INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED'09
24 - 27 AUGUST 2009, STANFORD UNIVERSITY, STANFORD, CA, USA

FORMAL REQUIREMENT FORMULATION AND SYNTHESIS IN SYSTEM ENGINEERING

William Brace, Eric Coatanéa
Helsinki University of Technology, Finland

ABSTRACT

Requirement engineering is a specific branch of system engineering and includes activities such as eliciting, analyzing, verifying, and implementation. In software engineering, requirement engineering is one of the steps which might result at the end of the formal process to automate synthesis of code. Requirement engineering is also used to improve the rigor of the analysis performed, and to make the reasoning steps explicit. Such type of formal description has not been attained in system engineering. This is due to the fact that knowledge in system engineering is still more empirical and concepts are not defined with the precision and uniformity of software engineering. Requirement engineering has to spend the gap between the informal world of customer needs, and the formal world of system behavior. In addition, the role of computer in early design has to be intensified due to increase in productivity requirements and competitiveness. In this article, we provide an approach aimed at diminishing this gap by formalizing the needs for design progressively, and to intensify computer use by integrating our approach in SysML and formalizing the specification representation.

Keywords: system engineering, SysML, formal requirements, model-based design

1 INTRODUCTION

According to the definition provided by Rechtin [1], a system is; “a construct or collection of different elements that together produce results not obtainable by the elements alone. The elements, or parts, can include people, hardware, software, facilities, policies, and documents; that is, all things required to produce systems-level results. The results include system level qualities, properties, characteristics, functions, behavior, and performances. The value added by the system as a whole, beyond that contributed independently by the parts, is primarily created by the relationship among the parts; that is, how they are interconnected.”

From the interconnection and dependency of those parts forming the systems reside complexity. Complexity is a subject of interest for institutions and researchers since the 40´s. As mentioned by Micaëlli and colleagues [2], the design of radically new systems during this period explains mainly the interest for the study of complexity of artificial objects and, or services created by humans.

Understanding and managing the complexity is important because it can provide a powerful tool which exceeds our own individual and social limitations. Understanding complexity is another part of a bigger research project trying to provide a rational perception of our individual limitations (concept of bounded rationality) [3], and of our social interactions in an evolutionary perspective [4].

Complexity can take different forms in system engineering (SE). Most of the systems implement numerous functions, contains several components, and exhibits several behaviors. We find complexity in the functions, organs, structure but also in the design process. In order to cope with this complexity, designers traditionally used strategies which consist of recycling existing knowledge to solve new problems [3].

Those strategies have limitations for many modern problems and need to be accompanied with other types of approaches. This is the goal of SysML, derived from UML, and is intended to unify the diverse modeling languages and tools currently used by system engineers. SysML supports the specification, analysis, design, verification, and validation of a broad range of complex systems. These systems may include hardware, software, information, processes, personnel, and facilities.

SysML is relatively new as a modeling language; the origins of the language can be traced to a strategic decision by the International Council on System Engineering’s (INCOSE) in January 2001 to

customize the Unified Modeling Language (UML) for systems engineering applications. The language has been issued by Object Management Group (OMG) in March 2003 [5].
This article will focus more specifically on the creation of formal requirements procedure and synthesis that can be included in SysML. This approach is possible because of the large flexibility of SysML that allows customization. Requirements are defined in SysML in the form of diagrams. The requirements diagram can depict the requirements in graphical, tabular, or tree structure format. Several aspects of the requirements such as traceability, verification, and reuse have been taken into account. Nevertheless, we consider that the internal structure of the requirements diagrams needs and can be defined in a more formal manner. This is the goal of this article to rapidly present this approach.

2 SYSTEM ENGINEERING AND SYSML MODELING IN BRIEF

2.1 Model-base system engineering

System engineering (SE) is a process which considers the complete problem of a systems life cycle. It is a technologically based interdisciplinary approach which allows the realization of successful systems. It focuses on defining customer needs, required functionality, and documentation early in the development cycle. This is continued with design synthesis and system validation while considering cost, schedule, performance, test, manufacturing, support, operations, and disposal. In SE, a structured development process is formed that proceeds from concept to production to phase-off and disposal [6], [7].
Fundamental to the application of SE is the systems life cycle process. A number of lifecycle development models have been created and applied to systems development and most of these models are embedded in one of three prominent models. These are Waterfall Model, Spiral Model, and the V-Model and they provide the life cycle development templates [8]. The V-model combines the important features of the classic waterfall model and the spiral development model and is used as a model for requirement definition in this paper.
SE can be characterized by a model-driven methodology and is supported by a collection of related *processes*, *methods*, and *tools*. Definitions from Martin [9] are used to clear erroneous consideration of methodology and process as synonymous. A *process* is a logical sequence of tasks performed to achieve a particular objective, and a *method* is the techniques for performing a task. A *tool* is an instrument when applied to a particular method can enhance its efficiency and most tools used to support SE are computer- or software based.
Based on these distinctions, a *methodology* can be defined as a collection and application of related *processes*, *methods*, and *tools* to a class of problem [10]. A process model that defines the primary activities that must be performed to implement SE and is related to the phases in an acquisition and utilization of life cycle model is the system engineering process (SEP).
Numerous methodologies have been created to assist the design process. These include Function-Behavior-Structure [11], or Function-Behavior-States [12]. Theories have been developed to support these methods. Value Analysis conceived by Lawrence Miles in the 1945 is one of the most used approaches [13]. A primary benefit of modeling is the opportunity provided for its analysis. There is a paradigm shift from traditional document-based life cycle approach most of which follows waterfall model of system design to model-based engineering (MBE). MBE is about elevating models in an engineering process to a central role in the *specification*, design, integration, validation, and operation of a system.

2.2 Review of model-base methodologies

A review of some of the more notable model-base system engineering (MBSE) methodologies reveals Object-Oriented Systems Engineering Method (OOSEM) with Systems Modeling Language (SysML) as an integrated top-down model base approach to help architect flexible and extensive systems that can accommodate changing requirements [14]. SysML is used to support specification, analysis, design, and verification of systems. It is a general purpose graphical modeling tool and a key enabler for transitioning the practice of SE from being document-centric to a model-centric approach.
The SysML diagram types are identified in Figure 1 and blocks are the basic unit of structure and can be used to characterize hardware, software, facilities, stakeholders, and or any other system element [15].

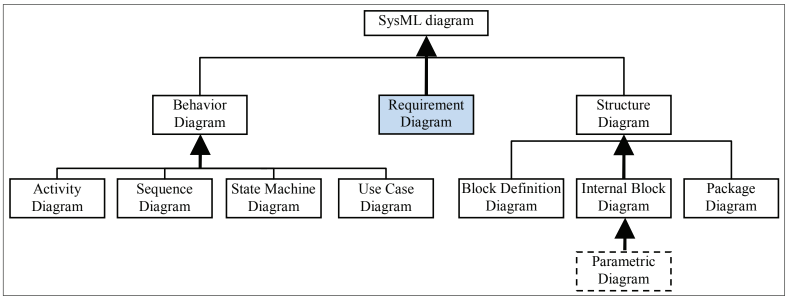


Figure 1. SysML diagram taxonomy [[5]

There are four main diagrams dubbed "the four pillars" and includes the requirement, structure, behavior, and parametric diagrams (Figure 2). For instance, the parametric diagram represents constraints on system property values such as performance, reliability, and mass properties, and facilitates the integration of specification and design models with engineering analysis models. The requirement diagram provides a bridge between the typical requirements management tools and the system models. The requirements diagram captures requirements hierarchies and requirements derivation, and the verify relationships permit a modeler to relate a requirement to a model element that satisfies or verifies the requirements. The internal structure of the requirement diagram is the focus of this research work. Depending on the design concept, diagrams in SysML can be used for different purposes.

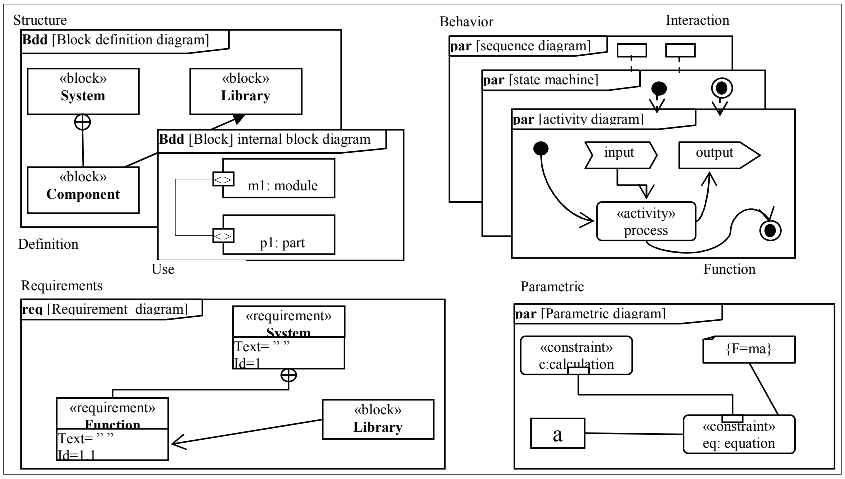
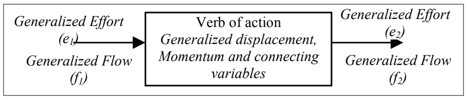


Figure 2. SysML pillars [5]

3 EVALUATION AND COMPARISON MODELS

The basic assumption in design processes is the existence of three classes of variables, namely function variables, behavior variables, and structure variables. In early phase of design the customer needs or requirements for a system are abstracted to formulate a structure variable which satisfies these requirements. The Function-Behavior-Structure (FBS) model by Gero [11] represents design by a set of processes linking function, behavior and structure together. It shows design as a set of eight processes which includes formulation, synthesis, analysis, evaluation, documentation and reformulation. Gero's proposal indicates the significance that function, as well as behavior is a mixed concept which can be intentional as well as structural.

Now introducing the definition of a function [16], a function is the boundary between inner and outer situations. A situation on the other hand represents a state of the elements in the environment within a volume of time and space. Inspired from the bond graph theory [17], a function combining both intention (for instance, verb of action) and the structure (five types of generalized variables) is as shown in Figure 3a.


Generalized Effort (e_1)
Generalized Flow (f_1)
Verb of action
Generalized displacement, Momentum and connecting variables
Generalized Effort (e_2)
Generalized Flow (f_2)

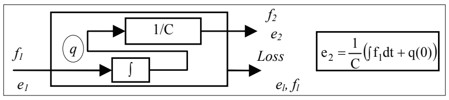


Figure 3 a) Generic model of a function, b) SADT representation of storage mechanism C-organ

Five elementary mechanisms have been defined by Karnopp and others [18] and were extended by Coatanéa [16] to include all the function vocabulary of Hirtz and colleagues [19]. A structured analysis and design technique (SADT) diagram is used to representation the mapping of the family of mechanisms with functions as shown in Figure 3b. The bond graph theory and the generic model specialization are not discussed in this article.

3.1 Dimensional analysis in brief

The Π-theorem has provided several well known Π-numbers especially in fluid mechanics and thermodynamic [20]. The Π-numbers or Vashy-Buckingham theorem is the core theorem of dimensional analysis and it demonstrates how the physical description of a phenomenon can be reduced to its minimum set of dimensionless variables. A dimensional number takes the following form (Equation 1).

$$\Pi_i = y_i (x_1^{\alpha_{j1}} x_2^{\alpha_{j2}} x_3^{\alpha_{j3}}) \tag{1}$$

Where (x_1, x_2, x_3) is the repeating variable set and (y_i) is the performance variable set [21]. Butterfield [20] proposed a table (Table 1) which lists and sorts variables governing a system.

Table 1. Selection of repeating and performance variables (adapted from Butterfield [20])

		V								
		R					P		O	
		Q			S					
		V1		Vm		Vo		Vp		Vn
D	d1	A(mxm)			B(mx(n-m))					
	dm									

In the table, V is the independent set of variables assumed to govern the system. R is variables selected from V with distinct dimensions other than 0. P contains variables not in R and constitutes the performance variables set. O is set of variables with zero dimensions. D is a possible set of m independent quantities from basic quantities. The selection of the set Q which is the repeating variables list is not unique as rules may be applied for its selection in a design context. A variable of interest whose behavior is to be investigated is included in P.
There is occurrence of interactions between variables in Π-numbers which is internal in this research. The interactions numbers are called partials and computed via the exponents sign (α_{ij}) used in Equation 1 for intra-Π-numbers partials (Equation 2) and inter-Π-numbers partials (Equation 3) using contact variables (x_p). Interested readers should refer to Coataneá [16], Butterfield [20], and Bhashkar [21] for broader explanation about the approach.

$$\frac{\partial y_i}{\partial x_j} = \alpha_{ij} \frac{y_i}{x_j} \tag{2}$$

$$\left[\frac{\partial y_i}{\partial y_j}\right]^{x_p} = \frac{\partial y_i / \partial x_p}{\partial y_j / \partial x_p} = \frac{-(\alpha_{ip} y_i)/x_p}{-(\alpha_{jp} y_j)/x_p} = \frac{\alpha_{ip}}{\alpha_{jp}} . \frac{y_i}{y_j} \tag{3}$$

4 CRITICAL ANALYSIS OF THE STATE OF ART IN SYSTEM ENGINEERING

Representation of design problem into requirements has been pointed out by several influential textbooks and some researchers as an important first step in design activities [22].

Bahill and Dean [23] suggest that requirements are the basic features defined for a system before and during design. Grady [24] lays more emphasis on the customer's need as the ultimate system requirements from which all other requirements flow.

Requirements are defined in IEEE P1220 standards as [6], [25]; "the statements which identify the essential needs for a system in order for it to have value and utility". From this definition, it is clear that requirements may be derived or based upon analysis of other stated requirements to assist in providing a common understanding of the desire characteristics of a system. We refer to stated requirements as *natural (or initial) requirement* in this study. The *methodology* to derive a formal requirement from the *natural requirements* is the focus of this study (Figure 4). This is to allow for a clear understanding of the uniqueness of a system for further manipulation by different stakeholders in an SE design environment.

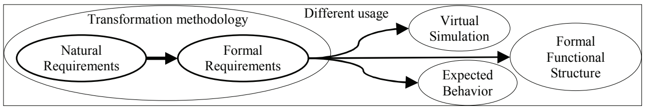


Figure 4. Requirement formalization for system analysis and modeling

SE possesses some defined characteristics which includes a *functional purpose* in response to an identified *need* and the ability to achieve some stated *operational objectives*. Thus, as stated by Blanchard and Fabrycky [7], SE concentrates on *what the entities do* before determining *what the entities are*. Simply stated, the identification of a *need* is followed by *function* and then *structure*. Fundamental to the application of SE is the systems life cycle process where design is different from design in the ordinary sense and involves different stakeholders. Life cycle focused design is responsive in real-time to customer needs or requirements expressed in functional terms. The functional definition of a complex system serves as the baseline for the identification of the overall requirements and thus, a reliable overall structure.

Figure 5 presents a modified FBS model [11]. In this model, several elements have been added in order to represent the method in which *natural requirements* (R) should be gradually transformed in the vision developed in this article into a *detail structure* (D). The present article focuses on the step 0, transformation of the natural requirements into *formal requirements* (R_F), step 0' transformation of the *formal requirements* (R_F) into *formal functions* (F_F), step 2' transformation of the *formal functions* (R_F) into *Generic structure* (S_G), step 3' partially by defining certain aspect of the *Generic expected behavior* (BGe).

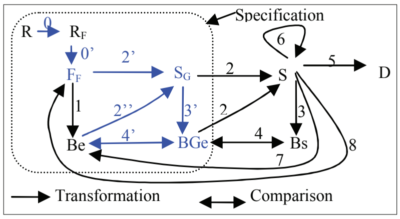


Figure 5. Modified FBS model and formal requirements/specifications

Readers can consider that our definition of requirements and specifications is too vast. Nevertheless, it seems necessary for us to define an intermediate step between functional representation and physical structure. The addition of this step represents several advantages. Indeed, by analogy with the TRIZ approach [26], it helps to solve the engineering design problem by first representing the problem in its generic form and solution and then derived from this generic solution a specific solution. The generic form is participating to the formalization of the system engineering problem. In addition, it provides a reference path ensuring later the comparability of the different behaviors. At the moment it is very difficult to verify the comparability of the *expected behavior* (Be) represented using for example Petri net and the *behavior derived from the structure* (Bs) represented using for example Modelica model. With the proposed type of architecture it becomes possible to imagine behavior models derived stepwise from Be.

5 PROPOSED APPROACH FOR FORMALIZING THE REQUIREMENT

5.1 Requirement definition in brief

Very often engineers talk about "gathering requirements". This phrase is quite misleading and it suggests that requirements are just on hand somewhere waiting to be extracted from the users' brain by an analyst. In reality requirements definition (RD) is a more appropriate description of acquiring the necessary system requirements. RD is subdivided into the critical process areas of *elicitation, analysis, specification*, and *validation*. These processes are critical in SE. One outstanding eliciting technique is modeling. *Modeling* is the construction of abstract descriptions that are open to interpretation. This is a fundamental activity in RD [27] which happens to be an activity also in SE.
There is some disparity between requirements and specifications which should be clarified [28]. Specification is a document that specifies, in a complete, precise, verifiable manner, the requirements of the system and often, the procedures for determining whether these provisions are satisfied.
Sources of requirements or requirements taxonomy are in abundance. Bahill and Dean [28] named twelve sources. Many authors disputes the number of requirements taxonomy. According to Wymore [29] there are only six of those sources which are necessary: input-output, technology, performance, cost, trade-off, and system test. All the rest of the sources can be included into one of these six. The Capability Maturity Model Integration (CMMI) [6] describes three levels of requirements namely customer, product, and product-company. Project managers say there are only three; cost, schedule, and performance.
The Unified Modeling Language (UML) states three requirements taxonomy; functional, nonfunctional, performance, and supplemental requirements that are spread throughout the system. Grady [24] dispute the numbers by also stipulating there are only five sources; functional, performance, constraints, verification, and pragmatics (as analysis in this article). He argues that most of the other sources are constraints. The five requirements taxonomy according to Grady [24] together with the first taxonomy of Wymore [29] (input-output) is used as the basis and integrated into the critical process areas of RD to create a formal requirements procedure.

5.2 Approach for formalizing requirement

Having introduce this summary of RD methods, we advocate that it follows logically the capability to build a common set of attributes (for instance, set of metrics) which can be used to formulate the requirement model. Figure 6 represents a vision of the specification combining both the input in the form of requirement taxonomy and the output (verification, analysis) based on Grady's [24] and Wymore's [29] requirements taxonomy. The inspiration of this vision comes from function modeling [22].

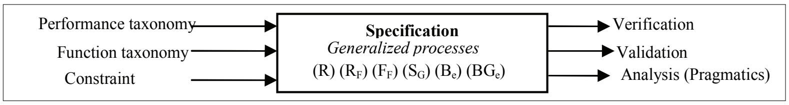


Figure 6. Requirement definition model

This model can be specialized. The specialization is based on the V-model which provides the life cycle development template on which the model is built [8]. The V-model template and the RD model, is use to define a structured requirement (Figure 7). This approach is justified to simplify the understanding of the complexity associated with design specification. RD is done through an iterative process involving several methodologies. Thus using the V-model, we have shown a specialized RD model by defining a uniform procedure for a formal requirement model.

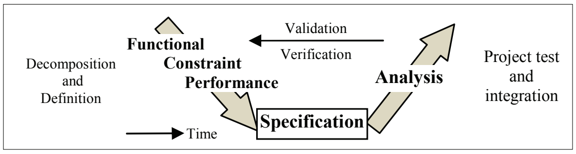


Figure 7. Formal requirement V-model template

In order to present the formalized approach, the V-model is populated presenting ontology of the formalization process (Figure 8). The sign (>) gives the orientation of the transformation and deduction processes. The populated verification and analysis fraction of the v-model is not presented in this article.

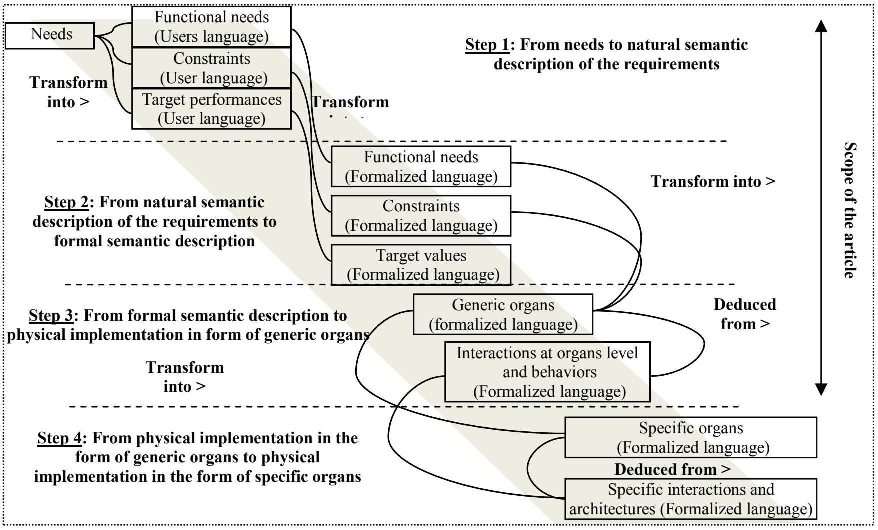

Needs
Functional needs (Users language)
Constraints (User language)
Target performances (User language)
Transform into >
Transform
Step 1: From needs to natural semantic description of the requirements
Functional needs (Formalized language)
Constraints (Formalized language)
Target values (Formalized language)
Transform into >
Step 2: From natural semantic description of the requirements to formal semantic description
Generic organs (formalized language)
Interactions at organs level and behaviors (Formalized language)
Deduced from >
Step 3: From formal semantic description to physical implementation in form of generic organs
Transform into >
Specific organs (Formalized language)
Deduced from >
Specific interactions and architectures (Formalized language)
Step 4: From physical implementation in the form of generic organs to physical implementation in the form of specific organs
Scope of the article

Figure 8. Ontology of the formalized requirements in a V-cycle

The modified FBS-model (Figure 5) allows us to position clearly different SysML model diagrams (Figure 1) and the use of dimensional analysis in the evaluation process as shown in Figure 9.

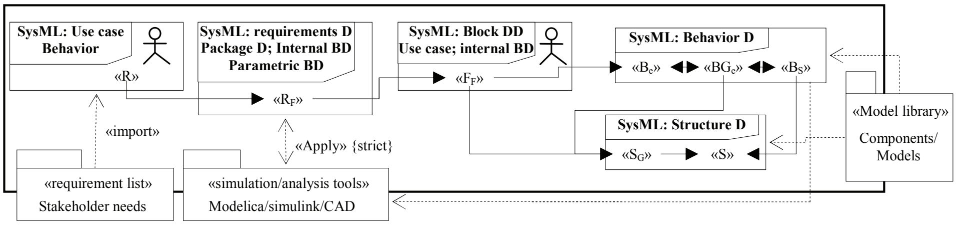

SysML: Use case Behavior
«R»
SysML: requirements D Package D; Internal BD Parametric BD
«R_F»
SysML: Block DD Use case; internal BD
«F_F»
SysML: Behavior D
«B_e» «BG_e» «B_S»
SysML: Structure D
«S_G» «S»
«import»
«Apply» {strict}
«requirement list» Stakeholder needs
«simulation/analysis tools» Modelica/simulink/CAD
«Model library» Components/ Models

Figure 9. The modified FBS-model and corresponding SysML diagrams for requirement formalization and synthesis

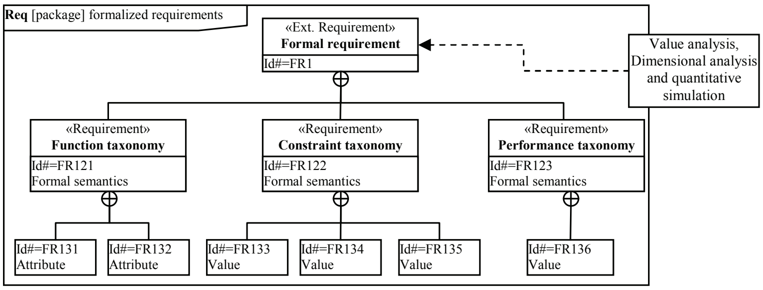

Req [package] formalized requirements
«Ext. Requirement» Formal requirement
Id#=FR1
Value analysis, Dimensional analysis and quantitative simulation
«Requirement» Function taxonomy
Id#=FR121 Formal semantics
«Requirement» Constraint taxonomy
Id#=FR122 Formal semantics
«Requirement» Performance taxonomy
Id#=FR123 Formal semantics
Id#=FR131 Attribute
Id#=FR132 Attribute
Id#=FR133 Value
Id#=FR134 Value
Id#=FR135 Value
Id#=FR136 Value

Figure 10. Integration of formal requirement and SysML for model evaluation

SysML provides modeling constructs to represent text based requirements, organize them in a suitable manner and relate them to other modeling elements. But it also allows customizing the requirement taxonomy to match specific modeling interest. The formal requirement model (Figure 7, Figure 8) is integrated in the requirement diagram and the SysML meta-model is modified to introduce logical properties that formalize the textual description of the requirement. The requirement diagram allows us to compute dimensionless groups, representing the synthesis of the procedure (Figure 10). The internal structure of the SysML formalized requirement model is not discussed into details in this article as it is an area identified for future work.

6 A CASE STUDY

The design problem studied in this short example applies a formal requirement procedure to the design of a structure able to support measurement instruments. The main need for the future user is to avoid or limit displacements due to the weight of the instruments. The procedure follows the expanded procedure described in Figure 8.

- Step 1- From needs to natural semantic description of the requirements

Classical tools of Value Analysis such as the interaction diagram are used to describe this step (Figure 11, Table 2).

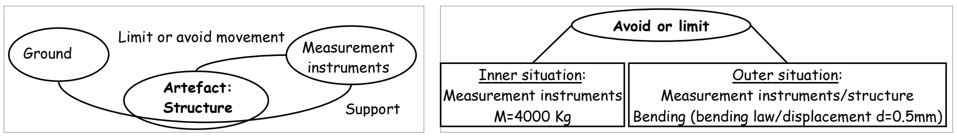


Figure 11. Interaction graph of the problem and representation of the constrained function inhibit or limit

Table 2. First level of requirement description

Functional need (user language)	Support
Constraints (user language)	Limit or inhibit
Target performances (user language)	M= 4000 kg max, d=0.5mm

- Step 2- From natural semantic description of the requirements to formal semantic description

For this purpose we use a semantic atlas combined with the taxonomy of Hirtz [19]. This taxonomy is our reference to create the formalize language. The mapping between the natural language and the taxonomy is made using queries sent to the semantic atlas available on the net and comparison of the results with the Hirtz's taxonomy (Table 3).

Table 3. Transformation of the natural language into formal language

Natural language	**Formalized language**
avoid	Primary function: Magnitude, Secondary function: Stop, Tertiary function: Prevent
limit	Primary function: Control, Secondary function: Regulate
support	Primary function: Support, Secondary function: Stabilize, Secure, Position

- Step 3- From formal semantic description to physical implementation with generic organs

For this task we use a semantic atlas combined with taxonomy of organs developed by Coatanéa [16] as briefly explained in section 3. The mapping between the formal functional language and the taxonomy is made once again using queries sent to the semantic atlas and by comparison of the results with the taxonomy of basic organs (Table 4, Table 5).

Table 4. Taxonomy of the basic organs

Families of mechanisms (verbs used to retrieve the families in the semantic atlas)	Basic organs contained in those families	Generic variables (e: effort, f: flow, q: displacement, p: momentum) and some of the laws
Storage (STORE)	C-Organs, I-Organs	$e = \frac{1}{C}(\int f.dt + q(0))$
Dissipative (DISSIPATE)	R-Organs	$e = R.f$
Source (PROVISION, SUPPLY)	Effort sources, Flow sources	$e_O = e_I$
Converting (CONVERT)	T-organs, G-organs	
Distribute (BRANCH, DISTRIBUTE)		
Painter's Junctions (CONNECT, COUPLE, JOIN)	Effort junctions, Flow junctions	
Mechanical junctions (CONNECT, COUPLE, JOIN)	Lock link, rotational link, etc..	
Calculation (SIGNAL, PROCESS)	NO, AND, OR organs	

Table 5. Mapping results of the formalized functional language with the organs

Formalized language	Families of organs	generic variables and laws involved
Primary function: Magnitude, Secondary function: Stop, Tertiary function: Prevent	Mechanical and/or painter's junctions (Connect, Couple, join)	effort XYZ, flow XYZ
Primary function: Control, Secondary function: Regulate	None	/
Primary function: Support, Secondary function: Stabilize, Secure, Position	Storage organs of type C (because input in form of displacement)	$e = \frac{1}{C}(\int f.dt + q(0))$

- The generic organ of type C is analyzed in a more detailed manner in order to establish (if possible) the form of the factor 1/C

This step is still in our viewpoint equivalent to the transformation 2' describe in Figure 5 and not to transformation 2.the step allows for mapping of the generic organs and laws with specific organs and laws. This stage requires making choices in the taxonomies of material and energy developed by Hirtz [19]. For simplicity purposes in this article, we consider a mechanical field and a solid substance. In this example, the storage organ of type C is considering with the following black box (Figure 12).

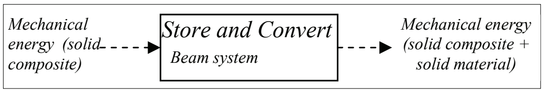

Mechanical energy (solid composite)
Store and Convert
Beam system
Mechanical energy (solid composite + solid material)

Figure 12. Functional structure of a C-organ

A C-organ in our case can be implemented practically in the form of a beam. It can be considered by readers that the fact of mentioning the term "beam" means that we are synthesizing (i.e. transformation 2) a *structure* S (Figure 5). In our perspective this is not the case because we are not yet considering any geometry of the beam but only the variables. The equations governing the physical phenomenon in the beam are of the following.

According to the generic law of a C-organ: $e = \frac{1}{C}(\int f.dt + q(0))$ (4)

The law ruling the beam bending is: $F = \frac{C_1 EI}{L^3} d$ (5)

The factor in Equation 5 is equivalent to the factor $1/C$ in the general law of a C-organ (Figure 3b, Equation 4). Consequently, the variables which can be used in order to describe beams are the displacement *d* due to the bending effort which is a state variable, the Young modulus (*E*), the Inertia (*I*) and the length of the beam (*L*) which are connecting variables. The SADT diagram can be completed by using these 4 variables completed by the input power variables F_1 (Figure 13). The speed v_1 and the output power variables are out of the interest in this example because they are not modeled in the initial representation of the concept of function (Figure 11). We have then 3 dimensionless groups:

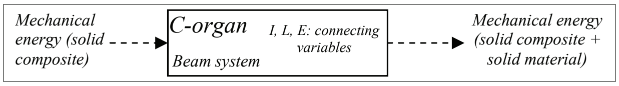

Mechanical energy (solid composite)
C-organ
I, L, E: connecting variables
Beam system
Mechanical energy (solid composite + solid material)

Figure 13. A beam and the main variables

- Selection of the repeating and performance variables:

Table 6. Selection of the performance and repeating variables (adapted from Butterfield [20])

		V									
		R				P				O	
		Q		S							
		E	***I***			***F_1/d***	***A***	***L***			
D	L	-1	4			0	2	1			
	T	-2	0			- 2	0	0			
	M	1	0			1	0	0			

The selection of performance variables is bases on dimensional analysis as explained in section 3.1 using Table 1. Table 6 shows the selection of the variables.

$$\Pi_1 = \frac{F_1}{d} E^{-1} I^{-1/4}, \quad \Pi_2 = AI^{1/2} \; and \; \Pi_3 = LI^{1/4} \tag{6}$$

Π_1 is combining the effort applied on the beam, the material of the beam and the shape of the beam. Π_2 is related to the shape of the beam. It should be noticed that this dimensionless group exhibit similarities with the shape factors introduced by Shanley [30] and used by Ashby [31]. Π_3 is related to the length combined with the shape of the beam. The partials of the intra-dimensionless groups are provided by the machinery of Bashkar and Nigam [21] from Equations 2 and 3:

$$\frac{\partial(F_1/d)}{\partial E} = \frac{F_1/d}{E} \rangle 0 \; and \; \frac{\partial(F_1/d)}{\partial I} = \frac{1}{4}\frac{F_1/d}{I} \rangle 0 \; for \; \Pi_1 = \frac{F_1}{d} E^{-1} I^{-1/4} \tag{7}$$

$$\frac{\partial A}{\partial I} = -\frac{1}{2}\frac{A}{I} \langle 0 \; for \; \Pi_2 = AI^{1/2} \tag{8}$$

$$\frac{\partial L}{\partial I} = -\frac{1}{4}\frac{L}{I} \langle 0 \; for \; \Pi_3 = LI^{1/4} \tag{9}$$

The partials of the inter-dimensionless group are all defined according to the contact variable *I*:

$$\left[\frac{\partial(F_1/d)}{\partial A}\right]^I = -\frac{1}{2}\frac{(F_1/d)}{A} \langle 0 \; for\ the\ relation\ between\ \Pi_1 \; and \; \Pi_2 \tag{10}$$

$$\left[\frac{\partial(F_1/d)}{\partial L}\right]^I = -\frac{(F_1/d)}{L} \langle 0 \; for\ the\ relation\ between\ \Pi_1 \; and \; \Pi_3 \tag{11}$$

$$\left[\frac{\partial A}{\partial L}\right]^I = 2\frac{A}{L} \rangle 0 \; for\ the\ relation\ between\ \Pi_2 \; and \; \Pi_3 \tag{12}$$

From these partials, we extract the following information. Several interactions are highlighted F_1/d and *E* (interaction 1), F_1/d and *I* (interaction 2), *A,* and *I* (interaction 3), F_1/d and *A* (interaction 4), *L* (interaction 5), finally *A* and *L* (interaction 6).
Table 7 and Table 8 formalized the functions, units, variables, targets, ideal values. Table 8 formalizes the interaction and the cascading direction of the flow of interactions. The light green line can be read in the following manner, F_1/d is influenced by the section A, the length L, the inertia I and the young modulus E. The light blue column means that the section A is influencing the Inertia I, A, and L are influencing each other backwards and/or forward. We have the same forward and backward influence for the Π numbers.

Table 7 List of attributes and quantities involved in the problem

Functions	Units	Attributes of the specialized C-organ mechanical field and solid substance				Target values	Ideal values
		Performance variables					
		Power variable type	State variables type	Connecting/comparison variable types	Connecting/comparison variable types		
STORE and CONVERT	MLT^{-2}	Effort $\mathbf{F_1}$				[30000, 50000]	∞
	L^2			**A** section of the beam		[0.2,0.6]	0
	L		**d** bending displacement			[0.03,0.08]	0
	L^4				**I** inertia of the beam	[0.4,0.9]	∞
	L			**L** length of the beam		[0.4]	4
	$ML^{-1}T^{-2}$				**E** Young modulus	[180000,250000]	∞

The formal requirements description is combining Figure 12, Figure 13, Π-numbers of Equation 6, partial of Equations 7 to 12, Table 7 and the modified FBS-model presented in Figure 5.
All these elements need to be later integrated in SysML as shown in Figures 9 and 10. This is already feasible at the current development stage of SysML. It can be noticed that the conjoint use of Petri net with Pi numbers modeling can be imagined to describe continuous-time *behaviors of the Generic structure* (BGe) (see Figure 5). Nevertheless, this has not been considered in the present article.

Table 8 Design Structure Matrix (DSM) table representing formal interactions between generic design variables

		Performance variables			Repeating variables		Pi-numbers		
		F_1/d	A	L	I	E	Π_1	Π_2	Π_3
Performance variables	F_1/d	■	X(Eq.10)	X(Eq.11)	X(Eq.7)	X(Eq.7)			
	A		■	X(Eq.12)					
	L		X(Eq.12)	■					
Performance variables	I		X(Eq.8)		■				
	E					■			
Pi-numbers	Π_1						■	X(Eq.10)	X(Eq.11)
	Π_2						X(Eq.10)	■	X(Eq.12)
	Π_3						X(Eq.11)	X(Eq.12)	■

7 CONCLUSION

In this paper, we introduced a systematic approach for requirement definition which can be applied in system engineering. The purpose of the study was to enhance and improve our understanding of the nature of design specification description and see how this formalized description can be included in SysML. This approach is used to synthesis the requirement solutions, thus enabling a formal requirement notations and languages. As requirements written in natural languages are fraught with vagueness, this formal requirement procedure will allow for enhanced readability, analysis, validation, and modeling. It can even be considered that in the near future a semi automatic specification procedure can be implemented.

The approach described in this paper goes further into the synthesis phase of the requirements solutions. In this respect, the methodology proposed is not only a requirements definition approach but a synthesis approach (at least the first level of the synthesis) which will lead to future work.

There was a case study to present the manner in which a formal requirement procedure can be realized.

The authors believe that this paper contains an interesting supplement to systems design, and we have established a first justification of our formal requirements procedure. Future work will complete this analysis by focusing on behavioral aspects and derivation of different types of behavioral models from an initial form of combined model approach.

REFERENCES

[1] Rechtin, E. *Systems Architecting of Organizations*, 2000 (CRC Press).

[2] Micaëlli J.P., Bonjour, E. and Deniaud, S. Architecture as a key activity of complex systems design. 2008, *Proceedings CRECOS 08*, Helsinki, September 11,12.

[3] Simon H-A. *The Sciences of the Artificial. 3rd edition,* 1997, (MIT Press, Cambridge).

[4] Gould Stephen Jay. *The Structure of Evolutionary Theory*. 2002 (Cambridge:Belknap Press of Harvard University Press). ISBN 0-674-00613-5

[5] OMG SysML, *OMG Systems Modeling Language.* 2007, (V1.0 OMG available specification)

[6] INCOSE SE, *Terms Glossary Version 0.* Copyright (c) 1998 (by INCOSE).

[7] Blanchard B. and Fabrycky W. *Systems engineering and analysis 4th edition,* 2006 (Prentice hall international series in industrial and systems engineering)

[8] Forsberg K. and Mooz H. Application of the "Vee" to Incremental and Evolutionary Development. *Proceedings of the Fifth Annual International Symposium of the National Council on Systems Engineering (NCOSE)*, July 1995, St. Louis, MO.

[9] Martin J. N. *Systems Engineering Guidebook: A Process for Developing Systems and Products,* 1996, (CRC Press, Inc.: Boca Raton, FL).

[10] Bloomberg J. and Schmelzer R. *Service Orient or Be Doomed!*: *How Service Orientation Will Change Your Business,* 2006 (John Wiley & Sons: Hoboken, New Jersey)

[11] Gero J. S. and Kannengiesser Udo. The situated function, behavior, structure, framework. *Design Studies* 2004, 25 pp. 373, 391

[12] Umeda Y., Ishii M., Yoshioka M. and Tomiyama T. Supporting Conceptual design based on the function-behavior-state modeler. *Artificial Intelligence for Engineering Design,*

Analysis and Manufacturing, 1996, 10(4), pp. 275-288.
[13] Miles, L. D. *Techniques of Value Analysis and Engineering, 3rd Edition*, 2006, (SAVE International VM Standard).
[14] Friedenthal S. Object Oriented Systems Engineering. Process Integration for 2000 and Beyond: *Systems Engineering and Software Symposium*, 1998, New Orleans, LA, (Lockheed Martin Corporation).
[15] Weilkiens T. *Systems Engineering with SysML/UML: Modeling, Analysis, Design,* 2008, (Publisher; Morgan Kaufmann)
[16] Coatanéa E. Conceptual Modeling of Life Cycle Design: a Modeling and Evaluation Method Based on Analogies and Dimensionless Numbers. *PhD dissertation*, 2005, ISBN 951-22-7853-7, ISBN 951-22-7852-9
[17] Shim T. Introduction to Physical System Modeling Using Bond Graphs, *University of Michigan-Dearborn,* 2002.
[18] Karnopp, D.C, Margolis D.L. and Rosenberg R.C. *System Dynamics: A unified Approach,* 1990, John Wiley & Sons, New York, Second revised edition.
[19] Hirtz J., Stone R. B., McAdams D. A., Szykman S. and Wood K. L. A functional basis for engineering design: Reconciling and evolving previous efforts, 2002, *Research in Engineering Design* 2002, 13, pp. 65–82. (Springer-Verlag)
[20] Butterfield R. Dimensional analysis revisited. Proceedings of the *I MECH E Part C Journal of Mechanical Engineering Science*, Volume 215, Number 11, 23 November 2001, pp. 1365-1375(11), Professional Engineering Publishing.
[21] Bhashkar R. and Nigam A. Qualitative physics using dimensional analysis*, Artificial intelligence*, 1990, vol. 45, pp. 73-111.
[22] Pahl G. and Beitz W. *Engineering Design. A Systematic Approach 3rd Edition.* 2007, Ken Wallace, Luciënne Blessing, translators and editors. (Springer, Berlin-Heidelberg).
[23] Bahill A.T. and Dean F.F. What is Systems Engineering? A Consensus of Senior Systems Engineers. *Proceedings of the Sixth Annual Symposium of the International Council on Systems Engineering (INCOSE)*, July 7-11, 1996, Boston, Vol. 1, pp. 503-508.
[24] Grady J.O. *System Requirements Analysis*. 1993 (New York: McGraw Hill Inc).
[25] IEEE P1220 *Standard for Systems Engineering.* 1994, (IEEE Standards, Dept., NY).
[26] Altshuller G. *Creativity as an exact science*, 1984 (Gordon & Breach, Luxembourg)
[27] Bashar N. and Easterbrook S. Requirements engineering: A roadmap. *Future of software engineering* Limerick Ireland. Copyright ACM 2000 1-58113-253
[28] Bahill, A.T. and Dean, F.F., Discovering system requirements, in A.P. Sage and W.B. Rouse (Eds), *Handbook of Systems Engineering,* John Wiley & Sons, 1999, pp. 175-220
[29] Wymore W. *Model-Based Systems Engineering.* 1993, (Boca Raton: CRC Press).
[30] Shanley F.R. *Weight-Strength Analysis of Aircraft Structures*, *2nd edition*, 1960, (Dover Publications, New York).
[31] Ashby M.F. *Materials Selection in Mechanical Design, 2nd Edition*, 1999 (Butterworth Heinemann)

Contact: William Brace
Engineering Design and Production
Helsinki University of Technology (TKK)
P.O. Box 4100
FIN-02015 HUT, Finland
Tel. +358 9 451 3451
William.brace@tkk.fi

William Brace has M.Sc. (Eng.) in product development from Odessa Lomonosov University, Ukraine and M.Sc. (Tech.) in Mechatronics from Helsinki University of Technology (TKK). He is currently a research scientist at the Department of Engineering Design and Production. His research interests focus on early design analysis and design for sustainability for complex systems.

Eric Coatanéa is Professor in Product Development at the Helsinki University of Technology. His research and teaching interests focus on the overall problem of designing sustainable and integrated systems. This includes topics such as: early design analyses, developing formal methodologies for complex system design, design theory and methods for designing sustainable systems.