
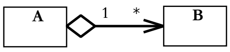
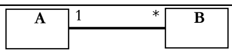
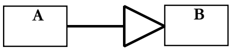
Representation	Name	Description
	Requirement containment relationship	Requirement 1 is composed of Requirement 1
	Derive Dependency	During the requirement analysis, new requirements are generated in connection with the origin requirements. These new requirements can be connected to the initial ones with the deriveReqt dependency. Requirement 2 derived from the Requirement 1.
	Trace Dependency	Lastly, a generic "Trace" dependency can be used to emphasize that a pair of requirements are related in some way or another.

B. Block definition diagram & Class diagram

1. Objects description

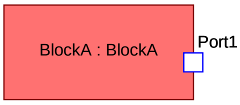
Representation	Name	Description
	Block	Blocks are modular units of system description. Each block defines a collection of features to describe a system or other element of interest. These may include both structural and behavioral features, such as properties and operations, to represent the state of the system and behavior that the system may exhibit.
	Class	Class is based on programming languages: a class is a template definition of the methods and variables in a particular kind of object.

2. Different kind of links

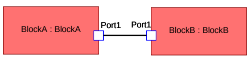
Representation	Name	Description
	Composite aggregation or Composition	A is composed of B
	Shared aggregation or Aggregation	A can be composed of B
	Association	A is connected to B
	Heritage	A inherits from all the properties and operations of B

C. Internal block diagram & Composite structure diagram

1. Object description

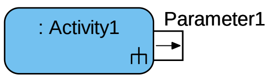
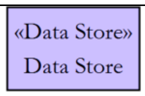
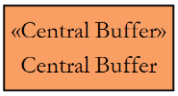
Representation	Name	Description
	Block and Port	Ports are parts available for connection from the outside of the owning block. Ports are typed by interfaces or blocks that define what can be exchanged through them.

2. Different kind of links

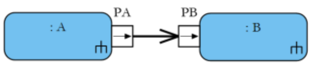
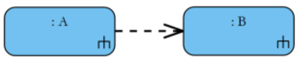
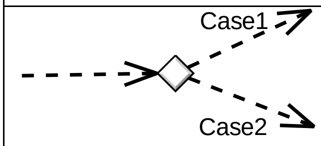
Representation	Name	Description
	Connector	Ports are connected using connectors that represent the kind of flow.

D. Activity diagram

1. Objects description

Representation	Name	Description
	Activity	Activity modeling emphasizes the inputs, outputs, sequences, and conditions for coordinating other behaviors. It provides a flexible link to blocks owning those behaviors.
	Initial Node	Initial Node is the first activity.
	Final Node	Final Node is the last activity.
	Data Store	A Datastore defines permanently stored data.
	Central Buffer node	A Central Buffer Node is an object node that manages flows from multiple sources and destinations

2. Different kind of links

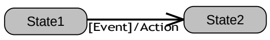
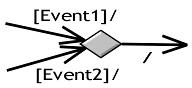
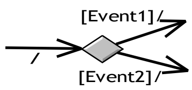
Representation	Name	Description
	Object Flow	Object flow is a path along which objects or data can pass
	Control Flow	A control flow shows the flow of control from activity A to activity B
	Decision Node	A decision Node conducts the control flow to the case adapted according to the condition.

E. State machine diagram

1. Objects description

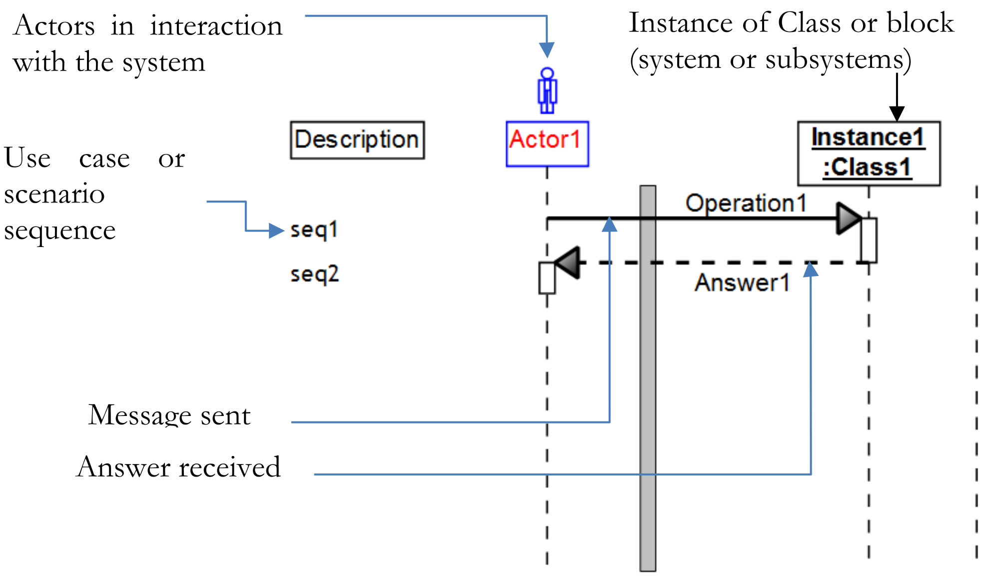
Representation	Name	Description
	State	State of the system that satisfies a certain condition, performs some activity and expects a certain event or condition
	Initial State	Initial State is the first state of the system.
	Final State	Final State is the last state of the system.

2. Different kind of links

Representation	Name	Description
	Transition	A transition between two states is characterized by an event, a guard condition) and an action. Guard condition is the condition to move to the next state, and action is the action executed when moving to the next state
	Junction state	In case of junction of two states, a Junction state allows to move to a new state if one of the two guard condition is validated.
	Junction state	If a transition can result in two states, according to the same guard condition, Junction state can move to the first or the second state depending on the guard condition realized.

F. *Sequence diagram*

The Sequence diagram describes the flows of control between actors and systems (blocks) or between parts of a system. This diagram represents the sending and receiving of messages between the interacting entities called lifelines, where time is represented along the vertical axis. The sequence diagrams can represent highly complex interactions with special constructs to represent various types of control logic, reference interactions on other sequence diagrams, and decomposition of lifelines into their constituent parts.



Appendix 4 – The thermal laws used

I. Conduction 244

A. Calculation of the Laplacian equation in a 3D space 244

1. Hypothesis 244
2. Definition of the system 244
3. Equations 244

B. Solving the Laplacian equation for 3D geometry 246

4. Rectangular parallelepiped in the Cartesian coordinate system 246
5. Cylindrical coordinate system 248
6. Spherical coordinate system 249

II. Convection 252

III. Radiation 254

I. Conduction

A. Calculation of the Laplacian equation in a 3D space

1. Hypothesis

- The steady state is reached, then $\frac{\partial T}{\partial t} = 0$
- The component does not generate power, the heat is exchanged only through its surfaces.
- The thermal conductivity does not depend on the temperature;
- The environment is isotropic. Thus the thermal conductivity does not depend on the direction of heat propagation.

2. Definition of the system

Considering an infinitesimal small volume element in a Cartesian coordinate system (Figure 156):

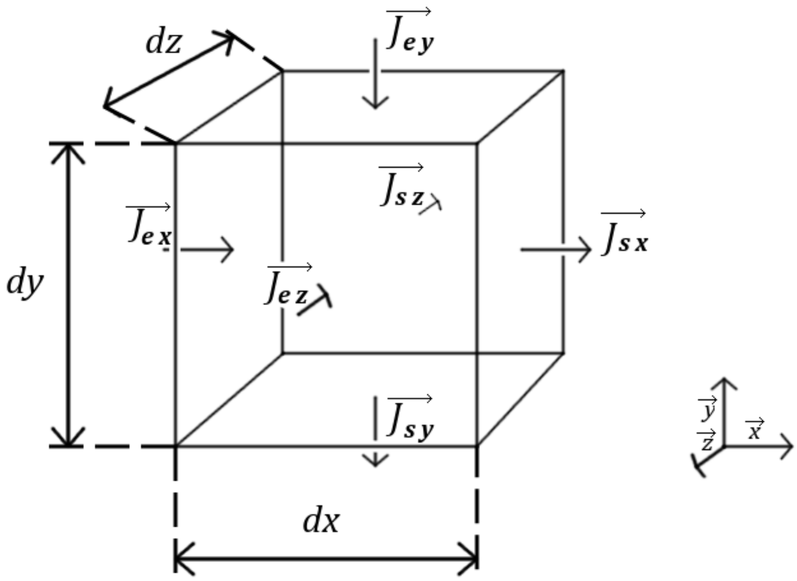


Figure 156: Cartesian coordinate system.

We consider that the input heat is equal to the output heat (the created heat and absorbed heat being null based on the previous hypothesis);

$$\partial Q_e = \partial Q_s \tag{47}$$

3. Equations

Considering the convection modeling, we can calculate the input heat:

$$\partial Q_e = \iint_{dydz} \overrightarrow{J_{e\,x}}.\overrightarrow{n_x} dS_1 + \iint_{dxdz} \overrightarrow{J_{e\,y}}.\overrightarrow{n_y} dS_2 + \iint_{dxdy} \overrightarrow{J_{e\,z}}.\overrightarrow{n_z} dS_3 \tag{48}$$

Using the Fourrier Law, we write:

$$\partial Q_e = \iint_{dydz} -\lambda.\overrightarrow{grad}\big(T(x,y,z)\big).\overrightarrow{n_x}dS_1 + \iint_{dxdz} -\lambda.\overrightarrow{grad}(T(x,y,z)\ ).\overrightarrow{n_y}dS_2 + \iint_{dxdy} -\lambda.\overrightarrow{grad}\big(T(x,y,z)\big).\overrightarrow{n_z}dS_3 \quad (49)$$

After writing and simplifying the expression of the gradient, we write:

$$\partial Q_e = \lambda.\left(\frac{\partial T(x,y,z)}{\partial x}dydz + \frac{\partial T(x,y,z)}{\partial y}dxdz + \frac{\partial T(x,y,z)}{\partial z}dxdy\right) \quad (50)$$

With the same method, we calculate the output heat.

$$\partial Q_s = \iint_{dydz} \overrightarrow{J_{s\,x}}.\overrightarrow{n_x}dS_1 + \iint_{dxdz} \overrightarrow{J_{s\,y}}.\overrightarrow{n_y}dS_2 + \iint_{dxdy} \overrightarrow{J_{s\,z}}.\overrightarrow{n_z}dS_3 \quad (51)$$

Using the Fourrier Law, we write:

$$\partial Q_s = \iint_{dydz} -\lambda.\overrightarrow{grad}\big(T(x+dx\ ,y,z)\big).\overrightarrow{n_x}dS_1 + \iint_{dxdz} -\lambda.\overrightarrow{grad}\big(T(x\ ,y+dy,z)\big).\overrightarrow{n_y}dS_2 + \iint_{dxdy} -\lambda.\overrightarrow{grad}\big(T(x,y,z+dz)\big).\overrightarrow{n_z}dS_3 \quad (52)$$

After writing and simplifying the expression of the gradient, we write:

$$\partial Q_s = \lambda.\left(\frac{\partial T(x+dx,y,z)}{\partial x}dydz + \frac{\partial T(x,y+dy,z)}{\partial y}dxdz + \frac{\partial T(x,y,z+dz)}{\partial z}dxdy\right) \quad (53)$$

Thus we deduce:

$$\frac{\partial T(x+dx,y,z)}{\partial x}dydz + \frac{\partial T(x,y+dy,z)}{\partial y}dxdz + \frac{\partial T(x,y,z+dz)}{\partial z}dxdy = \frac{\partial T(x,y,z)}{\partial x}dydz + \frac{\partial T(x,y,z)}{\partial y}dxdz + \frac{\partial T(x,y,z)}{\partial z}dxdy \quad (54)$$

Furthermore, as the volume considered is small enough, we make the following assumption:

$$\begin{cases} \dfrac{\partial T(x+dx,y,z)}{\partial x} - \dfrac{\partial T(x,y,z)}{\partial x} = \dfrac{\partial^2 T(x,y,z)}{\partial x^2}dx \\ \dfrac{\partial T(x,y+dy,z)}{\partial y} - \dfrac{\partial T(x,y,z)}{\partial y} = \dfrac{\partial^2 T(x,y,z)}{\partial y^2}dy \\ \dfrac{\partial T(x,y,z+dz)}{\partial z} - \dfrac{\partial T(x,y,z)}{\partial z} = \dfrac{\partial^2 T(x,y,z)}{\partial z^2}dz \end{cases} \quad (55)$$

Thus we have:

$$\frac{\partial^2 T(x,y,z)}{\partial x^2}dxdydz + \frac{\partial^2 T(x,y,z)}{\partial y^2}dydxdz + \frac{\partial^2 T(x,y,z)}{\partial z^2}dzdxdy = 0 \quad (56)$$

Simplified by $dxdydz$,

$$\frac{\partial^2 T(x,y,z)}{\partial x^2} + \frac{\partial^2 T(x,y,z)}{\partial y^2} + \frac{\partial^2 T(x,y,z)}{\partial z^2} = \vec{\nabla}^2 \text{T} = 0 \quad (57)$$

This equation is generalizable for other coordinate systems.

$$\vec{\nabla}^2 \text{T} = 0 \quad (58)$$

B. *Solving the Laplacian equation for 3D geometry*

4. Rectangular parallelepiped in the Cartesian coordinate system

We consider a parallelepiped in the rectangle parallelepiped coordinate system. The system is considered as an infinitesimal volume element (Figure 157).

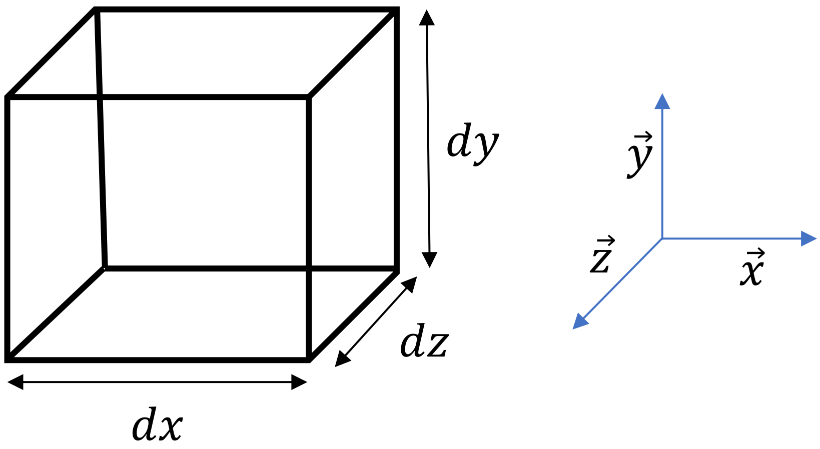


Figure 157 : Infinitesimal volume element in the rectangle parallelepiped coordinate system

We first consider that the material is not isotropic. Thus the general Laplacian equation becomes:

$$\lambda_x.\frac{\partial^2 T}{\partial x^2} + \lambda_y.\frac{\partial^2 T}{\partial y^2} + \lambda_z.\frac{\partial^2 T}{\partial z^2} = 0 \quad (59)$$

Using the separation of variables model, we propose:

$$T(x,y,z) = f(x).g(y).h(z) \quad (60)$$

Then the Laplacian equation in two dimensions becomes:

$$\lambda_x.\frac{1}{f}.\frac{d^2 f}{dx^2} + \lambda_y.\frac{1}{g}.\frac{d^2 g}{dy^2} + \lambda_z.\frac{1}{h}.\frac{d^2 h}{dz^2} = 0 \quad (61)$$

We consider that $\lambda_y.\frac{1}{g}.\frac{d^2 g}{dy^2} + \lambda_z.\frac{1}{h}.\frac{d^2 h}{dz^2}$ is independent of $-\lambda_x.\frac{1}{f}.\frac{d^2 f}{dx^2}$, because it does not depend on the same variables. Therefore we deduce that:

$$\lambda_y.\frac{1}{g}.\frac{d^2g}{dy^2}+\lambda_z.\frac{1}{h}.\frac{d^2h}{dz^2}=-\lambda_x.\frac{1}{f}.\frac{d^2f}{dx^2}=\alpha^2 \quad with\, \alpha \in \mathbb{R}^{+*} \tag{62}$$

Thus we can write two equations:

$$\begin{cases}\lambda_y.\frac{1}{g}.\frac{d^2g}{dy^2}+\lambda_z.\frac{1}{h}.\frac{d^2h}{dz^2}-\alpha^2=0\\ \frac{d^2f}{dx^2}+\frac{\alpha^2}{\lambda_x}f=0\end{cases} \tag{63}$$

Considering that $\lambda_y.\frac{1}{g}.\frac{d^2g}{dy^2}$ is independent of $\lambda_z.\frac{1}{h}.\frac{d^2h}{dz^2}-\alpha^2$, we can introduce another variable $\eta \in \mathbb{R}^{+*}$ and solve f

$$\begin{cases}\frac{d^2h}{dz^2}=\frac{\eta^2+\alpha^2}{\lambda_z}h\\ \frac{d^2g}{dy^2}=-\frac{\eta^2}{\lambda_y}.g\\ f(x)=C_1\cos\left(\frac{\alpha}{\sqrt{\lambda_x}}.x\right)+C_2\sin\left(\frac{\alpha}{\sqrt{\lambda_x}}.x\right)\end{cases} \tag{64}$$

We solve h and g.

$$\begin{cases}h(z)=C_5\cosh\left(\sqrt{\frac{\alpha^2+\eta^2}{\lambda_z}}.z\right)+C_6\sinh\left(\sqrt{\frac{\alpha^2+\eta^2}{\lambda_z}}.z\right)\\ g(y)=C_3\cos\left(\frac{\eta}{\sqrt{\lambda_y}}.z\right)+C_4\sin\left(\frac{\eta}{\sqrt{\lambda_y}}.z\right)\\ f(x)=C_1\cos\left(\frac{\alpha}{\sqrt{\lambda_x}}.x\right)+C_2\sin\left(\frac{\alpha}{\sqrt{\lambda_x}}.x\right)\end{cases} \tag{65}$$

Using the same approach as the two-dimensions analysis, using a projection of the two eigen values η_n and α_k with $(n,k) \in \mathbb{N}^2$, we write:

$$\begin{cases}h_{n,k}(z)=C_{5(n,k)}\cosh\left(\sqrt{\frac{{\alpha_k}^2+{\eta_n}^2}{\lambda_z}}.z\right)+C_{6(n,k)}\sinh\left(\sqrt{\frac{{\alpha_k}^2+{\eta_n}^2}{\lambda_z}}.z\right)\\ g_n(y)=C_{3n}\cos\left(\frac{\eta_n}{\sqrt{\lambda_y}}.z\right)+C_{4n}\sin\left(\frac{\eta_n}{\sqrt{\lambda_y}}.z\right)\\ f_k(x)=C_{1k}\cos\left(\frac{\alpha_k}{\sqrt{\lambda_x}}.x\right)+C_{2k}\sin\left(\frac{\alpha_k}{\sqrt{\lambda_x}}.x\right)\end{cases} \tag{66}$$

Finally, the result of the function is:

$$T(x,y,z)=\sum_{k=0}^{+\infty}\sum_{n=0}^{+\infty}f_k(x).g_n(y).h_{n,k}(z) \tag{67}$$

5. Cylindrical coordinate system

We consider a cylinder in the cylindrical coordinates system. The system is considered as an infinitesimal volume element (Figure 158).

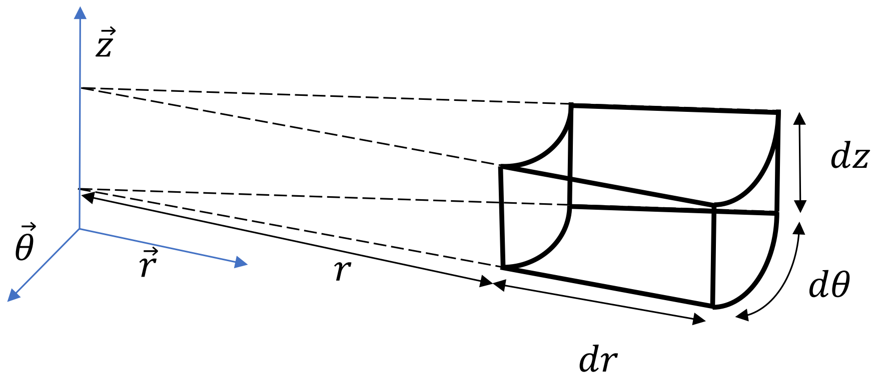


Figure 158 : Cylindrical coordinate system on an infinitesimal volume element.

The Laplacian is written in the cylindrical coordinate system:

$$\left(\frac{\partial^2 T}{\partial r^2}+\frac{1}{r}.\frac{\partial T}{\partial r}\right)+\frac{1}{r^2}.\frac{\partial^2 T}{\partial \theta^2}+\frac{\partial^2 T}{\partial z^2}=0 \tag{68}$$

Using the separation of the variables method, we write:

$$g(\theta).h(z).\left(\frac{d^2 f(r)}{dr^2}+\frac{1}{r}.\frac{df(r)}{dr}\right)+\frac{f(r).h(z)}{r^2}.\frac{d^2 g(\theta)}{d\theta^2}+f(r).g(\theta).\frac{d^2 h(z)}{dz^2}=0 \tag{69}$$

Simplifying by $f(r).g(\theta).h(z)$, the equations become:

$$\frac{1}{f(r)}.\left(\frac{d^2 f(r)}{dr^2}+\frac{1}{r}.\frac{df(r)}{dr}\right)+\frac{1}{g(\theta)}\frac{1}{r^2}.\frac{d^2 g(\theta)}{d\theta^2}+\frac{1}{h(z)}.\frac{d^2 h(z)}{dz^2}=0 \tag{70}$$

We consider that $\frac{1}{f(r)}.\left(\frac{d^2 f(r)}{dr^2}+\frac{1}{r}.\frac{df(r)}{dr}\right)+\frac{1}{g(\theta)}\frac{1}{r^2}.\frac{d^2 g(\theta)}{d\theta^2}$ is independent of $\frac{1}{h(z)}.\frac{d^2 h(z)}{dz^2}$ then, we introduce a constant $\eta \in \mathbb{R}^{+*}$ such that:

$$\frac{1}{f(r)}.\left(\frac{d^2 f(r)}{dr^2}+\frac{1}{r}.\frac{df(r)}{dr}\right)+\frac{1}{g(\theta)}\frac{1}{r^2}.\frac{d^2 g(\theta)}{d\theta^2}=-\frac{1}{h(z)}.\frac{d^2 h(z)}{dz^2}=\eta^2 \tag{71}$$

Thus we write two equations:

$$\Rightarrow \begin{cases} \frac{1}{f(r)}.\left(\frac{d^2 f(r)}{dr^2}+\frac{1}{r}.\frac{df(r)}{dr}\right)+\frac{1}{g(\theta)}\frac{1}{r^2}.\frac{d^2 g(\theta)}{d\theta^2}=\eta^2 \\ \frac{d^2 h(z)}{dz^2}+\eta^2.h(z)=0 \end{cases} \quad with\,\eta \in \mathbb{R}^{+*} \tag{72}$$

Moreover, we introduce a new constant $\nu \in \mathbb{R}^{+*}$ because $\frac{r^2}{f(r)}.\left(\frac{d^2 f(r)}{dr^2}+\frac{1}{r}.\frac{df(r)}{dr}\right)-\eta r^2$ is independent of $-\frac{1}{g(\theta)}\frac{d^2 g(\theta)}{d\theta^2}$ and we solve the second equation:

$$\begin{cases} \frac{r^2}{f(r)} \cdot \left(\frac{d^2 f(r)}{dr^2} + \frac{1}{r} \cdot \frac{df(r)}{dr} \right) - \eta r^2 = -\frac{1}{g(\theta)} \frac{d^2 g(\theta)}{d\theta^2} = \nu^2 \, with \, \eta \in \mathbb{R}^{+*} \\ h(z) = C_1 \cos(\eta . z) + C_2 \sin(\eta . z) \end{cases} \quad (73)$$

Therefore three equations are introduced:

$$\Rightarrow \begin{cases} \frac{d^2 f(r)}{dr^2} + \frac{1}{r} \cdot \frac{df(r)}{dr} - \left(\eta + \frac{\nu}{r^2} \right) f(r) = 0 \\ \frac{d^2 g(\theta)}{d\theta^2} + \nu^2 . g(\theta) = 0 \qquad with \, (\eta, \nu) \in \mathbb{R}^{+*2} \\ h(z) = C_1 \cos(\eta . z) + C_2 \sin(\eta . z) \end{cases} \quad (74)$$

The solution of these equations is given as follows:

$$\Rightarrow \begin{cases} f(r) = C_5 . I_\nu(\eta . r) + C_7 . K_\nu(\eta . r) \\ g(\theta) = C_3 \cos(\nu . \theta) + C_4 \sin(\nu . \theta) \, with \, (\eta, \nu) \in \mathbb{R}^{+*2} \\ h(z) = C_1 \cos(\eta . z) + C_2 \sin(\eta . z) \end{cases} \quad (75)$$

Where $I_\nu(\eta . r)$ is the modified Bessel function of the first kind and of the ν order and $K_\nu(\eta . r)$ is the modified Bessel function of the second kind and of the ν order.

Like the previous equations, as $g(\theta)$ and $h(z)$ are trigonometric functions, the solution can be projected in different eigen bases, according to the boundary conditions.

We write

$$\begin{cases} f_{n,k}(r) = C_{5n,k} . I_{\nu_k}(\eta_n . r) + C_{7n,k} . K_{\nu_k}(\eta_n . r) \\ g_k(\theta) = C_{3k} \cos(\nu_k . \theta) + C_{4k} \sin(\nu_k . \theta) \\ h_n(z) = C_{1n} \cos(\eta_n . z) + C_{2n} \sin(\eta_n . z) \end{cases} \quad (76)$$

Finally, the result of the function is

$$T(r, \theta, z) = \sum_{k=0}^{+\infty} \sum_{n=0}^{+\infty} f_{n,k}(r) . g_k(\theta) . h_n(z) \quad (77)$$

6. Spherical coordinate system

In the spherical coordinate system, we consider a sphere as an infinitesimal volume element (Figure 159).

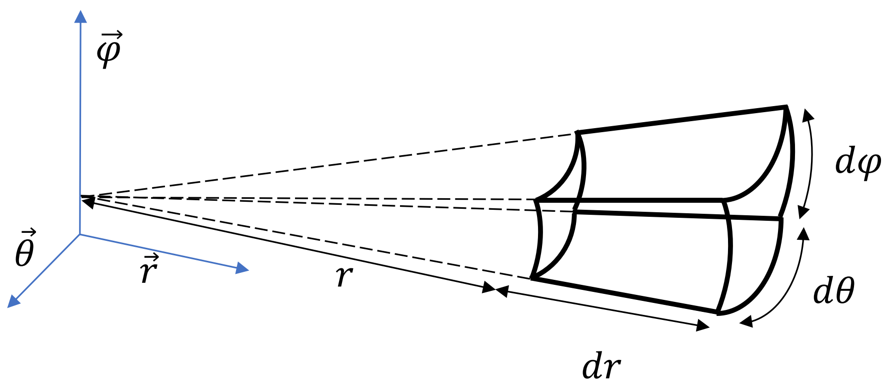


Figure 159: Spherical coordinate system infinitesimal volume element.

For the spherical coordinates system, we have to proceed to a change of variable. For this reason, we will integrate the whole system equation:

$$\frac{\partial^2 T}{\partial r^2} + \frac{2}{r}.\frac{\partial T}{\partial r} + \frac{1}{r^2 \sin(\theta)}\frac{\partial}{\partial \theta}\left(\sin(\theta).\frac{\partial T}{\partial \theta}\right) + \frac{1}{r^2 \sin^2(\theta)}.\frac{\partial^2 T}{\partial \varphi^2} = 0 \tag{78}$$

Then, we will consider that $\mu = \cos(\theta)$

$$\frac{\partial^2 T}{\partial r^2} + \frac{2}{r}.\frac{\partial T}{\partial r} + \frac{1}{r^2}\frac{\partial}{\partial \mu}\left((1-\mu^2).\frac{\partial T}{\partial \mu}\right) + \frac{1}{r^2(1-\mu^2)}.\frac{\partial^2 T}{\partial \varphi^2} = 0 \tag{79}$$

Finally, we consider that $V(r,\mu,\varphi) = \sqrt{r}.T(r,\mu,\varphi)$

$$\frac{\partial^2 V}{\partial r^2} + \frac{1}{r}.\frac{\partial V}{\partial r} - \frac{1}{4}.\frac{V}{r^4} + \frac{1}{r^2}\frac{\partial}{\partial \mu}\left((1-\mu^2).\frac{\partial V}{\partial \mu}\right) + \frac{1}{r^2(1-\mu^2)}.\frac{\partial^2 V}{\partial \varphi^2} = 0 \tag{80}$$

Now, it is possible to proceed to the separation of variables:

$$V(r,\mu,\varphi) = f(r).g(\mu).h(\varphi) \tag{81}$$

The expression of the equation starts with:

$$\frac{1}{f(r)}\left(\frac{d^2 f(r)}{dr^2} + \frac{1}{r}.\frac{df(r)}{dr} - \frac{1}{4}.\frac{f(r)}{r^4}\right) + \frac{1}{r^2 g(\mu)}\frac{d}{d\mu}\left((1-\mu^2).\frac{dg(\mu)}{d\mu}\right) + \frac{1}{h(\varphi)}.\frac{1}{r^2(1-\mu^2)}.\frac{d^2 h(\varphi)}{d\varphi^2} = 0 \tag{82}$$

This equation is multiplied by $r^2.(1-\mu^2)$

$$\frac{r^2.(1-\mu^2)}{f(r)}\left(\frac{d^2 f(r)}{dr^2} + \frac{1}{r}.\frac{df(r)}{dr} - \frac{1}{4}.\frac{f(r)}{r^4}\right) + \frac{(1-\mu^2)}{g(\mu)}\frac{d}{d\mu}\left((1-\mu^2).\frac{dg(\mu)}{d\mu}\right) + \frac{1}{h(\varphi)}.\frac{d^2 h(\varphi)}{d\varphi^2} = 0 \tag{83}$$

Thus: $\frac{r^2.(1-\mu^2)}{f(r)}\left(\frac{d^2 f(r)}{dr^2} + \frac{1}{r}.\frac{df(r)}{dr} - \frac{1}{4}.\frac{f(r)}{r^4}\right) + \frac{(1-\mu^2)}{g(\mu)}\frac{d}{d\mu}\left((1-\mu^2).\frac{dg(\mu)}{d\mu}\right)$ is independent of $\frac{1}{h(\varphi)}.\frac{d^2 h(\varphi)}{d\varphi^2}$, and we introduce a new constant $m \in \mathbb{R}^{+*}$ such that

$$\frac{r^2.(1-\mu^2)}{f(r)}\left(\frac{d^2 f(r)}{dr^2} + \frac{1}{r}.\frac{df(r)}{dr} - \frac{1}{4}.\frac{f(r)}{r^4}\right) + \frac{(1-\mu^2)}{g(\mu)}\frac{d}{d\mu}\left((1-\mu^2).\frac{dg(\mu)}{d\mu}\right) = -\frac{1}{h(\varphi)}.\frac{d^2 h(\varphi)}{d\varphi^2} = m^2 \tag{84}$$

The result is two equations:

$$\begin{cases} \frac{r^2.(1-\mu^2)}{f(r)}\left(\frac{d^2 f(r)}{dr^2} + \frac{1}{r}.\frac{df(r)}{dr} - \frac{1}{4}.\frac{f(r)}{r^4}\right) + \frac{(1-\mu^2)}{g(\mu)}\frac{d}{d\mu}\left((1-\mu^2).\frac{dg(\mu)}{d\mu}\right) = m^2 \\ -\frac{1}{h(\varphi)}.\frac{d^2 h(\varphi)}{d\varphi^2} = m^2 \end{cases} \tag{85}$$

It is possible to simplify the first equation by $(1-\mu^2)$

$$\begin{cases} \dfrac{r^2}{f(r)}\left(\dfrac{d^2f(r)}{dr^2}+\dfrac{1}{r}.\dfrac{df(r)}{dr}-\dfrac{1}{4}.\dfrac{f(r)}{r^4}\right)+\dfrac{1}{g(\mu)}\dfrac{d}{d\mu}\left((1-\mu^2).\dfrac{dg(\mu)}{d\mu}\right)-\dfrac{m^2}{1-\mu^2}=0 \\ \dfrac{d^2h(\varphi)}{d\varphi^2}+m^2.h(\varphi)=0 \end{cases} \tag{86}$$

We introduce another variable $n \in \mathbb{R}^{+*}$ and we consider that $\frac{r^2}{f(r)}\left(\frac{d^2f(r)}{dr^2}+\frac{1}{r}.\frac{df(r)}{dr}-\frac{1}{4}.\frac{f(r)}{r^4}\right)$ is independent of $\frac{1}{g(\mu)}\frac{d}{d\mu}\left((1-\mu^2).\frac{dg(\mu)}{d\mu}\right)-\frac{m^2}{1-\mu^2}$ and we write:

$$\begin{cases} -\dfrac{r^2}{f(r)}\left(\dfrac{d^2f(r)}{dr^2}+\dfrac{1}{r}.\dfrac{df(r)}{dr}-\dfrac{1}{4}.\dfrac{f(r)}{r^4}\right)=\dfrac{1}{g(\mu)}\dfrac{d}{d\mu}\left((1-\mu^2).\dfrac{dg(\mu)}{d\mu}\right)-\dfrac{m^2}{1-\mu^2}=n.(n+1) \\ h(\varphi)=C_1.\cos(m.\varphi)+C_2.\sin(m.\varphi) \end{cases} \tag{87}$$

This generates two new equations. Thus we have three differential equations to solve.

$$\begin{cases} -\dfrac{r^2}{f(r)}\left(\dfrac{d^2f(r)}{dr^2}+\dfrac{1}{r}.\dfrac{df(r)}{dr}-\dfrac{1}{4}.\dfrac{f(r)}{r^4}\right)=n.(n+1) \\ \dfrac{1}{g(\mu)}\dfrac{d}{d\mu}\left((1-\mu^2).\dfrac{dg(\mu)}{d\mu}\right)-\dfrac{m^2}{1-\mu^2}=n.(n+1) \\ h(\varphi)=C_1.\cos(m.\varphi)+C_2.\sin(m.\varphi) \end{cases} \tag{88}$$

$$\begin{cases} r^2\dfrac{d^2f(r)}{dr^2}+r.\dfrac{df(r)}{dr}+\left(n.(n+1)-\dfrac{1}{4}\right)f(r)=0 \\ \dfrac{d}{d\mu}\left((1-\mu^2).\dfrac{dg(\mu)}{d\mu}\right)+(n.(n+1)+\dfrac{m^2}{(1-\mu^2)}).g(\mu)=0 \\ h(\varphi)=C_1.\cos(m.\varphi)+C_2.\sin(m.\varphi) \end{cases} \tag{89}$$

Each differential equation is solved independently.

$$\begin{cases} f(r)=C_5.\sin\left(\sqrt{n.(n+1)-\dfrac{1}{4}}.\log(r)\right)+C_6.\cos\left(\sqrt{n.(n+1)-\dfrac{1}{4}}.\log(r)\right) \\ g(\mu)=C_3\,.P_n^m(\mu)+C_4\,.Q_n^m(\mu) \\ h(\varphi)=C_1.\cos(m.\varphi)+C_2.\sin(m.\varphi) \end{cases} \tag{90}$$

Where $P_n^m(\mu)$ and $Q_n^m(\mu)$ are the Legendre polynomials.

Like the previous equations, as $f(r)$ and $h(\varphi)$ are trigonometric functions, the solution can be projected in different eigen bases n_i, and m_j, according to the boundary conditions.

We write:

$$\begin{cases} f_i(r)=C_{5i}.\sin\left(\sqrt{n_i.(n_i+1)-\dfrac{1}{4}}.\log(r)\right)+C_{6i}.\cos\left(\sqrt{n_i.(n_i+1)-\dfrac{1}{4}}.\log(r)\right) \\ g_{i,j}(\mu)=C_{3i,j}\,.P_{n_i}^{m_j}(\mu)+C_{4i,j}\,.Q_{n_i}^{m_j}(\mu) \\ h_j(\varphi)=C_{1j}.\cos(m_j.\varphi)+C_{2j}.\sin(m_j.\varphi) \end{cases} \tag{91}$$

Finally, the result of the function is

$$T(r,\theta,\varphi) = \sum_{i=0}^{+\infty}\sum_{j=0}^{+\infty} f_i(r).g_{i,j}(\mu).h_j(\varphi)$$

II. Convection

For the natural convection, the Reynold number is not adapted, because the fluid velocity is not calculated. Therefore the Reynold's number is replaced by the Grashof number. This number is more adapted to the natural convection.

Table 59 Physical meaning and equation of the Grashof number.

Name	Physical meaning	Formula
Grashof number	The Grashof number is the ratio of the buoyance force to the viscous force. The buoyance force is due to the spatial variation in fluid density (due to temperature differences). The viscosity of a fluid can be considered as the resistance of fluid to movement. Thus a high difference of fluid density with low fluid resistance will imply a considerable fluid movement (in this case, the Grashof number is important). On the contrary, a small Grashof number implies a weak displacement of fluid. This number is used in natural convection to estimate the displacement of the fluid.	$Gr = \frac{L^3\,\rho^2\,g\,\beta\,\Delta T}{\mu^3}$

Table 60 describes the Nusselt correlation for the natural convection. Concerning natural convection, the angle of the element compared to gravity is very important, because the movement of the fluid depends on the direction of the geometrical element. Thus the Nusselt correlation depends on this angle for a solid compared to the gravity.

Table 60 : Calculation of the Nusselt correlation for natural convection

Geometry	Calculation	Condition
Horizontal plate	$Nu = 0.54.Ra^{\frac{1}{4}}$	-$10^4 \leq Ra \leq 10^7$ -$Pr \geq 0.7$
	$Nu = 0.15.Ra^{\frac{1}{3}}$	-$10^7 \leq Ra \leq 10^{11}$ -all Pr
Plate inclined with an angle α	$Nu = \left[0.825 + \frac{0.387.(Ra.\cos(\alpha))^{\frac{1}{6}}}{\left(1+\left(\frac{0.492}{Pr}\right)^{\frac{9}{16}}\right)^{\frac{8}{27}}}\right]^2$	-all Ra -all Pr -$0 \leq \alpha \leq 60°$
Horizontal Cylinder	$Nu = \left[0.60 + \frac{0.387.(Ra)^{\frac{1}{6}}}{\left(1+\left(\frac{0.559}{Pr}\right)^{\frac{9}{16}}\right)^{\frac{8}{27}}}\right]^2$	-$Ra \leq 10^{12}$
Sphere	$Nu = 2 + \frac{0.589.Ra^{\frac{1}{4}}}{\left(1+\left(\frac{0.469}{Pr}\right)^{\frac{9}{16}}\right)^{\frac{4}{9}}}$	-$Ra \leq 10^{11}$ -$Pr \geq 0.7$

III. Radiation

Concerning the radiation modeling, we propose to model a semi-transparent solid, as described in Figure 160, then the incident radiation heat is divided into three parts:

- the first part is the reflected heat,
- the second part is the absorbed heat,
- the last part is the transmitted heat.

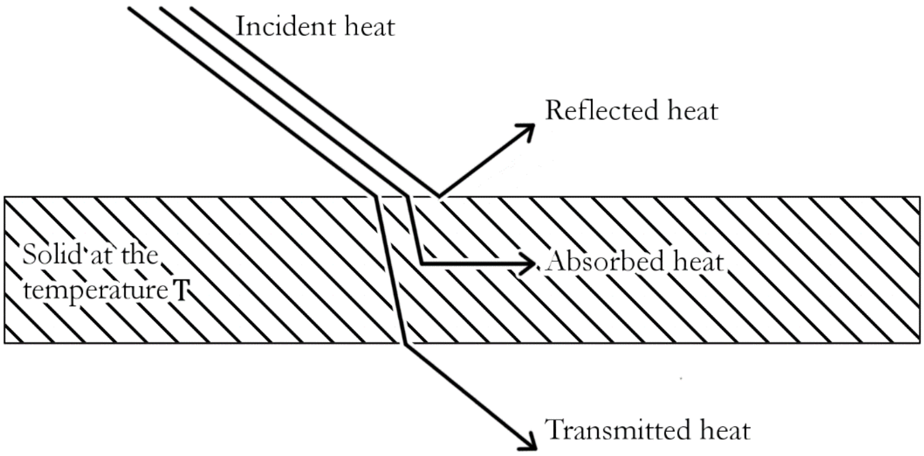


Figure 160 Transfer of electromagnetic radiation by a solid.

In order to consider the radiation of a semi-transparent solid, three ratios will be considered, as described in Table 61.

Table 61: Definition of radiation parameters.

Description	Equation	Variable used
a_s is the absorption capacity	$a_s = \frac{E_a}{E_i}$	• E_a is the absorbed heat • E_i is the incident heat
r_f is the reflection capacity	$r_f = \frac{E_r}{E_i}$	• E_r is the reflected heat • E_i is the incident heat
t_a is the transmission capacity	$t_a = \frac{E_t}{E_i}$	• E_t is the transmitted heat • E_i is the incident heat

These ratios depend on different parameters, such as the material, the surface conditions (e.g. roughness), and the temperature.

According to the heat conservation law, we write:

$$a_s + r_f + t_a = 1 \tag{93}$$

When considering an opaque material, i.e. the medium is considered as opaque, the transmission capacity is null: $t_a = 0$ and using this condition, we write:

$$a_s + r_f = 1 \tag{94}$$

The term of reflected heat has to be added to the heat equation. In order to define the radiosity is defined as:

$$J = \epsilon.\sigma.{T_s}^4 + r_f.E_i \tag{95}$$

Where E_i is the incident heat.

Finally,

$$\phi_{er} = F_{e\to r}.\mathrm{J}_e.S_e - F_{r\to e}.\mathrm{J}_r.S_r \tag{96}$$

It is possible to generalize the problem to n emitting elements towards one receiver element:

$$\phi_{er} = \sum_{j=0}^{n} F_{e\to rj}.\mathrm{J}_e.S_e - F_{rj\to e}.\mathrm{J}_{rj}.S_{rj} \tag{97}$$

References

AFNet, 2015. *STEP AP 242 benchmark.* [Online] Available at: http://afnet.fr/dotank/sps/ap242benchmark/ [Accessed 09 03 2017].

Albers, A. et al., 2010. *Contact and Channel Modelling Using Part and Function Libraries in a Function-Based Design Approach.* s.l., s.n., pp. 393-404.

Albers, A. & Zingel, C., 2011. *Interdisciplinary Systems Modeling Using the Contact & Channel-model for SysML.* s.l., s.n.

Albers, A. & Zingel, C., 2013. Extending SysML for Engineering Designers by Integration of the Contact & Channel--Approach (C&C 2-A) for Function-Based Modeling of Technical Systems. *Procedia Computer Science,* Volume 16, pp. 353-362.

Al-Fedaghi, S., 2016. Function-Behavior-Structure Model of Design: An Alternative Approach. *INTERNATIONAL JOURNAL OF ADVANCED COMPUTER SCIENCE AND APPLICATIONS,* Volume 7, pp. 133-139.

Ameri, A. & Felske, J. D., 1982. Radiation configuration factors for obliquely oriented finite length circular cylinders. *International Journal of Heat and Mass Transfer,* Volume 25, pp. 728-736.

Andrianarison, E. & Piques, J.-D., 2010. SysML for embedded automotive Systems: a practical approach. *Embedded Real Time Software and Systems ERTS.*

Anthes, C., Heinzlreiter, P. & Volkert, J., 2004. *An adaptive network architecture for close-coupled collaboration in distributed virtual environments.* s.l., s.n., pp. 382-385.

Austin, S. et al., 2001. Mapping the conceptual design activity of interdisciplinary teams. *Design studies,* Volume 22, pp. 211-232.

Baysal, M. M. et al., 2005. *Product information exchange using open assembly model: Issues related to representation of geometric information.* s.l., s.n., pp. 601-612.

Ben Hamida, S., Jankovic, M., Huet, A. & Bocquet, J.-C., 2016. *ValYOU – Towards a Design-to-Value approach.* [Online] Available at: ValYOU – Towards a Design-to-Value approach

Bergman, T. L. & Incropera, F. P., 2011. *Introduction to heat transfer.* s.l.:John Wiley & Sons.

Bohnke, D., Reichwein, A. & Rudolph, S., 2009. *Design language for airplane geometries using the unified modeling language.* s.l., s.n., pp. 661-670.

Botto, D., Zucca, S., Gola, M. M. & Salvano, S., 2002. *A method for on-line temperature calculation of aircraft engine turbine discs.* s.l., s.n., pp. 49-55.

Boujut, J.-F. & Laureillard, P., 2002. A co-operation framework for product--process integration in engineering design. *Design studies,* Volume 23, pp. 497-513.

Buchenrieder, K., 1993. Codesign and concurrent engineering. *Computer,* Volume 26, pp. 85-86.

Buckingham, E., 1914. On physically similar systems; illustrations of the use of dimensional equations. *Physical review,* Volume 4, p. 345.

Cabot, J. & Gogolla, M., 2012. Object constraint language (OCL): a definitive guide. In: *Formal methods for model-driven engineering.* s.l.:Springer, pp. 58-90.

Cao, Y., Liu, Y., Fan, H. & Fan, B., 2013. SysML-based uniform behavior modeling and automated mapping of design and simulation model for complex mechatronics. *Computer-Aided Design,* Volume 45, pp. 764-776.

Chandrasegaran, S. K. et al., 2013. The evolution, challenges, and future of knowledge representation in product design systems. *Computer-aided design,* Volume 45, pp. 204-228.

Chase, S. C. & Liew, P., 2001. A framework for redesign using FBS models and grammar adaptation. In: *Computer Aided Architectural Design Futures 2001.* s.l.:Springer, pp. 467-477.

Chen, M., Hammami, O. & Callot, M., 2014. *Architecture framework associated to vehicle architecture definition. sl.* s.l.:CESUN.

Cheutet, V., 2006. *Towards semantic modelling of free-form mechanicals products,* s.l.: s.n.

Clayton, M. J., Kunz, J. C. & Fischer, M. A., 1996. Rapid conceptual design evaluation using a virtual product model. *Engineering Applications of Artificial Intelligence,* Volume 9, pp. 439-451.

Clement, A., Riviere, A. & Serre, P., 1996. A declarative information model for functional requirements. In: *Computer-aided Tolerancing.* s.l.:Springer, pp. 3-16.

Clement, A., Riviere, A., Serre, P. & Valade, C., 1998. The TTRSs: 13 constraints for dimensioning and tolerancing. In: *Geometric design tolerancing: theories, standards and applications.* s.l.:Springer, pp. 122-131.

Cubélès Valade, C.-I., 1998. *Contribution à la synthèse des languages déclaratifs de spécifications géométriques,* s.l.: s.n.

Cutting, D. & Noppen, J., 2015. An Extensible Benchmark and Tooling for Comparing Reverse Engineering Approaches. *International Journal on Advances in Software,* Volume 8, pp. 115-124.

Czarnecki, K. & Helsen, S., 2003. *Classification of model transformation approaches.* s.l., s.n., pp. 1-17.

Dantan, J.-Y., ANWER, N. & MATHIEU, L., 2003. Integrated tolerancing process for conceptual design. *CIRP Annals-Manufacturing Technology,* Volume 52, pp. 135-138.

Dassault systemes, 2017. *3D experience platform - Dassault Systèmes 3DS.* [Online]
Available at: https://www.3ds.com/products-services/3dexperience/
[Accessed 09 03 2017].

Dassault systemes, 2017. *Multi-Engineering Modeling and Simulation - Dymola - CATIA.* [Online]
Available at: https://www.3ds.com/products-services/catia/products/dymola/
[Accessed 09 03 2017].

Dassault systems, 2017. *Solidworks.* [Online]
Available at: http://www.solidworks.fr/sw/education/education-edition-2013-2014-overview.htm
[Accessed 09 03 2017].

De Micheli, G., 1993. Extending CAD tools and techniques. *Computer,* 26(1), pp. 85-87.

De Monte, F., 2004. Transverse eigenproblem of steady-state heat conduction for multi-dimensional two-layered slabs with automatic computation of eigenvalues. *International Journal of Heat and Mass Transfer,* Volume 47, pp. 191-201.

Demoly, F. et al., 2010. Multiple viewpoint modelling framework enabling integrated product--process design. *International Journal on Interactive Design and Manufacturing (IJIDeM),* Volume 4, pp. 269-280.

Détienne, F., Martin, G. & Lavigne, E., 2005. Viewpoints in co-design: a field study in concurrent engineering. *Design studies,* Volume 26, pp. 215-241.

DODSMC, D., 2001. *Systems Engineering Fundamentals.* s.l.:DOD Departementof Defense (USA).

Do, E. Y.-L., Gross, M. D., Neiman, B. & Zimring, C., 2000. Intentions in and relations among design drawings. *Design Studies,* Volume 21, pp. 483-503.

Eclipse, 2017. *Papyrus.* [Online] Available at: http://www.eclipse.org/papyrus/ [Accessed 09 03 2017].

Eichelberger, H., Eldogan, Y., Schmid, K. & Platz, M., 2009. A Comprehensive Survey of UML Compliance in Current Modelling Tools.. *Software Engineering,* Volume 143, pp. 39-50.

Estefan, J. A., 2007. Survey of Model-Based Systems Engineering (MBSE) methodologies. *Incose MBSE Focus Group,* Volume 25, p. 8.

Feingold, A. & Gupta, K. G., 1969. *New Analytical Approach to the Evaluation of Configuration Factors in Radiation from Spheres in Infinitely Long Cylinders.* s.l., s.n.

Felske, J. D., 1978. Approximate radiation shape factors between two spheres. *Journal of Heat Transfer,* Volume 100, pp. 547-548.

Feng, S. C., Nederbragt, W., Kaing, S. & Sriram, R. D., 1999. *Incorporating process planning into conceptual design.* s.l., s.n.

Fontaine, G. & Hammami, O., 2016. Simulation model backbone generation through Model Identity Card graph representation. *CESUN conference.*

Fowler, J., 1995. STEP for data management, exchange and sharing.

FreeCAD, 2017. *FreeCAD: An open-source parametric 3D CAD modeler.* [Online] Available at: http://freecadweb.org/ [Accessed 09 03 2017].

FreeCAD-community, 2015. *FreeCAD.* s.l.:s.n.

Friendenthal, S., Steiner, R. & Moore, A., 2009. A practical Guide to SysML.

Gero, J. S. & Kannengiesser, U., 2000. *Towards a situated Function-Behaviour-Structure framework as the basis for a theory of designing.* s.l., s.n.

Göknur, S. et al., 2015. A Model Identity Card to Support Engineering Analysis Model (EAM) Development Process in a Collaborative Multidisciplinary Design Environment. *IEEE Systems Journal,* Volume 9, pp. 1151-1162.

Graignic, P. et al., 2013. Complex system simulation: proposition of a MBSE framework for design-analysis integration. *Procedia Computer Science,* Volume 16, pp. 59-68.

Gross, J., Messe, C. & Rudolph, S., 2012. *A model based thermal systems engineering approach.* s.l., s.n., pp. 17-19.

Gross, J. & Rudolph, S., 2016. Geometry and simulation modeling in design languages. *Aerospace Science and Technology,* Volume 54, pp. 183-191.

Guerrero-Martinez, F. J., Younger, P. L., Karimi, N. & Kyriakis, S., 2017. Three-dimensional numerical simulations of free convection in a layered porous enclosure. *International Journal of Heat and Mass Transfer,* Volume 106, pp. 1005-1013.

Gupta, S. & Okudan, G. E., 2008. Computer-aided generation of modularised conceptual designs with assembly and variety considerations. *Journal of Engineering Design,* Volume 19, pp. 533-551.

Hagel, J., Durchslag, S. & Seely Brown, J., 2002. Orchestrating loosely coupled business processes: The secret to successful collaboration. *Paper, Copyright Hagel, Brown, Durchslag.*

Hahn, A. S., 2010. *Vehicle sketch pad: a parametric geometry modeler for conceptual aircraft design.* s.l., s.n., pp. 2010-657.

Hahn, D. W. & Özişik, M. N., 2012. Heat Conduction Fundamentals. *Heat Conduction, Third Edition,* pp. 1-39.

Haji-Sheikh, A., Beck, J. V. & Agonafer, D., 2003. Steady-state heat conduction in multi-layer bodies. *International Journal of Heat and Mass Transfer,* Volume 46, pp. 2363-2379.

Hammond, M., 2015. *Pywin32.* s.l.:s.n.

Harvey, F., 2001. Constructing GIS: actor networks of collaboration. *URISA journal,* Volume 13, pp. 29-37.

Hassan, A., Siadat, A., Dantan, J.-Y. & Martin, P., 2010. Conceptual process planning--an improvement approach using QFD, FMEA, and ABC methods. *Robotics and Computer-Integrated Manufacturing,* Volume 26, pp. 392-401.

Heimbigner, D. & McLeod, D., 1985. A federated architecture for information management. *ACM Transactions on Information Systems (TOIS),* Volume 3, pp. 253-278.

Holt, J. & Perry, S., 2008. *SysML for systems engineering.* s.l.:IET.

Howell, J. R., 2014. *A CATALOG OF RADIATION HEAT TRANSFER CONFIGURATION FACTORS.* [Online] Available at: http://www.thermalradiation.net/tablecon.html [Accessed 09 03 2017].

Hsu, W., Fuh, Y. H. J. & Zhang, Y., 1998. Synthesis of design concepts from a design for assembly perspective. *Computer Integrated Manufacturing Systems,* Volume 11, pp. 1-13.

Hsu, W. & Woon, I. M. Y., 1998. Current research in the conceptual design of mechanical products. *Computer-Aided Design,* Volume 30, pp. 377-389.

Hughes, J. F., Van Dam, A., Foley, J. D. & Feiner, S. K., 2014. *Computer graphics: principles and practice.* s.l.:Pearson Education.

IBM, 2017. *IBM Rational Rhapsody Developer.* [Online] Available at: http://www-03.ibm.com/software/products/fr/ratirhap/ [Accessed 09 03 2017].

International Standard Organization, 2000. *Industrial automation systems and integration -- Product data representation and exchange -- Part 513: Application Interpreted Construct: Elementary Boundary Representation.* s.l.:s.n.

International Standard Organization, 2005. *Geometrical product specifications (GPS) -- General concepts -- Part 1: Model for geometrical specification and verification.* s.l.:s.n.

JModelica, 2015. *JModelica.org.* [Online] Available at: http://jmodelica.org/ [Accessed 09 03 2017].

Jouault, F., Allilaire, F., Bézivin, J. & Kurtev, I., 2008. ATL: A model transformation tool. *Science of computer programming,* Volume 72, pp. 31-39.

Kada, M., 2006. 3D building generalization based on half-space modeling. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences,* Volume 36, pp. 58-64.

Kapos, G.-D.et al., 2014. *Model-based system engineering using SysML: Deriving executable simulation models with QVT.* s.l., s.n., pp. 531-538.

Kappel, G. et al., 2012. Model transformation by-example: a survey of the first wave. In: *Conceptual Modelling and Its Theoretical Foundations.* s.l.:Springer, pp. 197-215.

Kellner, A., Hehenberger, P., Weingartner, L. & Friedl, M., 2015. *Design and use of system models in mechatronic system design.* s.l., s.n., pp. 142-149.

Kern, H., 2014. *Study of Interoperability between meta-modeling tools.* s.l., s.n., pp. 1629-1637.

Kim, S. H., 2014. Automating building energy system modeling and analysis: An approach based on SysML and model transformations. *Automation in Construction,* Volume 41, pp. 119-138.

Kleiner, S. & Kramer, C., 2013. Model based design with systems engineering based on RFLP using V6. In: *Smart Product Engineering.* s.l.:Springer, pp. 93-102.

Komoto, H. & Tomiyama, T., 2012. A framework for computer-aided conceptual design and its application to system architecting of mechatronics products. *Computer-Aided Design,* Volume 44, pp. 931-946.

Kossiakoff, A., Sweet, W. N., Seymour, S. J. & Biemer, S. M., 2011. *Systems engineering principles and practice.* s.l.:John Wiley & Sons.

Kramer, G. A., 1991. Using degrees of freedom analysis to solve geometric constraint systems. *Proceedings of the first ACM symposium on Solid modeling foundations and CAD/CAM applications,* Issue 371--378.

Krishnaprakas, C. K., 1997. View factor between inclined rectangles. *Journal of thermophysics and heat transfer,* Volume 11, pp. 480-481.

Krysl, P., 2006. *A pragmatic introduction to the finite element method for thermal and stress analysis: with the Matlab Toolkit Sofea.* s.l.:World Scientific Publishing Co Inc.

Kvan, T., 2000. Collaborative design: what is it?. *Automation in construction,* Volume 9, pp. 409-415.

Lee, H. & Shepherdson, J., 2004. *Mediator-based recovery mechanism for multi-agent system.* s.l.:Google Patents.

Licht, J., Anderson, M. & Corradini, M., 2008. Heat transfer to water at supercritical pressures in a circular and square annular flow geometry. *International Journal of Heat and Fluid Flow,* Volume 29, pp. 156-166.

Lightsey, B., 2001. *Systems engineering fundamentals,* s.l.: s.n.

Li, W. D., Fuh, J. Y. H. & Wong, Y. S., 2004. An Internet-enabled integrated system for co-design and concurrent engineering. *Computers in Industry,* Volume 55, pp. 87-103.

Lowe, D., n.d. *From Paper to Interactive Electronic Technical Publications.* s.l.:s.n.

Maarten Bonnema, G. & Houten, V. a. F. J. A. M., 2006. Use of models in conceptual design. *Journal of Engineering Design,* Volume 17, pp. 549-562.

Maillet, D., 2000. *Thermal quadrupoles: solving the heat equation through integral transforms.* s.l.:John Wiley & Sons Inc.

Mao, J., Ching, H., Su, D. & Du, R., 2008. *A tolerance representation model for conceptual design.* s.l., s.n., pp. 413-416.

Maplesoft, 2017. *MapleSim - High Performance Physical Modeling and Simulation - Technical Computing Software.* [Online] Available at: http://www.maplesoft.com/products/maplesim/index1a.aspx [Accessed 09 03 2017].

Mathiak, F. U., 1985. Berechnung von Konfigurationsfaktoren polygonal berandeter ebener Gebiete. *Heat and Mass Transfer,* Volume 19, pp. 273-278.

McCord, K. R., 1993. *Managing the integration problem in concurrent engineering,* s.l.: s.n.

Mens, T. & Van Gorp, P., 2006. A taxonomy of model transformation. *Electronic Notes in Theoretical Computer Science,* Volume 152, pp. 125-142.

Mhenni, F. et al., 2014. A SysML-based methodology for mechatronic systems architectural design. *Advanced Engineering Informatics,* Volume 28, pp. 218-231.

Microsoft, 2015. *COM Clients and Servers.* s.l.:s.n.

ModelioSoft, 2017. *Modelio.* [Online] Available at: https://www.modelio.org/ [Accessed 07 03 2017].

MOF, O. M. G., 2007. Query/View/Transformation Specification. *OMG (ed.).*

Moullec, M.-L.et al., 2013. *Proposition of Combined Approach for Architecture Generation Integrating Component Placement Optimization.* s.l., s.n.

Muller, G., 2011. *Systems architecting: A business perspective.* s.l., s.n., pp. 1845-2142.

NASA-STD-7009, 2008. *Standard for models and simulations.* s.l.:NASA TECHNICAL.

Ng, L. X., Ong, S. K. & Nee, A. Y. C., 2014. Conceptual design using functional 3D models in augmented reality. *International Journal on Interactive Design and Manufacturing (IJIDeM),* Volume 9, pp. 115-133.

Ng, L. X., Ong, S. K. & Nee, A. Y. C., 2015. Conceptual design using functional 3D models in augmented reality. *International Journal on Interactive Design and Manufacturing (IJIDeM),* Volume 9, pp. 115-133.

No magic, 2017. *No Magic, Inc.* [Online] Available at: https://www.nomagic.com/ [Accessed 09 03 2017].

Object Management Group, 2006. *OMG Systems Modeling Language.* s.l.:Version.

Object Management Group, 2011. *OMG Unified Modeling Language(OMG UML).* s.l.:OMG Standard document.

Object Management Group, 2016. *Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification.* OMG standard ed. s.l.:formal/2016-06-03.

Object Management Group, 2016. *start [Model Interchange]*. [Online] Available at: http://www.omgwiki.org/model-interchange/doku.php [Accessed 09 03 2017].

on Systems Engineering, I. C. & Haskins, C., 2011. *Systems engineering handbook: A guide for system life cycle processes and activities.* s.l.:International Council of Systems Engineering.

Ooshima, M. & Masukata, S., 2013. *Design methodology for miniaturization of motor equipped in a fluid intermittent control system.* s.l., s.n., pp. 1194-1197.

OpenCascade, 2017. *OpenCascade.* [Online] Available at: https://www.opencascade.com/ [Accessed 09 03 2017].

OpenModelica, 2017. *Open Modelica.* s.l.:s.n.

OpenModelica, 2017. *Welcom to OpenModelica.* [Online] Available at: https://www.openmodelica.org/ [Accessed 09 03 2017].

Orton, J. D. & Weick, K. E., 1990. Loosely coupled systems: A reconceptualization. *Academy of management review,* Volume 15, pp. 203-223.

Pache, M. & Lindemann, U., 2003. Sketching in 3D What should Future Tools for Conceptual Design look like?. In: *Human Behaviour in Design.* s.l.:Springer, pp. 243-252.

Papa, S. et al., 2015. *Towards the integration of thermal physics and geometrical constraints for a 3D-multiphysical sketcher.* s.l., s.n., pp. 248-252.

Paredis, C. J. J. et al., 2010. *5.5. 1 An Overview of the SysML-Modelica Transformation Specification.* s.l., s.n., pp. 709-722.

Perry, N., Mauchand, M. & Bernard, A., 2005. Costs models in design and manufacturing of sand casting products. In: *Advances in Integrated Design and Manufacturing in Mechanical Engineering.* s.l.:Springer, pp. 69-80.

Pinelle, D. & Gutwin, C., 2005. *A groupware design framework for loosely coupled workgroups.* s.l., s.n., pp. 65-82.

Plateaux, R. et al., 2009. *Introduction of the 3d geometrical constraints in Modelica.* s.l., s.n., pp. 526-530.

Pop, A., Akhvlediani, D. & Fritzson, P., 2007. *Towards unified system modeling with the ModelicaML UML profile.* s.l., s.n.

PTC, 2017. *Integrity Modeler.* [Online] Available at: http://www.ptc.com/model-based-systems-engineering/integrity-modeler [Accessed 09 03 2017].

Qin, S. F. et al., 2003. A framework of web-based conceptual design. *Computers in Industry,* Volume 50, pp. 153-164.

Reichwein, A. et al., 2012. *Maintaining consistency between system architecture and dynamic system models with SysML4Modelica.* s.l., s.n., pp. 43-48.

Roa Castro, L., Stal-Le Cardinal, J. & Callo, M., 2015. Towards an adaptive model for collaborative simulation: from system design to lessons learned. A use case from Aircraft industry. *Procedia Computer Science,* pp. 267-273.

Römer, A. et al., 2001. Effort-saving product representations in design�results of a questionnaire survey. *Design Studies,* Volume 22, pp. 473-491.

Sadlauer, A. & Hehenberger, P., 2012. *Perspectives for the usage of design and modelling languages in mechatronic systems development.* s.l., s.n., pp. 279-332.

Schamai, W., 2009. *Modelica modeling language (ModelicaML): A UML profile for Modelica.* s.l.:Linköping University Electronic Press.

Schamai, W., Fritzson, P., Paredis, C. & Pop, A., 2009. *Towards unified system modeling and simulation with ModelicaML: modeling of executable behavior using graphical notations.* s.l., s.n., pp. 612-621.

Scicos, 2016. *Scicos Homepage.* [Online] Available at: http://www.scicos.org/ [Accessed 09 03 2017].

Selim, G. M. K., Wang, S., Cordy, J. R. & Dingel, J., 2015. Model transformations for migrating legacy deployment models in the automotive industry. *Software \& Systems Modeling,* Volume 14, pp. 365-381.

Shah, J. J. & Rogers, M. T., 1988. Functional requirements and conceptual design of the feature-based modelling system. *Computer-Aided Engineering Journal,* Volume 5, pp. 9-15.

Shen, W., Hao, Q. & Li, W., 2008. Computer supported collaborative design: Retrospective and perspective. *Computers in Industry,* Volume 59, pp. 855-862.

Siemens, 2017. *NX : Siemens PLM software.* [Online] Available at: https://www.plm.automation.siemens.com/en_us/products/nx/ [Accessed 09 03 2017].

Simpson, T. W., Bauer, M. D., Allen, J. K. & Mistree, F., 1995. *Implementation of DFA in conceptual and embodiment design using decision support problems.* s.l., s.n., pp. 119-126.

Sirin, G., 2015. *Supporting multidisciplinary vehicle modeling: towards an ontology-based knowledge sharing in collaborative model based systems engineering environment,* s.l.: s.n.

Sodhi, R. & Turner, J. U., 1994. Towards modelling of assemblies for product design. *Computer-Aided Design,* Volume 26, pp. 85-97.

Software, T., 2017. *TDC Software - softwares for product and prcess design quality, risk management and methodology.* [Online] Available at: https://tdc.fr/en/catalog_en.php [Accessed 09 03 2017].

Starkey, E., Toh, C. A. & Miller, S. R., 2016. Abandoning creativity: The evolution of creative ideas in engineering design course projects. *Design Studies,* Volume 47, pp. 47-72.

Steiner, I. D., 2007. Group Process and Productivity (Social Psychological Monograph).

Taentzer, G., 2003. *AGG: A graph transformation environment for modeling and validation of software.* s.l., s.n., pp. 446-453.

Taine, J. & Petit, J.-P., 2003. Transferts thermiques. *Sciences Sup. Paris, dunod edition.*

Temam, R., 1984. *Navier-stokes equations.* s.l.:North-Holland Amsterdam.

Thramboulidis, K., 2013. Overcoming mechatronic design challenges: the 3+ 1 SysML-view model. *Comput Sci Technol Int J,* Volume 1, pp. 6-14.

TM4, 2017. *medium and heavy duty vehicles TM4.* s.l.:s.n.

Tor, S. B., Britton, G. A., Zhang, W. Y. & Deng, Y.-M., 2002. Guiding functional design of mechanical products through rule-based causal behavioural reasoning. *International Journal of Production Research,* Volume 40, pp. 667-682.

Turki, S. & Soriano, T., 2005. *A SysML extension for Bond Graphs support.* s.l., s.n.

Turkyilmazoglu, M., 2013. The analytical solution of mixed convection heat transfer and fluid flow of a MHD viscoelastic fluid over a permeable stretching surface. *International Journal of Mechanical Sciences,* Volume 77, pp. 263-268.

Umeda, Y., Takeda, H., Tomiyama, T. & Yoshikawa, H., 1990. Function, behaviour, and structure. *Applications of artificial intelligence in engineering V,* Volume 1, pp. 177-193.

Varró, D., Varró, G. & Pataricza, A., 2002. Designing the automatic transformation of visual languages. *Science of Computer Programming,* Volume 44, pp. 205-227.

Walden, D. D. et al., 2015. *Systems engineering handbook: A guide for system life cycle processes and activities.* s.l.:s.n.

Walton, G. N., 2002. *Calculation of obstructed view factors by adaptive integration.* s.l.:Citeseer.

Wami, E. N. & Ibrahim, M. O., n.d. Model Equation for Heat Transfer Coefficient of Air in a Batch Dryer.

Wang, L. et al., 2002. Collaborative conceptual design�state of the art and future trends. *Computer-Aided Design,* Volume 34, pp. 981-996.

Warniez, A., Penas, O., Plateaux, R. & Soriano, T., 2014. *SysML geometrical profile for integration of mechatronic systems.* s.l., s.n., pp. 709-714.

Weiler, K. J., 1986. Topological structures for geometric modeling.

Wolff, R., Roberts, D. J., Steed, A. & Otto, O., 2007. A review of telecollaboration technologies with respect to closely coupled collaboration. *International Journal of Computer Applications in Technology,* Volume 29, pp. 11-26.

Ye, J., Campbell, R. I., Page, T. & Badni, K. S., 2006. An investigation into the implementation of virtual reality technologies in support of conceptual design. *Design Studies,* Volume 27, pp. 77-97.

Zang, T. A., 2008. *NASA Standard for Models and Simulations (M&S): Development Process and Rationale.* s.l.:NASA Engineering and Safety Center.

Zwolinski, P., Tichkiewitch, S. & Sghaier, A., 2007. The use of virtual reality techniques during the design process: from the functional definition of the product to the design of its structure. *CIRP Annals-Manufacturing Technology,* pp. 135-138.

ÉCOLE DOCTORALE
INTERFACES
Approches Interdisciplinaires : Fondements, Applications et Innovation

Titre : Gestion des interactions pour l'évaluation en phase de préconception, des architectures 3D de systèmes sous contraintes multi-physiques, application à la thermique

Mots clés : Conception préliminaire, Thermique, Simulation, Géométrie, Ingénierie Systèmes, Transformation de modèles

Résumé : La préconception est une phase aboutissant à la génération d'une architecture physique de concept. Dès cette phase, il est crucial de choisir une architecture qui prenne en compte les contraintes multi-physiques. Cette thèse permet de répondre à cette problématique : comment évaluer des architectures physiques d'un système complexe sous contraintes multi-physiques pendant les phases amont, afin de limiter les risques de couplages multi-physiques dans les étapes suivantes, qui engendrent une augmentation conséquente de la durée et du coût de la conception ?

Pour répondre à cette problématique, nous proposons tout d'abord un processus nommé SAMOS permettant aux acteurs de cette phase d'échanger des informations en limitant le risque d'incohérences et d'incompréhensions. Puis, en nous limitant à l'analyse thermique, nous avons développé une plateforme « modeleur 3D thermique », s'appuyant sur un environnement 3D, deux extensions SysML et des transformations de modèle, pour faciliter les échanges d'informations et de données entre l'architecte système, l'architecte 3D et les équipes de simulation, et ainsi diminuer la redondance et le temps de conception.

Ainsi, pour permettre la gestion de l'encombrement et du positionnement spatial de chaque équipement, une extension SysML GERTRUDe a été proposée pour pouvoir spécifier des exigences géométriques : formes géométriques simples des composants, dimensions, contraintes de positionnement. GERTRUDe utilise le modèle géométrique des SATT enrichi avec les paramètres intrinsèques. De la même façon, l'extension SysML TheReSE, basée sur GERTRUDe, permet la gestion des exigences thermiques : propriétés thermiques de composants et spécification des interactions thermiques susceptibles d'intervenir entre les composants de l'architecture.

De même les interactions thermiques entre les différents équipements peuvent être soit spécifiées en SysML, soit être ajoutées dans l'environnement 3D. Ces informations permettent de définir alors un réseau d'interactions thermiques, qui intègre à la fois les informations géométriques et thermiques. Ce réseau est alors transformé en un modèle thermique implémenté en Modelica, qui permet par simulation d'évaluer la température des faces des composants.

Les approches proposées ont été implémentées dans un démonstrateur, afin de démontrer la faisabilité du concept sur plusieurs cas d'études industriels, et ainsi valider les attentes industrielles vis-à-vis de l'approche proposée et ses perspectives.

Title : Interaction management in conceptual design for the assessment of 3D system architectures under multi-physical constraints: application to thermal analysis

Keywords : Conceptual design, Architecture evaluation, Thermal Simulation, Geometry, Systems Engineering, Model Transformation

Abstract: Conceptual design leads to the generation of a physical concept architecture. From this phase, it is crucial to select an architecture that takes into account multi-physical constraints. We propose in this thesis to solve the following research issue: how can the physical architectures of a complex system under multi-physical constraints be evaluated during the earlier design phases, in order to limit the risks of multi-physical coupling in the following phases that generate a considerable increase in design time and cost?

To tackle this problem, we first propose a framework called SAMOS which allows the actors in the design to exchange information during this phase while limiting the risks of inconsistencies and misunderstandings. Then, by focusing on the thermal analysis, we develop a "thermal 3D sketcher" platform, based on a 3D environment, two SysML extensions and several model transformations. It will facilitate human and data exchanges between System architects, 3D architects and simulation teams, thus reducing redundancy and design time.

Thus, in order to manage the geometry requirements and spatial positioning of each item of equipment, the GERTRUDe SysML extension is proposed. It allows specifying geometrical requirements such as simple geometrical shapes for the components, their dimensions and positioning constraints. GERTRUDe uses TTRS (Technologically and Topologically Related Surfaces) geometrical modeling enriched with intrinsic parameters. Likewise, the TheReSE SysML extension, based on GERTRUDe, allows the management of thermal requirements: the thermal properties of components and the specification of thermal interactions that may occur between the architecture components.

The transformation rules are described. They automatically generate a specified architecture which includes possible geometrical constraints that can be transformed from a SysML environment into a 3D environment; the direction of transformation can be reversed so that a 3D architecture can be traced from a 3D environment to a SysML environment.

Similarly, the thermal interactions between the different components can be either specified in SysML or be added in the 3D environment. This information allows defining a thermal interactions network which integrates both geometrical and thermal data. This network is then transformed into a thermal model implemented in Modelica, which allows simulation to evaluate the temperatures of the components' faces.

The approach proposed is implemented in a demonstrator to provide proof of concept based on several industrial case studies, thus validating the industrial expectations with regard to the approach proposed and its perspectives.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France