A conceptual, methodological and technical contribution for modeling and V&V in MBSE context

Vincent Chapurlat, Blazho Nastov, Jeremy Bourdon

A conceptual, methodological and technical contribution for Modeling and V&V in MBSE context

Vincent Chapurlat
Laboratory for the Science of Risks, IMT Mines Ales, Ales, France
vincent.chapurlat@mines-ales.fr

Blazho Nastov
Ansys
4 Avenue des Saules, 59160 Lille
blazho.nastov@ansys.com

Jérémy Bourdon
Laboratory for the Science of Risks, IMT Mines Ales, Ales, France
jeremy.bourdon@mines-ales.fr

Abstract – The role of modeling, verification, and validation, even not new in Systems Engineering, is increasingly highlighted as crucial expectation in Model Based System Engineering context (MBSE). However, some difficulties, needs, and locks must be considered to allow MBSE to grow in capabilities and maturity. This article clarifies some fundamental expectations for MBSE and proposes research findings to address some of the MBSE practitioners' needs more precisely.

Keywords – Model, Modeling, Domain Specific Modeling Languages, Model-Based Systems Engineering, models versus system Verification / Validation

I. INTRODUCTION

The role of modeling, verification, and validation, even though not new in Systems Engineering, is more and more requested and promoted for various advantages regarding a more classical document-oriented engineering. This induces particular interests and relevance for the Model-Based System Engineering approach (MBSE). Several academic works and industrial developments have taken up these orientations. However, some fundamentals still need to be discussed, may be improved, or even laid down, both in terms of conceptual, methodological, and technical aspects. This article attempts to address some of these fundamentals for MBSE and proposes research findings to address the needs of MBSE practitioners.

II. PROBLEMATIC

A. MBSE position

Model-Based Engineering is defined in [2][3] as "*an approach to engineering that uses models as an integral part of the technical baseline that includes the requirements, analysis, design, implementation, and verification of a capability, system, and/or product throughout the acquisition life cycle*" hereafter called System of Interest (SoI) [1]. In coherence with SE expectations and principles, Model-Based System Engineering (MBSE) is defined as "*the formalized application of modelling to support system requirements, design, analysis, verification and validation beginning in the conceptual design phase and continuing throughout development and later life cycles phases*" [4]. To complete this definition, [5] considers more globally MBSE as "*a collection of related processes, methods, and tools*". Last, MBSE aims to support operational actors involved in SE projects and, more generally, all Stakeholders themselves concerned, involved or impacted by such projects. The goal is to engineer complex systems by creating, checking, and handling models. So MBSE is to be considered a mean for SE practitioners. As mentioned by [6], "*MBSE doesn't replace traditional SE Rather, MBSE formalizes part of SE"*.

B. MBSE practitioners' global needs

As evidence, MBSE must provide or be based on shared and recognized theoretical, methodological, and technical bases to support these actors in its implementation covering all the SE stakeholders' needs. MBSE must therefore allow SE Stakeholders to:

- model any viewpoint of any (part of) complex system and more generally of any kind of System of Interest (SoI) in conformance with systemic principles. However, what is considered as a model and modeling activity in the MBSE context?
- check models of (part of) SoI, i.e., to verify and validate (V&V) models and pay attention to qualities and default of such models, e.g., precision, use of reductionist hypotheses, etc.) prior to verify and validate the modeled SoI itself. However, how can these V&V activities, both focusing on models or on system, be more appropriately defined and implemented in the MBSE context?
- manipulate these models to justify decision-making processes, requesting, for instance, simulations [7], formal or semi-formal analysis techniques that can be directly or indirectly (via model transformations) applied to different models of the same SoI. However, these models are often considered in isolation to test separately various expectations or hypothesis. Today, they must be regarded globally as interconnected and interdependent modeling elements whose whole forms a more complete, if possible faithful and realistic, description of the same SoI;
- use these models as much as possible in confidence, i.e., to analyze, evaluate, compare alternative solutions, optimize, trace, and at the end, to generate documents;
- remain coherent with SE principles and processes that are subject to standardization [1] or adaptation according to the type of company [8];
- dispose of data, information, and knowledge repositories accessible all along the life cycle of the SoI. The knowledge includes models that must then be accepted or at least consensual, shareable without a considerable effort and loss of meaning, and considered mature or authoritative in specific fields of systems engineering and business engineering.

Last, MBSE must also:

- dispose of relevant "*modeling languages that support rigorous modeling techniques and integration of various systems engineering disciplines (structural, electrical, mechanical, software, etc.) and stakeholders*" [6];
- consider usages and practices of actors involved in engineering projects, for instance, in terms of modeling

means, objectives, or even tools, by preferring the use of existing tools and avoiding as much as possible specific developments of new tools.

C. *Proposed contributions*

This article intends to:

- Formalize more precisely some basics of modeling concepts for the MBSE;
- Formalize more precisely some basics, even propose alternative point of view about V&V meeting the needs of the MBSE Stakeholders;
- Propose then an operational and equipped approach to promote modeling and V&V in line with these expectations based on so-called *xviDSML* (executable verifiable and interoperable DMSL [12] (see section III.C);
- Propose how to develop a support tool for the use of *xviDSML* in modeling and V&V.

III. CONTRIBUTION: CONCEPTUAL ASPECTS

A. *Modeling, view and viewpoint, model*

Before proposing how to characterize a model, some hypotheses are fixed. First, modeling describes a system, a phenomenon, or any element, by respecting the conventions specified generally by a modeling language and for a given purpose and objectives. In other words, it leads to create, according to given rules and formalisms, a model of the system, phenomenon, or element on which it must be possible to make reasoning and judgements coherently to serve these objectives and finality.

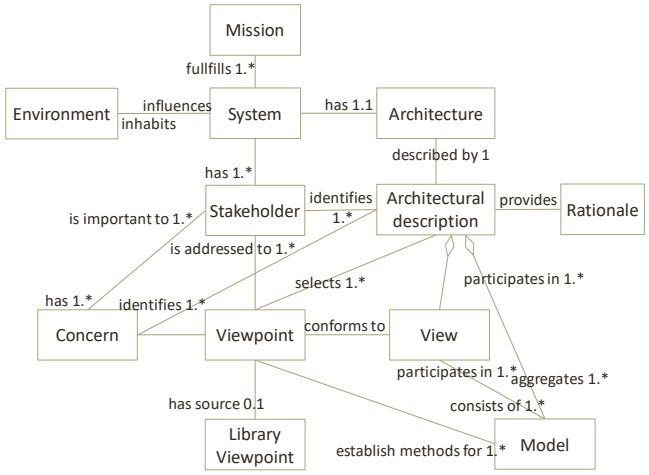

Fig. 1: IEEE Std 1471-2000 (interpretation overview)

Second, [9] (see Fig. 1) defines:

- A view as "*a representation of a whole system from the perspective of a related set of concerns*".
- A viewpoint as "*a specification of the conventions for constructing and using a view. A pattern or template from which to develop individual views by establishing the purposes and audience for a view and the techniques for its creation and analysis*".

Fig. 2 proposes some views considered in the next as relevant for the MBSE context and some appropriate Domain Specific Modeling Languages (DSML) [10] or *model kinds* that are to be used by MBSE practitioners to express one or several models in each view.

As a first conclusion, the position of the views and viewpoints is detailed and specified by considering the Modeling Pyramid initially proposed by OMG in [11] (see Fig. 3).

Related view…	*Examples of model kind*
System: what seems to be or must be S? What are the relevant and available data, information and knowledge about S? interactions? …	Life-cycle diagram (steps and milestones), Context diagram (services and fields), Stakeholders list, Data / Information / Knowledge…
Requirements, Values and Properties: what should S answer to?	Requirements Baseline, Requirements models, Values and preference diagram, Properties influence graph,…
Functional/Logical: what is the mission and what should S do, what services should it render? **what are the components of S, and how are they structured independently of any physical solution?**	Functional Architecture(s), Logical Architecture(s), Decision models… Logical Architecture(s), Decision models…
Physical/Organic: in the reality of **implementation,** what does S consist of to fulfil this mission?	Physical Architecture(s), Decision models…
Behavioural: how does S evolve **and behave** in order to fulfil its mission?	(*discrete, continuous, or hybrid modelling*) Operational Scenarios, Operational Modes, Equations, **…**
Risk management: events, situations and dangers to be detected, anticipated or avoided in order to gain S stability and integrity all along its lifecycle	Risks (effects, impacts), Influence relations network diagram, …

Fig. 2: main relevant views and model kinds

Then, as commonly defined, a model M of an SoI S refers to the data obtained after modeling S and is "*a perception of (maybe imaginary) reality*". Indeed, by hypothesis:

- M must reproduce how S behaves, evolves, or interacts with its environment when placed in the same conditions as S; M provides the same outputs as S when subjected to the same inputs. Indeed, a model M_i expresses a stakeholder's advice in a given view, allowing him or her to select and focus on a particular set of concerns (see Fig. 3). The whole approach is used to model by conforming to a viewpoint$_i$ or, eventually, various viewpoints that are equivalent or complementary for the view purpose.

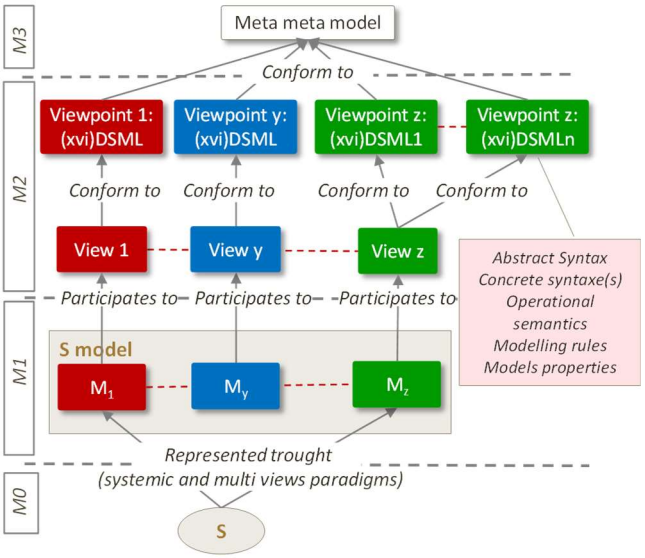

Fig. 3: revisiting OMG metamodeling pyramid

- A model has to be "u*seful when it answers a question!*" [13] but M remains only an image that is not necessarily complete or faithful, filtering out unnecessary details of reality of S or of its environment with respect to the modeling objectives and viewpoint of the stakeholder who built it. So, M is a snapshot therefore likely to be limited or even invalid for other objectives.

Finally, [14] defines a System model as "*an interconnected set of model elements which represent key*

system aspects including its structure, behavior, parametric, and requirements" as synthesized in Fig. 4. This proposition converges with the concept of Digital Mock-Up (DMU) proposed in [15]. As a second conclusion, a first version of model formalization for the MBSE domain is proposed in [15] and is then not recalled in this article.

More generally, as this is often the case with other engineering disciplines, MBSE may use two types of models (see Fig. 5 being inspired from [16]).

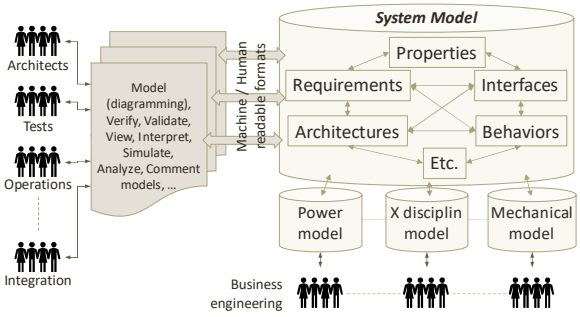

Fig. 4: System model as a global, common, and shared modeling artifact inspired from [6]

- **Black Box model:** Model M results from dedicated modeling activities focusing on data sets extraction, classification, training, and analysis processes (all data are then considered as related to the SoI S or equivalent systems) by using various techniques and approaches for data extraction, indexation, identification, treatment, etc. including for instance more or less classical statistical approaches, fuzzy approaches or neural networks approaches from AI domain. Considering some of these approaches, stakeholders could face various difficulties in justifying and explaining its contents, inducing a lack of confidence in terms of fidelity, credibility, and plausibility of M;

- **White Box model:** Model M results from an explicit modeling activity and uses a formalized DSML (analytical model). M is to conform to this DSML then it may be more or less formally interpretable and justifiable by respecting DSML operational semantics and properties, and by proving and evaluating model M properties. This article focuses on this type of model.

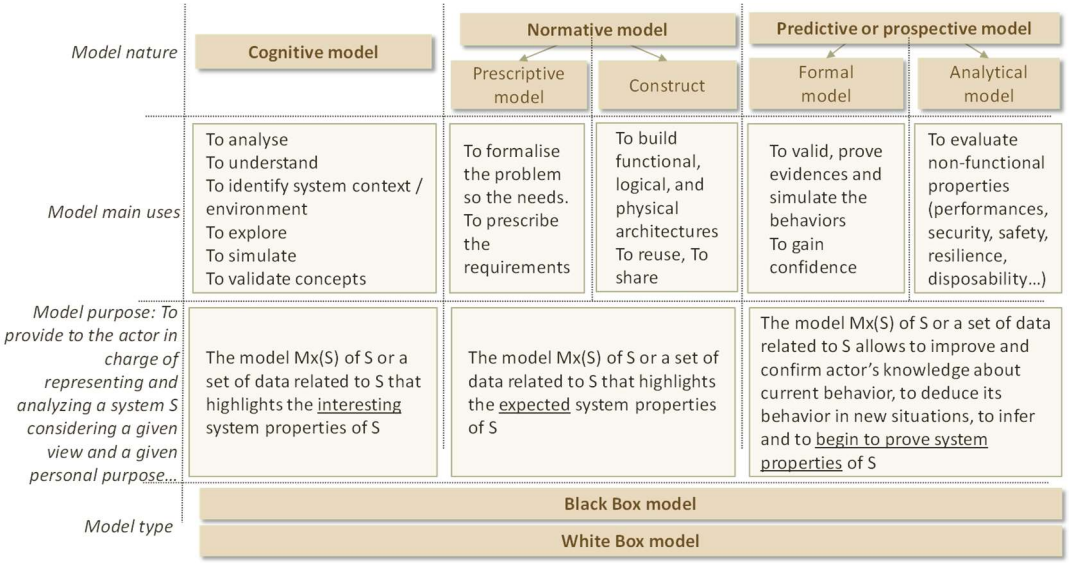

Fig. 5: Roles and use of models, inspired by [16]

Last, MBSE requests models from various natures (Fig. 5) allowing us to precise roles and objectives of a model whatever may be its type:

Cognitive model solicited as:

- a means of understanding the problem (iteratively, step by step), the perceived phenomenon, the demand, the environment, and the solution: *to make stakeholders express themselves and to capture opinions, dissatisfactions, expectations, and visions.*

- a communication vehicle: formalizing and sharing knowledge (at any stage in the life of the system), *getting teams to collaborate*

Normative model (prescriptive or construct) solicited:

- for designing ("engineering") a solution i.e. finding it: respect the requirements i.e. the modeling properties, the properties translating the 'native' requirements, and the requirements induced by possible previous choices

- to optimize a solution: *is it multi-disciplinary (wishful thinking)? This would imply modeling the field of possibilities and being able to "walk-around" in it, aligning and reconciling models of costs, business, efficiency, risk, decision, etc.*

Predictive or prospective model solicited to check and evaluate, then help decide and argue (choice, rejection, quantified or qualified), to describe and evaluate a (beginning of) solution track (which may allow initiating several different and more complete solutions) by simulation/evaluation/... *verify certain properties of the solution and validate it against the problem and requirements*

B. V&V

Verification is *"a set of activities that compares a system or system element against the required characteristics. This includes, but is not limited to, specified requirements, design description, and the system itself. The system was built right"* [1] (section 6.4.6). The verification of a model of the SoI, however, remains evasive.

Validation is *"the set of activities ensuring and gaining confidence that a system can accomplish its intended use, goals, and objectives (i.e., meet stakeholder requirements) in the intended operational environment. The right system was*

built" [1] (section 6.4.8). Particularly, model validation is defined by [17] as "*the process of ensuring., e.g. the model correctly represents the domain or system-of-interest"*.

By operating iteratively during the technical processes of systems engineering e.g. as proposed in [8], V&V is a necessary step to guide, assure or reassure oneself, to progress in the design step by step and to improve the level of maturity of the solution, by ensuring that one's own needs are met. It is, therefore, a necessary step to guide, ensure, or, failing that, reassure oneself, to progress in the design step by step and by improving the maturity level of the solution, by ensuring that one progresses in conformity with business and domain customs and practices without cutting oneself off from possible innovations, by helping to detect errors, omissions or ambiguities, to anticipate and to test situations by bringing into play only models, to test the non-regression of the solution, etc.

In fact, by focusing on the design phase of an SoI that puts forward diverse and varied models, the questions that V&V must answer, and thus the early V&V that is increasingly referred to today, are a priori:

- How to improve confidence in a model of an SoI? We consider here a model in isolation from the others: the goal is to ensure the level of 'quality' of what it represents independently of the other models of the same SoI;

- How to improve the confidence in the set of models that describe the same SoI? We consider here all the models of this SoI, which should then be put into interaction (composed and/or federated as allowed by the FMI/FMU standard [18], for example) to move towards the system model. The latter offers a holistic and more complete vision and is considered sufficient if not totally faithful to the solution (notion of duplication). It is then necessary to consider the differences between these models during this federation/composition (e.g. semantics, level of maturity, modeling language (DSML) used, level of detail addressed in the SoI, objectives of the modeler at the origin of this model, ...). The goal is to ensure a coherent and, if possible, complete multi-point of view representation with respect to the objectives of the actors to produce proofs, simulations, non-functional property evaluations, analyses (e.g. sensitivity or dependency analysis, impact analysis, effects propagation analysis, etc.)

- How can we then rely on these models to ensure that the solution found (or the various alternative solutions) for the SoI is indeed the expected system? The two hypotheses to consider to answer this question are 1) we rely on previously verified and validated models and 2) we perceive this solution only through these models. There is therefore a possible bias between reality (the expected system) and perception (model) since models are only filters of reality and are themselves based on assumptions (reductive and/or simplifying) that had to be adopted to consider the objectives, experience, and process employed by the modeler. It is at this level that we speak of early V&V, which requires the use of models with a maximum level of confidence.

So, a stakeholder involved in V&V activities seeks overall to improve and justify his or her level of confidence, that is, he or she seeks to establish and be able to argue that there is a balance between:

- Credibility: Credibility: 1) the credibility of the model concerning the SoI and the domain, as well as 2) the organization put in place to produce this model: skills and recognition of the modeler's experience, relevance, use, and availability of resources (tools, methods, testbeds, etc.) and overall maturity of the V&V process (see for example [19]);

- Plausibility of the model i.e. likelihood or acceptability of the model by other stakeholders considered as experts from a domain and regarding the SoI requirements;

- Fidelity of the model i.e. the model ideally converges to a necessary and sufficient duplicate of the SoI, as it is understood by the stakeholder and considered a requested level of details, the modeling assumptions imposed by the view, the modeling language, or even the modeling tool used, and the limiting or simplifying assumptions adopted by the stakeholder related to its objectives;

- Relevance of the model to the modeler's objectives and answer various kinds of questions!

Finally, it should be noted that trust is an expectation that is constantly evolving and may even collapse as a justifiable belief. So, to assume, or at least improve stakeholders' confidence level, it is then proposed to define four kinds of V&V activities. Each intends to provide essentially theoretical justifications, sometimes to provide results that can be considered as more or less empirical justifications considering the confidence level of the model/system model:

- **Model verification** (or model quality checking): "*did I do the model right?*" (or "*did I follow the rules and practices to model the system of interest?*").

- (A previously verified) **Model validation**: "*did I make the right model?*" (or, at a minimum, "*did I model the system of interest I have in mind?*").

- (A previously verified and validated Model is used for) **System Model early verification:** "*have I modeled the system of interest correctly?*" or, is the system of interest, as modeled (i.e. represented by a set of heterogeneous, federated or composed models), coherent, unambiguous, compliant with business and domain rules, and compliant with some of the requirements specified for the system of interest, provided that these requirements are analyzable through these models?

- **(**A previously verified and validated Model that has been already used successfully for early verification) is used for **System Model early validation** [20]): "*have I modeled the right system of interest so that I can argue with confidence that this system of interest is a good solution?"*.

Table 1, Table 2, Table 3, and Table 4 (next page) show respectively model Verification, Validation, system (early) verification, and system (early) validation expected outcomes in terms of theoretical or empirical justification and considering two model cases, focusing on static and dynamic aspects:

(**Case 1**) a model M (considered alone and independent from other models of the same SoI SOI), or,

(**Case 2**) a model M that results from the federation or composition of a set of models i.e. M is a system model.

In both cases, expected returns are errors, mistakes, oversights, misunderstanding, requests for model modification, modeling rules explanation or model clarification or precision, requests for requirements explanation, modification or clarification, justifications, performance evaluation, non-functional properties evaluation (values, limit, dependence, ...), ...

For these aims, V&V strategies can be classified into: Model Appraisal, Guided Modeling, Simulation, and Formal Proof [21]. Particularly, we focus hereafter on the Simulation and Formal proof strategies for which a tool-equipped approach for the design and use is proposed below.

C. DSML vs. xviDSML

A Domain Specific Modeling Language (DSML) formalizes a set of modeling conventions, and rules to create

a model from both syntactic, semantic and pragmatic rules and conventions i.e. a methodological way to build and use it.

Model verification	
Focus on the static aspect	*Focus on the dynamic aspect*
(Case 1) aims to: demonstrate the absence of modeling errors, mistakes, and oversights, the respect to particular modeling rules (i.e., conformity to a metamodel, constraints, and invariants), and rules corresponding to the domain and business field modeling expertise (e.g. good practices and modeling patterns)	**(Case 1) aims to:** demonstrate that applying the DSML operational semantic allows executing the model (considered alone and independent from other models of the same SoI) without errors, mistakes, or ambiguities when applying these behavioral semantics then 'executing' the model
(Case 2) aims to: demonstrate the whole coherence, absence of federation, or composition errors (e.g. models' connections, models i/o definition, etc.). If federation or composition is checked, demonstrate the absence of mistakes and oversights, the respect to particular modeling rules (i.e., conformity to a metamodel, constraints, and invariants), and rules corresponding to domain expertise (i.e., good practices and patterns)	**(Case 2) aims to:** demonstrate the whole dynamic coherence and executability of the system model M

Table 1: Model Verification aims

(verified model is then used for) ***Model validation***	
Focus on the static aspect	*Focus on the dynamic aspect*
(Cases 1 and 2) aims to: demonstrate M is the right one that is to say: 1) M is relevant considering the modeling objectives and hypotheses, and 2) M is trustworthy i.e. it provides a sufficient and accurate representation of the SoI as it is modeled in a view (e.g. functional, structural, requirements, or behavioral)	**(Cases 1 and 2) aims to:** demonstrate M execution strengthens the demonstration obtained from a static aspect, by executing the model facing various scenarios considered necessary and sufficient by all stakeholders.

Table 2: Model validation aims

(verified and validated model is then used for) ***System Model early verification***	
Theoretical justification	*Empirical justification*
(Case 1) aims to: demonstrate and provide justifications that M (previously verified and validated) provides an accurate representation of the SoI 1) when modeled in a given view (e.g. functional, structural, requirements, or behavioral), and 2) M respects various, at least one, stakeholders' requirements or system requirements thanks to the modeling objectives for which M has been built.	not yet applicable nor relevant for M use in case 2
(Case 2) aims to: demonstrate and provide justifications that federating or composing each model that composes M allows modeling entirely the SoI to become able to prove the SoI is correctly built thanks to all stakeholders' and system requirements.	**(Case 2) aims to:** use M as a numeric representation (e.g. numeric prototype i.e. an equipped DMU) to assume the SoI is correctly built thanks to stakeholders' and system requirements, business field and domain expectations, usages, and best practices

Table 3: System (early) verification aims

(verified and validated model previously used for system early verification is then used for) ***System Model early validation***	
Theoretical justification	*Empirical justification*
(Case 2 mainly) aims to: demonstrate that, M being previously used to verify the system, allows to test SoI to assume its relevance and it reaches its objectives thanks to stakeholders' and system requirements and considering various scenarios as necessary and sufficient by all stakeholders.	**(Case 2 mainly) aims to:** use M as a basis of a specific Digital Twin to be developed, or 2) prefer to complete the demonstration by developing and using prototypes or demonstrators to assume the SoI is relevant and reach its objectives thanks to stakeholders' and system requirements, and considering various scenarios as necessary and sufficient by all stakeholders.

Table 4: system (early) validation aims

A DSML is then considered equivalent to a model kind that is defined by [22] as a "*kind conventions for a type of modeling (examples of model kinds include data flow diagrams, class diagrams, Petri nets, balance sheets, organization charts, and state transition models)".*

As proposed in other relevant works e.g. [23][24], this article promotes to building and use executable, verifiable, and interoperable DSML (*xviDSML*) that, besides classical system (parts or elements) modeling in conformance with views definition, allows direct verification without model transformation, support validation then evaluation of the models i.e. allows to prove various kinds of properties and simulating or even emulating the behavior of the model. As it could be done for other DSML, building *xviDSML* is to be done by defining Abstract syntax, Concrete (and alternative but equivalent) syntaxes, Operational semantics, and Modeling Properties as detailed below. To illustrate the concepts, an application example is given in [25] and illustrated hereafter by defining an *xviDSML* named Operational Mode Analysis Guide (OMAG) [26]. OMAG aims to help engineers and architects describe, share, discuss and formalize:

- what are the expected operational modes of an SoI from its realization to its end of life;
- how they must be chained thanks to various events and considering system requirements;
- what are then the expected operational scenarios and the requested SoI configurations that must be achievable in each mode or during the transition from one mode to the next.

The goal is then to facilitate the obtaining of a basic functional architecture of this SoI satisfying a priori these operational scenarios and the whole set of system requirements. Applied to the OMAG case, the requested elements are:

- **Abstract syntax**: It gathers and formalizes concepts, relations between concepts, attributes, and constraints

imposed to be used to model (partially or entirely) one (eventually various) views(s). It is commonly defined as metamodel respecting meta-modeling principles [27] as depicted in Fig. 6 where metamodel conforms to Ecore meta metamodel language [28];

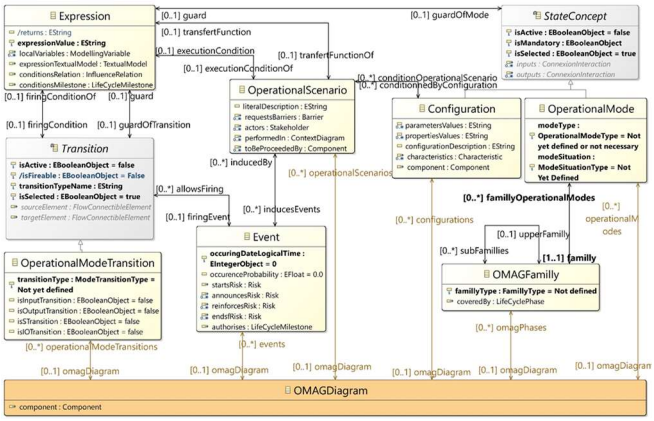

Fig. 6: abstract syntax (partial view of OMAG meta model)

- **Concrete syntax**: It defines at least one, eventually various equivalent graphical or textual rendering allowing stakeholders to create and handle the elements highlighted in the abstract syntax. It could be not unique considering the usages and habits of stakeholders, the whole concrete syntaxes remaining fully semantically interoperable allowing to facilitate model sharing without ambiguities. Any of these are defined thanks to the development environment, hereafter using OBEO Designer Team environment [29] as proposed in Fig. 7 in OMAG case that presents the initial concrete syntax and the one that has been then developed in this environment;

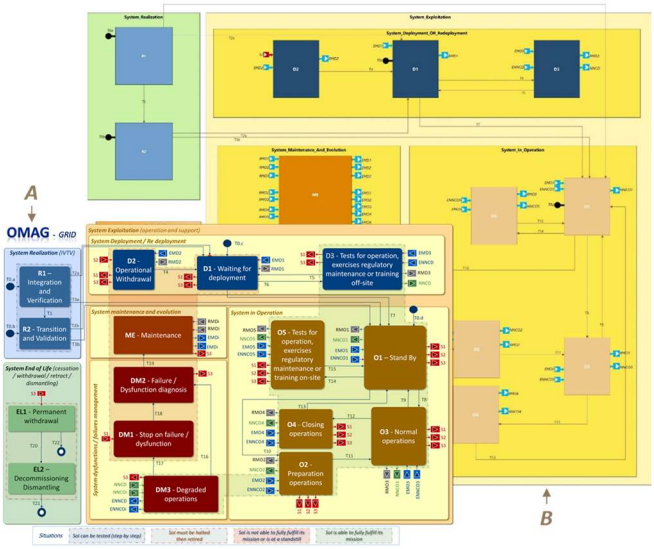

Fig. 7: OMAG concrete syntax (as expected and as currently implemented)

- **Operational Semantic**: It is composed of a set of formal interpretation/execution rules that define how must evolve the model or each of the concepts that compose the model (autonomous mode if considered alone or in controlled mode when considered as a model of System Model, then being synchronized and dependent from the evolution of the whole set of models that compose the system model). It could be formalized for instance by using a state machine or software code. It remains unique to avoid any ambiguities during model execution i.e. model simulation or model emulation. Fig. 8 shows the unformal version of such a set of interpretation rules that have been translated under ACCELEO code [30] enabling then direct simulation on the OBEO Designer environment. The stakeholder can then check directly, without any transformation, any OMAG model.

Initialise model (set xviDSML OMAG monitor)
- Return global result of modelling properties checking (all static MP must be checked)
- Return global result of modelling properties checking (all dynamic MP must be checked)
- if OK then
  - Activate Initial Operational Mode
  - Set initial SOI Configuration
  - Set initial Operational Scenario
  - Set execution environment
    - Set and initialise content of Black-board for Logical Time Unit T=0
    - Set and initialise shared variables / events / clocks
    - Set Logical Time Unit T=0
    - Set Logical Time Unit Tmax
    - [optional: launch and initialise execution of other models i.e. set xviDSML monitors e.g initial Operational Scenario xviFunctionalDiagram monitor])
    - Initialise and update simulation interface rendering
- endif

Read OMAG inputs
- Set Logical Time Unit T=T+1
- Read inputs from Black-board (messages, events,shared values from other xviDSML monitors)
- Read inputs from user

Compute next OMAG State
- Compute set {Tr} of fireable transitions
- if card{Tr}=0 (no possible OMAG evolution) or card{Tr}>1 (non dynamic determinism) then STOP
- Fire selected transition Tr (unique)
  - Deactivate source operational mode MS
    - Check current SOI configuration used in MS
    - [optional: halt current Operational Scenario xviDSML monitor to be analysed when MS is
    - Check functional state of the SOI
  - Activate destination operational mode MD
    - Check differences between current SOI configuration used in MS and default SOI configuration promoted in MD
    - [optional: launch and initialise the Operational Scenario xviDSML monitor simulating the SOI reconfiguration scenario associated to Tr]
    - [optional: launch and initialise default Operational Scenario xviDSML monitor
    - Check new functional state of the SOI
- [Optional: check and trace System properties evolution]

Write OMAG outputs
- Update local variables
- Write outputs to Black-board (messages for other xviDSML monitors, events and shared values)
- Update simulation interface rendering

Iterate to ER 2 until STOP or execution is halted by user or T>=Tmax

Fig. 8: unformal OMAG operational semantic

- **Model properties**: A property is "*a provable or assessable characteristic of an artifact [which is 1) a system S, or 2) a model M of S] that reflects all or part of the stakeholder's expectations that must be met by that artifact*" [21]. They express then the expected general qualities (both static, or dynamic i.e. dependent on temporal hypotheses as defined in [31]) of the model. Some are mandatory and all allow verifying and partially validating the model in model verification and validation.

Examples of static model properties: informal and formal definitions of such Model Properties are hereafter done by using ACCELEO and can be proved by using proof mechanisms allowed in the OBEO Designer environment.

MP1::= Is the operational mode definition complete?

$MP1::= P1 \wedge P2 \wedge P3$ where:

- P1::=The mode has a name i.e. P1 ::= [thisEObject.name<>''/]
- P2::=The mode has a guard expressed in Logical Unit Time, eventually set to the 'O.O' value i.e.

P2 ::= [thisEObject.guard->asSet()->size()<>0 and ([if] (thisEObject.guard->asSet()->size()<>0) [then] thisEObject.guard- >=0.0[endif]/](/]

- P3::= If and only if (P3.1 $\wedge$ P3.2)=true, the default configuration C conditions at least the default operational scenario OS i.e.

P3::= P3.1 $\wedge$ P3.2 $\wedge$ [thisEObject.authorisesScenariosAndConfigurations.defaultOperationalScenario.conditionnedByConfiguration=(thisEO

bject.authorisesScenariosAndConfigurations.defaultConfigur ation)/] where:

- P3.1::=A default operational scenario OS is associated to the operational mode describing the default expected behavior of the SoI when considered in this mode i.e.

P3.1 ::= [thisEObject.authorisesScenariosAndConfigurations.defa ultOperationalScenario->size()=1 /]

- P3.2::= A default configuration of the SoI C is associated to the operational mode describing the default expected configuration of the SoI when considered in this mode, i.e.

P3.2 ::= [thisEObject.authorisesScenariosAndConfigurations.defa ultOperationalConfiguration->size()=1/]

Example of dynamic model properties: informal and formal definitions of such Model Properties are hereafter simplified to facilitate comprehension. It can be verified by using a model transformation developed with ACCELEO then allowing stakeholder to use formal properties proof (applied then essentially to dynamic model properties) allowed by UPPAAL tool [32] as it has been proposed in [33].

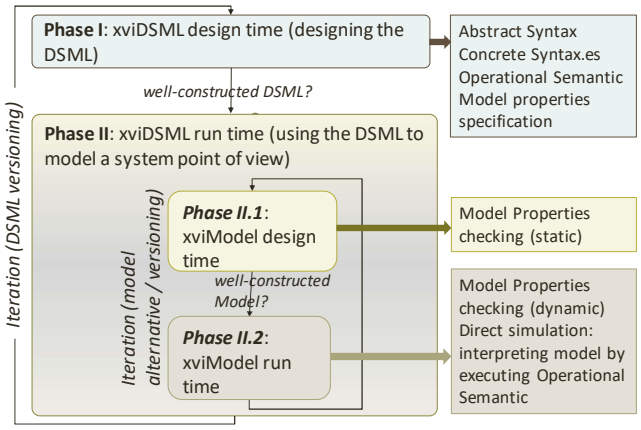

Fig. 9: *xviDSML* engineering and use: methodological overview

First of all, a static property MP2 must be checked prior to any other verification and using then same proof mechanism as it is done MP. This property is:

- MP2::=There is a priori one and only one crossable transition (static determinism) in the OMAG graph (the crossing conditions and the events associated with each exit transition of a mode that has been selected are to be compared and differentiated) i.e.

MP1 ::= ∀OperationalMode, M/M.isSelected= true, ∀OperationalModeTransition Ti∈M.outputTransitions ⇒XOR(Ti.condition)=true

Then MP3, MP4 and MP5 dynamic properties are to be formalized using UPPALL syntax then checked:

- MP3::=At any time t<>t0, there is a priori one and only one crossable transition (dynamic determinism) in the OMAG graph (the crossing conditions and the events associated with each exit transition of a mode that has been selected are to be compared and differentiated) i.e.:

MP3 ::= ∀OperationalMode M/M.isSelected= true, ∀OperationalModeTransition Ti∈M.outputTransitions ⇒XOR(Ti.condition(t))=true

- MP4::=At time t0 there is one and only one active mode i.e.

∃! OperationalMode M/ M.isSelected= true and M.isActive=true

- MP5::=At any time t<>t0 there is one and only one active mode (confirm dynamic determinism) i.e.

MP5 ::=∃! OperationalMode M// M.isSelected= true and M.isActive(t)=true

IV. Contributions: Methodological and Technical Aspects

First of all, engineering and using such *xviDSML* follows a process summarized in Fig. 9 making appear two phases focusing respectively on:

- *xviDSML* design time i.e. modelling language construction, verification and validation being conform to a meta meta model called then *xviCORE*;
- *xviDSML* runtime consisting to create, verify, and validate as much as possible the model of a SoI (model design time being then conform to the *xviDSML* definition), then use the resulting model (model run time) to check SoI.

Technically, intending to equip this *xviDSML* approach has been done as proposed in [12] and the meta modelling pyramid here applied to OMAG *xviDSML* (Fig. 10).

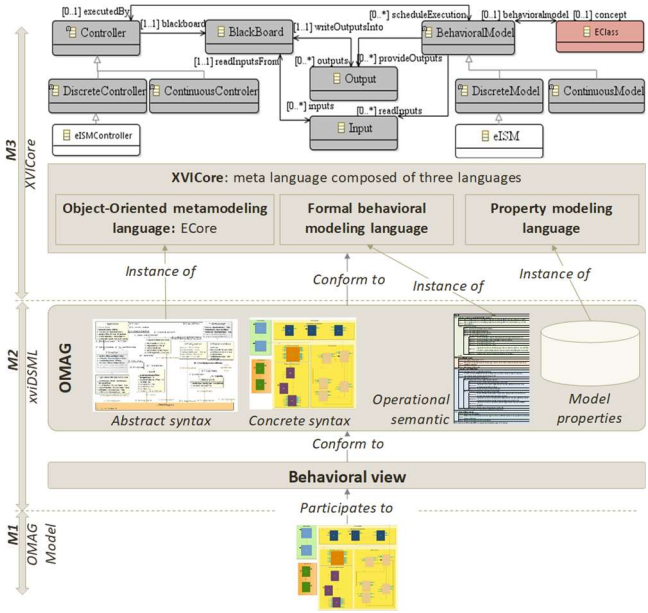

Fig. 10: *xviDSML* implementation: example of OMAG

V. Discussion and Conclusion

This article intents first to fix some concepts and definitions, even not new but expecting a consensual and common point of view when considering MBSE domain. Indeed, authors think MBSE remains poorly formalized, even if numerous DSML, tools and techniques exist actually, with no real convergence. Second, it proposes a methodology and an equipped environment, at least a proof of concept to demonstrate the interest of both contributions.

The goal is now to generalize and to focus on system model level i.e. to formalize federation and composition mechanisms allowing then stakeholders to dispose of and share without ambiguities a more complete representation of the SoI, then to converge on Digital Mock-Up concept and promote system early V&V [15].

VI. References

[1] ISO/IEC 15288:2015(E) / Systems and Software Engineering - System Life Cycle Processes. Geneva, Switzerland: International Organization for

Standardization (ISO)/International Electrotechnical Commission (IEC)/Institute of Electrical and Electronics Engineers (IEEE).

[2] Final report of Model-Based engineering subcommittee, NDIA, Feb 2011 (available on https://www.ndia.org/-/media/sites/ndia/meetings-and-events/divisions/systems-engineering/modeling-and-simulation/reports/model-based-engineering.ashx, last visited June 23th, 2022)

[3] INCOSE, Survey of Model-Based Systems Engineering (MBSE) Methodologies, Model Based Systems Engineering (MBSE) Initiative from International Council on Systems Engineering (INCOSE), June 2008

[4] INCOSE Systems Engineering Vision 2020, 2007

[5] Dave Kaslow, INCOSE Model-Based Systems Engineering (MBSE) CubeSat Modeling Efforts, Space Systems Working Group (SSWG) - GSFC Systems Engineering Seminar - June 2015

[6] Model Based-Engineering MBSE 101 - INCOSE IW January 30th, 2016, moderated by Elyse Fosse (available on https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:2016_iw-mbse_101.pdf, last visited June 23th, 2022)

[7] Gianni, D., D'Ambrogio, A. and Tolk, A. (eds) (2018) Modeling and Simulation-Based Systems Engineering Handbook, CRC Press, doi:10.1201/b17902

[8] ISO/IEC 29110:2015, Software engineering - Lifecycle profiles for Very Small Entities (VSEs), 2015

[9] IEEE Std 1471-2000 Recommended Practice for Architectural Description of Software-Intensive Systems, 2000

[10] B.Combemale, J.Deantoni, B.Baudry, R.France, J.-M. Jezequel et al., Globalizing Modeling Languages. Computer (IEEE), 2014, pp.10-13

[11] ISO/IEC/JTC 1/SC 32. ISO/IEC 19502:2005, Information Technol-ogy - Meta Object Facility. Multiple. Distributed through American National Standards Institute, 2007

[12] Nastov, B., Contribution to a tool-based method for the design of interoperable, analyzable and provable business modeling languages for Model Based System Engineering. University of Montpellier, 2016 [in English, accessible on https://hal.archives-ouvertes.fr/tel-01809000, last accessed June 30th, 2022]

[13] Natalia Sidorova, Master course on Process Modelling (available on https://www.win.tue.nl/~sidorova/pm/index.html, last visited May 10th, 2022)

[14] ISO/IEC CD 24641:2020(E, Systems and software engineering, Methods and tools for Model-based systems and software engineering

[15] Vincent Chapurlat, Blazo Nastov, Deploying MBSE in SME context: revisiting and equipping Digital Mock-Up, 6th IEEE International Symposium on System Engineering, 6th IEEE ISSE 2020, Vienna, Austria, October 12th-14th 2020

[16] J.-M. Penalva et al., Le systémographe - la méthode SAGACE, version 1.0, CEA, 2000 [in French]

[17] Friedenthal et al. 2009 - Friedenthal, S., A. Moore, and R. Steiner, A Practical Guide to SysML: The Systems Modeling Language, Needham, MA: OMG Press, 2009

[18] Functional Mock-Up Interface (FMI/FMU), see https://fmi-standard.org/ (last accessed: June 2020)

[19] David Gouyon, Stéphane Chaigneau, Jean-Marc Quiot, Nicolas Veaux, Vincent Chapurlat, A notation to measure and improve efficiency with regards to Integration: Integration Verification Validation Assessment Notation (IV²AN), 13th IFAC Symposium (INCOM '09), Moscow, Russia, June 3-5, 2009

[20] Pesola, J-P., Building Framework for Early Product Verification and Validation". Espoo 2010. VTT Publications 736, (2010)

[21] Chapurlat V., UPSL-SE: A Model Verification Framework for Systems Engineering, Computers in Industry, Computers in Industry 64 (2013), pp. 581–597

[22] ISO/IEC/ IEEE 42010:2011 Systems and software engineering - Architecture description, International Organization for Standardization, Geneva, Switzerland

[23] GEMOC Initiative (available on http://gemoc.org/ with a set of related publications, last accessed June 2020)

[24] Adrian Pop, Peter Fritzson, An Eclipse-based Integrated Environment for Developing Executable Structural Operational Semantics Specifications, Electronic Notes in Theoretical Computer Science 175 (2007) 71–75

[25] Nastov, B., Chapurlat, V., Dony, C., Towards V&V Suitable Domain Specific Modeling Languages for MBSE: A Tooled Approach, 26th Annual INCOSE International Symposium, Wiley pub., 2016

[26] Vincent Chapurlat, Nicolas Daclin, Proposition of a guide for investigating, modeling and analyzing system operating modes: OMAG, , International Conference on Complex System Design and Management CSDM 2013 December 2013, Paris

[27] OMG, 2015a. Meta Object Facility (MOF) Specification 2.5.1, (available at: https://www.omg.org/spec/MOF, last accessed on June 27th, 2002

[28] Ecore metamodel (accessible on http://www.kermeta.org/docs/org.kermeta.ecore.documentation/build/html.chunked/Ecore-MDK/ch02.html, last accessed June 30th, 2022)

[29] OBEO Designer (accessible on https://www.obeosoft.com/en/, last accessed June 30th)

[30] ACCELEO (accessible on https://www.eclipse.org/acceleo/, last accessed June 30th, 2022)

[31] Claudius Ptolemaeus editor, System Design, Modeling, and Simulation using Ptolemy II, 2014

[32] UPPAAL, integrated tool environment for modeling, validation and verification of real-time systems (accessible on https://uppaal.org/, last accessed June 30th, 2022)

[33] S.Mallek, Contribution au développement de l'interopérabilité en entreprise : vers une approche anticipative de détection de problèmes d'interopérabilité dans des processus collaboratifs, PhD Thesis University of Montpellier, 2014 [In French] (accessible on https://hal.archives-ouvertes.fr/tel-00666099, last accessed June 30th, 2022)