Verification, validation, qualification and certification of enterprise models: Statements and opportunities

Vincent Chapurlat, Christian Braesch

Verification, Validation, Qualification and Certification of Enterprise Models: statements and opportunities

V.Chapurlat[a,*], C.Braesch[b]

[a] *LGI2P - Laboratoire de Génie Informatique et d'Ingénierie de Production, Site EERIE de l'Ecole des Mines d'Alès, Parc Scientifique George Besse, 30035 Nîmes cedex 1*

[b] *LISTIC - Université de Savoie - BP 80641 Avenue de la Plaine - 74016 Annecy Cedex*

Abstract

The first aim of this article is to demonstrate the high level of dependency between modeling activities and VVQC activities (Verification, Validation, Qualification and Certification) during the entire course of a project in an enterprise. It therefore aims to show the interest and relevance of VVQC to the enterprise modeling domain, before highlighting the objectives to be achieved and requirements to be fulfilled by future work on VVQC. The first part introduces the concepts and definitions required. The second part describes the state of the art concerning the uses, best practices and main current research related to VVQC in the enterprise field. The third part proposes some orientations for future research to be prioritized in order to deal with crucial challenges in the enterprise. It appears that these research orientations have been applied for some years by both academic and industry-based researchers. © 2007 Elsevier B.V. All rights reserved

Keywords: Enterprise modeling, Verification, Validation, Certification, Qualification

1. Introduction

An enterprise is a complex socio-technical system which must remain competitive in an unsettled market. Therefore, it must become more reactive, and flexible in an unpredictable environment while remaining attractive to its customers.

Various industrial challenges highlighted in [1] have to be fulfilled before 2020 in order to help the actors of the enterprise to reach their objectives. For example, several research works focus on increasing the interoperability of systems [2, developing new rules of thumb about virtual organization [3], improving the application of distributed control [4] and so on.

Several of these works stress the need to manipulate and analyze enterprise models [5]. Consequently, it is important to raise crucial questions about the quality and relevance of these models before considering their transferability to the enterprise. These questions have already been posed in other domains such as software engineering, electronic system engineering and system engineering [6]. The goal of this verification, validation, qualification and certification (VVQC) is to prove the suitability of a model by means of its requirements, objectives, construction rules, usages and so on. However VVQC remains little recognized and used in Enterprise Modeling. This paper therefore proposes:

- To show the need to link any part of

* Corresponding author Tel. (+33) 466 387 066 / email: Vincent.Chapurlat@ema.fr

enterprise modeling to an act of VVQC throughout the various project steps,

- To assess the state of the art regarding the uses, best practices and works in progress in this domain,
- To propose priority research orientations for the coming years concerning both conceptual and methodological aspects.

2. Context and requirements

This section establishes the links between enterprise modeling and VVCQ tasks and the requirements that need to be taken into account by these tasks. In the first part, we define the concept of a project in detail. We consider that most enterprises various to manage projects in order to solve different problems: organization, behavior of a production unit, design of new products, etc. In the second part various kinds of models identified by this framework will be described. The third part shows how those models are characterized and used to understand and modify enterprise systems. Then, before completely defining VVQC, the third part identifies the VVCQ requirements for enterprise projects.

2.1. *Enterprise projects*

GERAM [7] proposes a three-axis reference framework, summarized in Figure 1, which is interpreted here as follows.

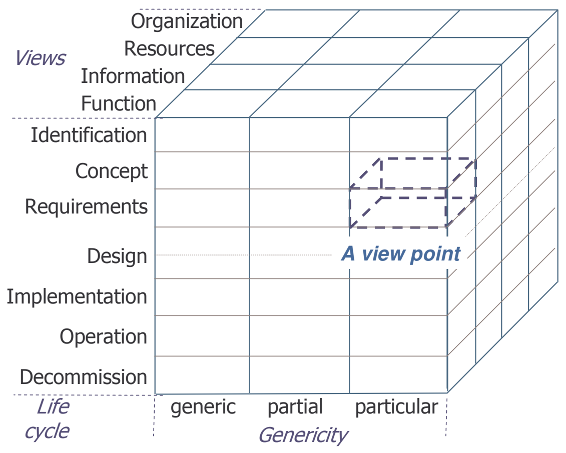


Figure 1: three-axis reference framework (inspired by GERAM [7])

This framework formalizes the aspects that actors need to take into account when they manage such projects according to the objectives of the project. This is done by means of:

- Different views through which the objectives of the project must be considered: functional, resource, information or organization views,
- A sequence of steps to be performed in order to cover project life cycle requirements. However, some of these steps may not be required in a particular project. For example, designing a new product does not require the same steps as improving logistics.
- Different levels of abstraction, from generic to particular, again according to the objectives of the project.

2.2. *Enterprise models in enterprise projects*

Each element of the matrix identifies a particular viewpoint which requires one or several models. In other words, each viewpoint contains model(s) of the aim of the project. These models are dedicated to a given step in the project, are seen at a given level of abstraction through a specific view.

They are built, on the one hand, to take into account the requirements and objectives of the step and, on the other hand, to allow the various actors involved to work together.

Commonly, these models are classified in three categories in function of the life cycle axis, as shown in Figure 2:

- AS_IS models,
- TO_BE models,
- IMPLEMENTATION models.

The AS_IS model describes the current situation that has to be taken into account to achieve the intended objectives. This model is built during the identification and concept steps of the project. It is used in the next steps of the project.

The TO_BE model describes the structure, functions and behavior of the solution that has been designed. This kind of model is built during the requirements and design steps of the project

The IMPLEMENTATION model then describes the equipment, devices and resources that have to be used to implement the solution. This model is built and used during the implementation and operation steps and is required during the decommission step.

Finally, each step in a project overlaps with the others. As a result, the AS_IS, TO_BE and IMPLEMENTATION models are built, modified and improved simultaneously rather than sequentially. In the enterprise context, these models are used to adapt the enterprise structure, operation or behavior to frequent changes in the environment.

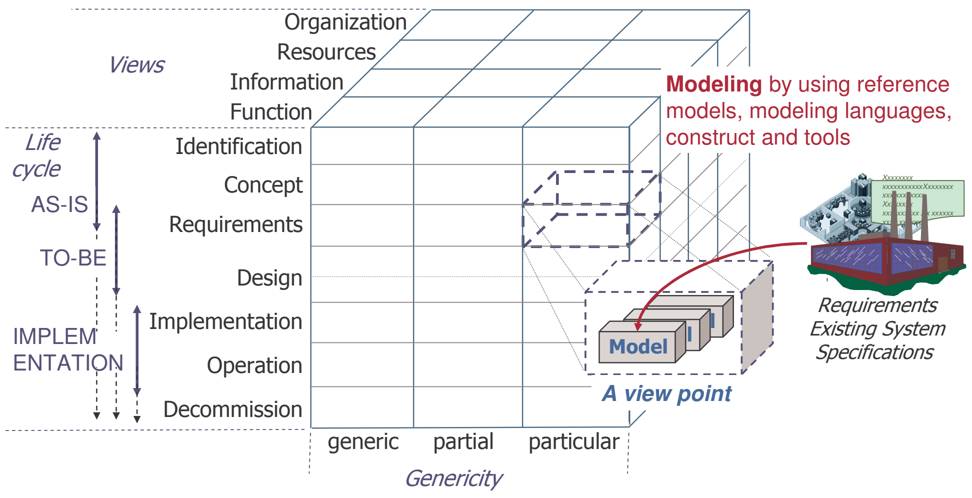


Figure 2: AS-IS, To-Be and IMPLEMENTATION models

2.3. *Enterprise models adapted for enterprise projects*

First of all, the enterprise is considered as a system in order to take into account its complexity. Most current approaches formalize the enterprise through various projects, providing a global behavior for fulfilling a given mission, taking into account:

- Constraints due to possible divergence between the missions of the enterprise and its actors' expectations.
- Constraints due to the interactions between resources, flows, activities and processes.
- Constraints due to interactions between the enterprise and its environment (suppliers, customers and other partners).

These different kinds of constraints determine the capability of the enterprise to react to changes, and require the enterprise to be considered as a system.

A system is defined as the representation of an active phenomenon that can be identified by its projects in an active environment in which it evolves [8].

General system theory provides a framework for modeling a phenomenon to make it more intelligible and facilitate its modification.

Consequently, in an enterprise context, these different models are used [9]:

- to understand the current situation in an enterprise,
- to design a new enterprise or a new part of an existing one,
- to modify an enterprise to adapt its behavior to environmental changes.

The **intelligibility of an existing system** is necessary for several reasons:

- To understand why a dysfunction occurs.
- To compare a current situation with an expected one for controlling a process.
- To characterize an initial state when a change is planned.
- To check the conformance with a specific standard or norm (for example, ISO 9000 or ISO 14000).

In this case, the actors need to have confidence in the model to be sure that their arguments are based on reliable premises.

The **design of a new system** requires the ability, on the one hand, to define its mission and, on the other hand, to guarantee that the system will be capable of fulfilling this mission whatever the situation during its life cycle. In this case, the TO_BE model describes the structure, the functions and the behavior of the future system. However, this model must be interpreted, taking into account technological constraints and existing solutions, in order to produce the IMPLEMENTATION model. This describes the equipment, devices and resources of the future system. In this case, actors must have confidence firstly in the relevance of the model and secondly in the process of transformation from the TO_BE to the IMPLEMENTATION model without loss of sense.

A **system modification** is required when the system is not capable of fulfilling the intended mission or when the environment changes. In other words, the enterprise must enact continuous improvement processes. This approach requires being able to manage several concurrent projects in order to understand the causes of the problems, to find solutions and to implement them.

System Theory [8] gives us a framework for studying the modification of a system, shown in Figure 3. This framework identifies four cases by considering whether or not the mission or the environment changes.

Regulation defines a project that maintains the system in a stable state. In this case, the actors should know whether their projects are consistent.

		Mission	
		Constant	*Variable*
Environment	*Constant*	*Regulation*	*Structural adaptation*
	Variable	*Project adaptation*	*System evolution*

Figure 3: Control framework of a system [8]

Project adaptation is performed when the current project and steps are not able to fulfill the intended mission. This case requires firstly an evaluation of the current state of the system and secondly correction of the project by:

- Deleting actions.
- Modifying actions.
- Adding new actions.

Most of the time, it is necessary to modify the sequence of actions. Then, the actors should be able to verify that the new project fits the mission. For example, a model should be able to prove to the customer that his or her requirements are fulfilled by a given solution.

Structural adaptation is applied when changes in the environment lead to the definition of new objectives. In some cases, project adaptation is not sufficient and it is necessary to define a new organization to supply new functions and new behaviors. The actors must then check the consistency of the new structure and its capability of fulfilling the new missions.

System evolution is a combination of the last two cases (project and structural adaptation). The modification of the system therefore relies on the AS_IS, TO_BE and IMPLEMENTATION models.

This part has shown the different use of models in the enterprise context and has highlighted some actors' expectations. The next part will describe more precisely the requirements of VVCQ

2.4. *From enterprise models to VVQC requirements*

We describe the different VVCQ requirements expected in the enterprise models through different situations.

Firstly, during a design step, several simulation models may be necessary in order to test or to compare the performances of different alternative solutions. However, these models may represent the same system taking into consideration different view as proposed in Figure 1. So, these models must be coherent, and not contain ambiguities or misunderstandings.

Secondly, a model can result from the decomposition of another one in order to detail, for example, a part of the enterprise or a given behavior. Each of these models must remain coherent with the model corresponding to the highest level of detail.

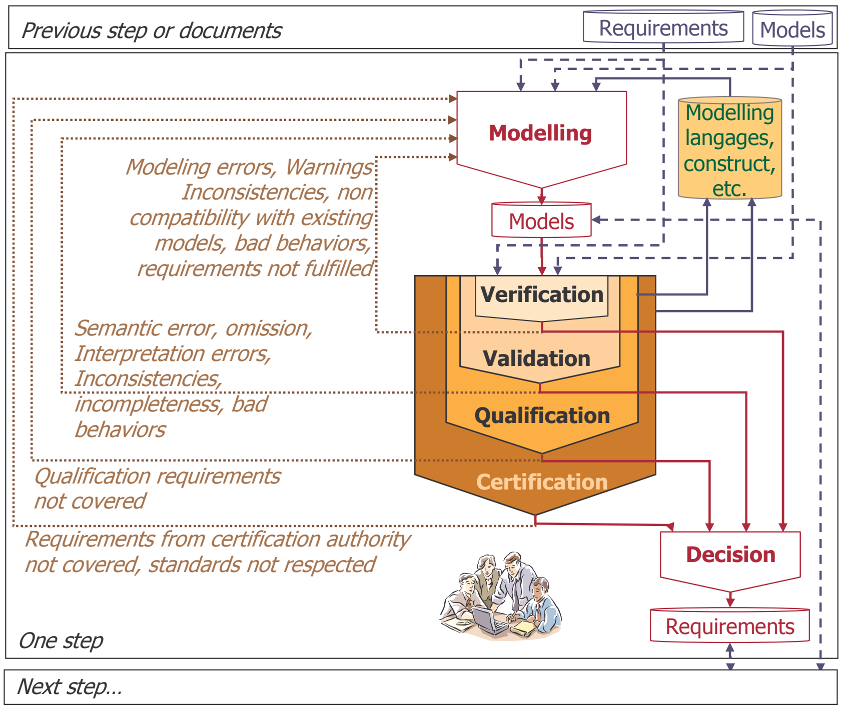
Lastly, different modeling languages are usually used to take into account objectives, life cycle, actors' competencies and available time for the modeling tasks. These modeling languages do not use the same syntax, do not share a unique semantic, are not available using the same tool and so on. In the enterprise-modeling domain, there is a plethora of modeling languages, constructs and approaches [10] used during the different steps of a project. So, many types of models can emerge from the project.

Let us make a parallel with the software engineering domain to illustrate this problem. Software engineering introduces two kinds of refinement of models. Usually, a distinction is made between vertical and horizontal refinements.

The first type covers the decomposition of a model, which is used to add new details using the same modeling language. In this case, mechanisms are required to prove that the new representation is coherent with the initial one. For example, IDEF0 uses the MECS checking approach to guarantee that a child diagram is coherent with its parent.

The second type covers the rewriting of a model in another modeling language. For example, the TO_BE model has to be transformed, or translated respecting technical constraints or existing solutions, into an IMPLEMENTATION model. In this case, refinement enables technical details to be added and new viewpoints about the future system to be highlighted. Refinement therefore enables the use of specific modeling concepts different from those used during the TO_BE model design. These concepts are required for unambiguous modeling of the same real object view from different viewpoints. There is therefore a semantic distance between the modeling languages that needs to be taken into account. In this case, actors must be confident of the mechanisms which guarantee that the properties of each model are maintained during a transformation.

Vertical refinement is used to facilitate exchanges communication between actors, model exploitation and even model re-use in the case of very similar problems. Horizontal refinement is used to obtain more and more details about a given phenomenon, i.e. to facilitate the understanding of the actors.



Previous step or documents
Requirements
Models
Modelling
Modelling langages, construct, etc.
Models
Modeling errors, Warnings Inconsistencies, non compatibility with existing models, bad behaviors, requirements not fulfilled
Verification
Semantic error, omission, Interpretation errors, Inconsistencies, incompleteness, bad behaviors
Validation
Qualification
Qualification requirements not covered
Certification
Requirements from certification authority not covered, standards not respected
Decision
Requirements
One step
Next step...

Figure 4: A step in a project

The actors involved in a given step have several requirements:

- They must have a demonstrated and good (or sufficient) level of confidence in each model (good construction, no errors of omission, no mistakes linked to the modeling process, fidelity, completeness when possible and relevance of the model).
- They expect improved interoperability between models by ensuring the coherence of all the possible models and of all of the possible detail levels of a given enterprise model.

So, the information and knowledge incorporated into the models must firstly be verified and secondly validated. When required, this information should be qualified for the actor's use and certified if it correspond to standardized and/or re-usable information in the domain.

3. VVQC in enterprise modeling

3.1. *Generalities*

Any project activity in an enterprise, as shown in Figure 4 is a sequence of Modeling, VVQC and Decision/Action activities.

It involves a group of actors from the problem formulation stage (understand, validate, manage) till the exploitation of the obtained model(s) in order to define an (a set of) action(s) to solve the problem. In the enterprise modeling context, taking into account the role of the "AS-IS", "TO-BE" and "IMPLEMENTATION" models, **Erreur ! Source du renvoi introuvable.** shows where and when VVQC activities are required.

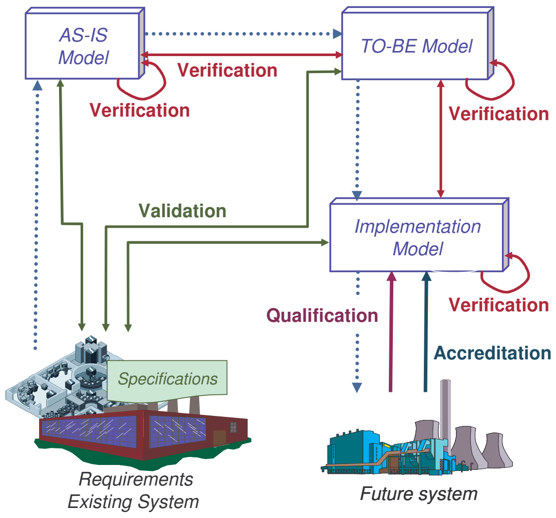


Figure 5: Verification, Validation, Qualification and Certification in enterprise modeling

These activities can thus be defined as follows by taking into account [11,12,13]:

- **Verification**: as illustrated in Figure 6 the goal is firstly to detect any mistakes by focusing on a given model; that is to say to prevent any misunderstanding concerning the interpretation of the meta-model during the modeling task. Secondly, it aims to detect any incoherence between two models which must, for example, respect the same set of properties. So, the outputs of the verification activity may be a correctly built model or a set of errors, mistakes or warnings about model structure or behavior, or about the coherence and thus non-alignment of two models. In all cases, the modeling task must be reinitialized in order to improve the model. Concerning coherence, as proposed in the previous section, verification enables the following aspects to be demonstrated:

– The coherence between models obtained by decomposition i.e. the horizontal refinement principle. In this case, the goal of refinement should be to characterize in more and more detail the behavior (for example the behavioral decomposition of an activity), the structure (for example a description of the internal organization of a given organizational unit), the function, the topology, etc. of an enterprise part. It uses the same modeling language throughout the model decomposition.

– The coherence between models obtained by vertical refinement; that is to say after a rewriting phase or interpretation phase using different modeling languages. For example, the two models may describe the same phenomenon in the enterprise, but one model may investigate its effects while the other is used only to simulating the behavior of the identified system in which the phenomenon occurs.

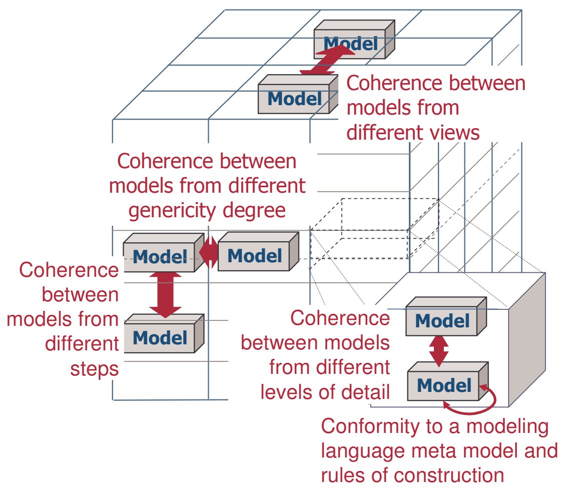

Model
Model
Coherence between models from different views
Coherence between models from different genericity degree
Model
Model
Coherence between models from different steps
Model
Coherence between models from different levels of detail
Model
Model
Conformity to a modeling language meta model and rules of construction

Figure 6: Verification

• **Validation**: as illustrated in Figure 7 the goal of validation is to demonstrate that the model is an accurate and relevant representation of reality and that it takes into account the specified requirement coming from the enterprise part. The outputs of this activity are exploited directly in a decision activity. So, the 'quality' of these outputs, i.e. their adequacy with the system and their interest with respect to the project objectives to be solved, directly impacts the decision process especially when a human interpretation of these outputs remains necessary, as shown below. This activity enables the detection of any semantic errors or omissions which could interfere with the interpretation of the human expert. This again requires improving the model.

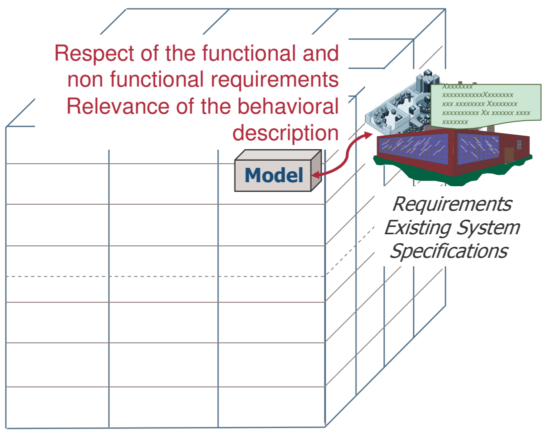

Respect of the functional and non functional requirements
Relevance of the behavioral description
Model
Requirements
Existing System
Specifications

Figure 7: Validation

• **Qualification**: The model needs to be interpreted with a sufficient confidence level. Knowledge incorporated into the model must be reused without loss or bad interpretation by actors coming from different domains and involved in other decision processes in the enterprise.

• **Certification**: as illustrated in Figure 8, the model is then recognized by an authority as sufficiently relevant and generic in order to establish or to furnish the base for a certification goal in different domains.

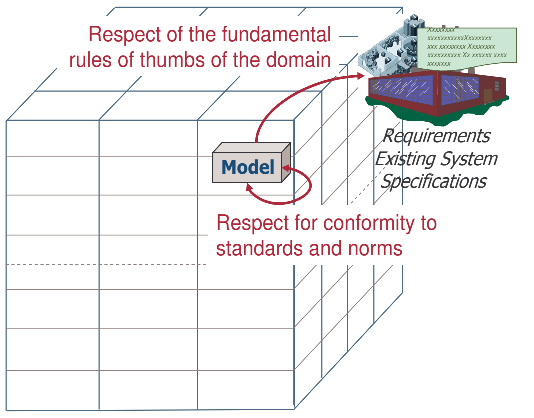

Respect of the fundamental rules of thumbs of the domain
Model
Requirements
Existing System
Specifications
Respect for conformity to standards and norms

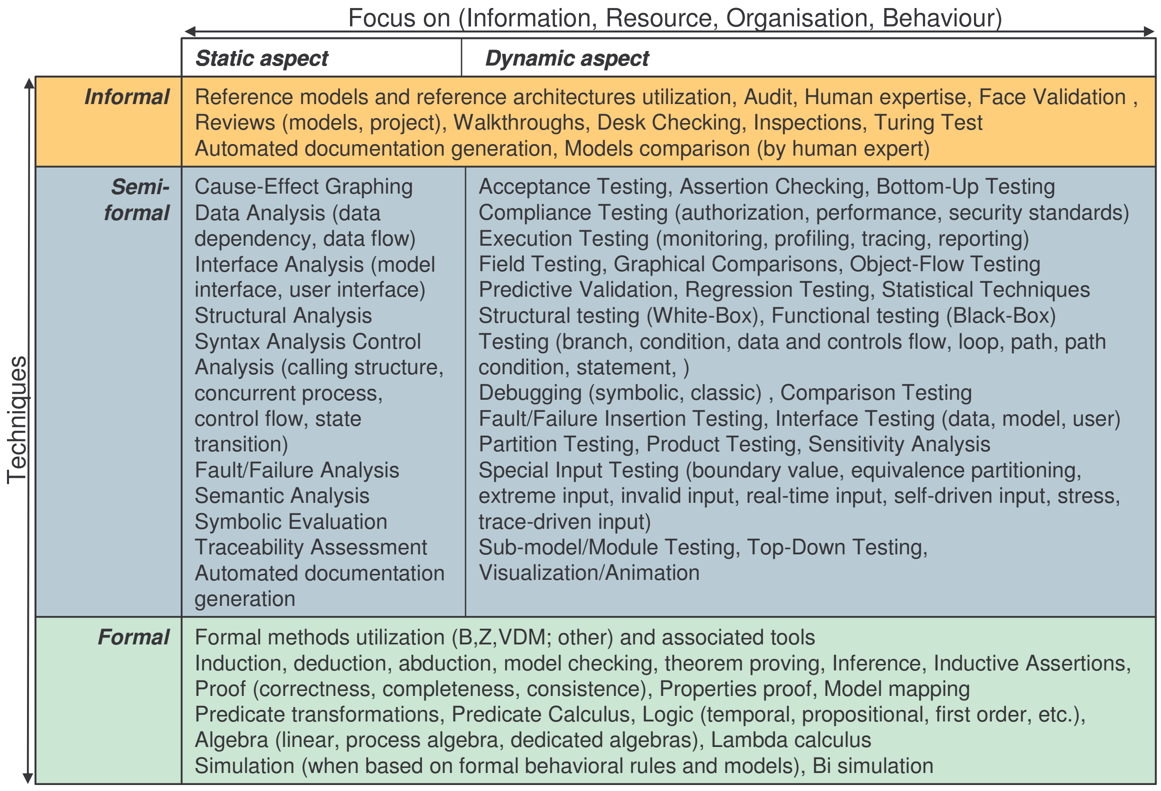
Figure 8: Certification

As suggested by Figure 4, a model must be verified as fully as possible before being validated. This validation is required in order to decide if the model can be qualified for some other purpose. Lastly, certification, which is an 'external' acknowledgement of a model by a community, should as far as possible be based on a validated and qualified model which is a model of the enterprise recognized and shared by all the enterprise actors. This requires a new encapsulation of all these activities, a new process henceforward called VVQC, linked to the modeling and decision processes. Of course, not all these activities are required for all projects. However, verification and validation cannot be missed out.

3.2. *Interests*

The implementation of VVQC activities presents a number of interests for the actors involved. The following section gives some examples:

- **Decision-making process support**: Let us consider project aiming on for example reorganization of a workshop to respond to an opportunity, acquisition of additional resources, modification of the resource control code, schedule optimization, etc. Any decision taken during these projects is usually based on the evaluation or interpretation of models of the problem. It is thus necessary to obtain a relevant and reliable model, i.e. checked and validated, before it can be exploited in a decision task. It is then possible to use the same checking mechanisms (coherence in a given situation or configuration) and validation mechanisms (situation parameter settings, simulation, expertise, etc.) to exploit the model.
- **Interoperability** [2,14,15,16,17] which is defined according to [13] as `the ability of two or more systems or components to exchange information and to use the information that has been exchanged'. ‘Physical related interoperability’, ‘information related interoperability’ and ‘semantic related interoperability’ are three different formulations of interoperability problems. However, existing standards and norms describe physical formats, interfaces between components, connectors, protocols, etc. so physical related interoperability can be assumed by respecting these standards independently of technological aspects. Information and semantic related interoperability problems remain important. For example, the enterprise networking approach involves several enterprises that need to exchange and interpret data, information and knowledge at all times. To achieve this, the various members of the network share data models, protocol transfer models etc. that have to be recognized with a high level of confidence, safety and security by all members. More precisely, within the more operational framework of information systems, [18] defines interoperability as `*the capacity to communicate, cooperate and exchange models or data between two or several applications in spite of the differences in the languages of implementation, the environments of execution or the models of abstraction*'. It is thus important to ensure the conformity of a service, the adequacy of data or the coherence of an exchange (from a temporal or semantic viewpoint). Checking and execution mechanisms for the models describing these services, data or exchanges are therefore necessary. This position is implicit in a universe where *everything is a model* as proposed in the MDA approach (Model Driven Environment) [19,20]. This considers models to be the basis for any communication, so any transformation between models must be described as a new model itself.
- **Certification** [21,22]: certification processes require representations of enterprise parts, for example the organization of processes required by ISO for ISO 9000-9002. The authority in charge of this certification is thus implicitly involved in a VVQC process.
- **Knowledge management**: Particularly during verification and validation, the actor in charge of these activities often requires more information or knowledge than that contained in the model. Indeed, in order to achieve sufficient proof of results and confidence, it is necessary for the actor to include and understand additional information, not incorporated in the model, which considerably opens the actor's field of vision. For example, validating a communication protocol model may require more information than that contained only in the model describing the sequence of the phases of the protocol.
- An **information system** is a system that is at the same time technical, technological and human charged with collecting, storing and then processing an ensemble of information according to established procedures. An information system is now considered as a system of knowledge management. It can ensure the coherence of the data, information and knowledge, their integrity, their enrichment (by cross-matching mechanisms, fusion, exploration by data mining techniques , etc.) and their provision at the right time and place using recognized interchange formats like XML (eXtensible Markup Language) [23]. Information systems are currently becoming more and more open to new means of communication induced by: the needs of the market (electronic trade); the development of emerging and portable technologies; the need for nomadism, division of information and navigation via the Web using information portals; the use of decentralized and distributed systems such as data warehouses etc. We are seeing the transition from a `data driven environment' to a `co-operative information and knowledge-driven environment'. The engineering of such a system is a complex project in which it is impossible to circumvent the concepts of VVQC [24,25]
- **Risk management** [26]: the objective here is to formalize operational, technical, financial, human or organizational risks and to help the actors involved in a project to detect the emergence of such situations.


Focus on (Information, Resource, Organisation, Behaviour)
Techniques

	Static aspect	Dynamic aspect
Informal	Reference models and reference architectures utilization, Audit, Human expertise, Face Validation , Reviews (models, project), Walkthroughs, Desk Checking, Inspections, Turing Test Automated documentation generation, Models comparison (by human expert)	
Semi-formal	Cause-Effect Graphing Data Analysis (data dependency, data flow) Interface Analysis (model interface, user interface) Structural Analysis Syntax Analysis Control Analysis (calling structure, concurrent process, control flow, state transition) Fault/Failure Analysis Semantic Analysis Symbolic Evaluation Traceability Assessment Automated documentation generation	Acceptance Testing, Assertion Checking, Bottom-Up Testing Compliance Testing (authorization, performance, security standards) Execution Testing (monitoring, profiling, tracing, reporting) Field Testing, Graphical Comparisons, Object-Flow Testing Predictive Validation, Regression Testing, Statistical Techniques Structural testing (White-Box), Functional testing (Black-Box) Testing (branch, condition, data and controls flow, loop, path, path condition, statement,) Debugging (symbolic, classic) , Comparison Testing Fault/Failure Insertion Testing, Interface Testing (data, model, user) Partition Testing, Product Testing, Sensitivity Analysis Special Input Testing (boundary value, equivalence partitioning, extreme input, invalid input, real-time input, self-driven input, stress, trace-driven input) Sub-model/Module Testing, Top-Down Testing, Visualization/Animation
Formal	Formal methods utilization (B,Z,VDM; other) and associated tools Induction, deduction, abduction, model checking, theorem proving, Inference, Inductive Assertions, Proof (correctness, completeness, consistence), Properties proof, Model mapping Predicate transformations, Predicate Calculus, Logic (temporal, propositional, first order, etc.), Algebra (linear, process algebra, dedicated algebras), Lambda calculus Simulation (when based on formal behavioral rules and models), Bi simulation	

Figure 9: Overview of techniques [32,33,34,35]

Generally, modeling approaches provide limited concepts for modeling or analyzing risk. It may be interesting to become able to check models in order to analyze the occurrence of risks, their possible causes, and their impacts and effects on the performance of the system.

- **The generation** of **software applications** such as Workflows [27], EAI [28,29] or ERP systems [30]. The Implementation model makes it possible to obtain a more rigorous and safe code covering the requirements without any doubt or misunderstanding, ensured by means of checking techniques. For example, these techniques may consist in assisting the diagnosis of modeling errors using simulation tools. In the same way, it is interesting to analyze the role of resources, as suggested in [30], so as to analyze dysfunctions and impacts on the system situation.

3.3. *Techniques*

These VVQC tasks are supported using several techniques some of which are summarized in Figure 9, taken from [32,33,34,35]. These techniques can be classified according to the formal level of analysis they provide (non formal, semi formal or formal) and the aspect of the enterprise concerned by the project: information, resources, organization or behavior.

Finally, all the proposed techniques can be also classified according to the point of view to which they are most appropriate.

Firstly, a technique may focus on the model to be analyzed considering only a static point of view i.e. without requiring the execution of the model. This avoids construction errors, mistakes or omissions, enables the detection of syntactic anomalies or semantic problems and, more generally, can check that systems or models comply with construction standards. These techniques can be used during the verification or validation tasks.

Secondly, a technique may require the execution of a model to evaluate its operational semantic. This aspect requires the use of formal modeling language to determine the validity of the model by studying the output data corresponding to data inputs or orders. For this type of analysis, the techniques used can be based on simulation, emulation or testing. This prevents bad behaviors, detects deadlocks or evaluates the performance level of the modeled system i.e. it essentially covers the validation and qualification goals.

- **Non formal techniques**

These are essentially based on human expertise and do not require such a high level of formalization of the model.

They consist in discussing and appraising the model within the framework of reviews, meetings or by using certain simple tools like the automatic generation of documentation starting from the model. So, they focus essentially on model validation. In enterprise modeling, this kind of technique is the most commonly used approach and the best practice.

It can be based on:

– An anticipative way of modeling, which may consist in strictly guiding the modeling tasks of the project. Indeed, the actor who models the system must strictly respect given reference models [36,37,38], the maturity of enactment rules [39] modeling rules, reference architectures or standardized constructs [40] even simple rules of modeling and good practices such as proposed in [41]. A model is thus built up on the basis of more or less rigorous groundwork. This reduces errors or mistakes and the occurrence of warnings, and of course can avoid incorrect interpretation of the modeled system on the part of the modeler and avoid irrelevant models.

– After modeling the pointed out system, a human expert can check the model. Indeed, a model filters reality. It disregards certain detail levels considered as non necessary and arbitrarily selected. So, the knowledge and the know-how of an expert or a group of experts can interpret this model or interpret results resulting from its simulation, and draw a certain amount of additional information from it. If the expert has a long experience on using some modeling languages or modeling frameworks, this can be used to check the model. The expert must respect some experimental construction rules, and the syntax and semantic of the domain. In other cases, it can be used to validate and certify a given model with a given level of confidence.

- **Semi formal techniques**

These techniques are essentially based on model execution. They remain a popular approach [42,43] and are even regarded as sufficient and relevant in a large number of projects [44,45]. However, their appropriateness is arguable in some cases. Indeed, they require the formalization of a single and unambiguous operational semantic fundamentally defined by a set of formal execution rules, temporal hypotheses and initialization rules for the model. This enables effective execution of the model a high level of confidence in the execution results. In addition, it frequently requires the development of dedicated and often specific models known as simulation models [46], and the definition of scenarios defined a priori, thus excluding objectively or subjectively `forgotten' behaviors that it would however be interesting to analyze.

In the same way, emulation consists in translating a model into another one for which execution tools and an operational semantic already exist. This requires unambiguous and formal rules of translation making it possible to produce a model equivalent to that under study. The techniques of analysis based on simulation, emulation or, more generally, the testing of models through execution, are widely used in experiments to check that correct results are obtained starting from a sample of initial data. This may reassure the user but does not exclude erroneous results because of the non-exhaustiveness of the scenarios taken into consideration.

- **Formal techniques**

Such techniques are strongly related to formal methods [47,48,49,50]. They require a high level of formalization of the modeling language. In other words, the modeling language used must be equipped with an adequate mathematical semantic based on interpretation rules which guarantee the absence of ambiguity in the descriptions produced and deduction rules which make it possible to reason about the specifications in order to discover potential errors, mistakes or inconsistencies by proving properties [51,52].

Some formal approaches implementing a formal language have been used, such as the Z method [53], B method [54], VDM method [55], or derived approaches such as Object-Z [56, 57], inspired by both Z and the object paradigm. For example, B and Z have been employed in manufacturing system engineering [58] to establish design patterns i.e. reference models for production system structuring and modeling. Certain work in enterprise modeling was directed inspired from formal approaches. We can also cite the Process Specification Language PSL [59] which includes mechanisms for checking the specifications.

In the same way, other works [60,61] tackle the problem of process formalization. The Albert II modeling language [62,63] enables the formal description of enterprise requirements in order to facilitate their validation by means of the generation of scenarios. Lastly, as suggested in [64,65], algebras have been developed in order to describe manufacturing processes.

In all cases, these approaches can provide actors with rigorous proof or, on the contrary, can highlight any problem in the model. Two main techniques are then available: model checking and theorem proving.

Model Checking is intended for the checking of behavioral models of systems modeled using states automata. Given an ensemble of such automata and a property, this kind of tool explores the whole set of reachable states in order to check that the desired property is satisfied. If a property cannot be checked, the sequence of transitions from driving state to property violation is generated as a counterexample, showing that the system is incorrect. A property is thus regarded as a formalized statement of expectation by means of a formal language. This one may may be for example a temporal logic [67,68] or a specific logic such as HOL (High Order Logic). It can be also inspired by communicating process concepts as suggested by C.A.R. Hoare with CSP (Communicating Sequential Process) [69], Estelle [70], Promela which is used in the tool SPIN [71], LOTOS or process languages in an algebraic form (mixing LOTOS and Petri nets) in CADP (Caesar Aldebaran Development Package) [72]. Lastly, it can use more abstract languages inspired by the μ-calculus or Pi calculus approach [73].

However, model checking is often limited by the combinative explosion of the number of possible states in a sequence. Several works therefore propose rules of abstraction [74], substitution, partial projection of the model or simplified assumptions for which traceability of proof is possible.

On the other hand, Theorem Proving is based on the specification of a system model by using a formal description. It thus consists in progressively creating a mathematical proof. In other words, it is a question of describing the property to be checked in the form of a theorem, stated using logic as in model checking, and of showing that it can be directly deduced from the specification of the system and the axioms by means of the deduction rules suitable for the logic used.

Moreover, such techniques can be implemented by means of quite a vast panorama of tools, described in [75]. These are tools of the "theorem prover" or "model checker" type. These tools often support both the description of the model and the properties to be checked, the formal checking mechanisms for proving those properties and, for many tools, validation starting from an instantiation of the model according to different scenarios. This is the case, for example of the B Tool, Z-Eves [76] or even STEP [77,78] which is based on different temporal logics. There are also more indirect approaches based on the translation, refinement or rewriting of models in an intermediate language, as proposed in another domain in [79].

Several Model Checking tools like SMV [80] or PVS [81] are used at the industrial level for example within the framework of critical applications in safety [82,83,84].

The following tools allow also to handle theorem proving functionalities. For example, Z-Eves [76] can check the consistency of Z specifications, Step [77] can check properties described using temporal linear logic. We can also mention the tools COQ [86], Otter [87], ACL-2 [88] which supports High Order Logic (HOL) in its various versions (from the HOL80 version to the current version HOL 4 [89]), and the environment called Isabelle [90], also based on HOL or SPIN.

However, a theorem prover is generally regarded rather as an “evidence assistant” usually helping to guide the user to build a proof. Indeed, a theorem prover cannot be considered as an automatic proof support system.

The use of formal methods therefore remains a delicate issue. They are relatively expensive in time in order to reach a good level of proof [91,92] and are obviously not always usable or relevant, depending on the project of the modeler.

Moreover, the modeler must have a good knowledge of the tools, and at the same time of concepts that are often relatively abstract or at least not easily comprehensible or comparable with reality for the other actors involved in the project. The required knowledge about mathematical of formal theory is always more or less complex. Last the problem of non decidability in certain cases of proof remains an obstacle to their use in certain contexts. For example, a non synchronism in parallel systems can only be checked using certain assumptions.

There is therefore a real contradiction between the completeness or exhaustiveness offered by formal methods and the required level of confidence, the effective cost, the time and abilities needed, etc.

This comparison of these different techniques allows us to highlight some problems to be solved in order to adopt and become able to support VVQC process in any kind of project. There are some opportunities of development for future research axes in the enterprise modeling domain.

4. Proposed research axes

This part aims to propose research axes that should be investigated in the enterprise modeling domain. These axes are defined from the requirements identified in the previous sections, which show the interest of VVQC tasks, methods and tools.

In the enterprise domain, projects are planned to allow the enterprise to become more flexible, attractive and robust, and to respond to evolutions in the market and customer’s demands. Each project step requires a modeling task to obtain a representation of the part of the enterprise concerned.

To be able to propose, decide and control the actions that need to be performed, actors should have:

- To prove the coherence of a model whatever it may be (AS_IS, TO_BE or IMPLEMENTATION), the modeling language used and the viewpoint described in the model. This requires checking the syntax (not relevant here), meta-model conformance, consistency, non ambiguity and non redundancy of the model, and also the correctness of the model and its compliance with existing standards.

- To prove horizontal coherence:

  – First case: between models which represent two different viewpoints of the system. This may be done by using different modeling languages often dedicated to a given viewpoint, sometimes adapted for several simultaneous viewpoints. These modeling languages handle concepts that sometimes reflect the same semantic but have different names, describe the same class of objects in the real world but define them differently and so on.

  – Second case: between two models, one of which was produced by the rewriting, partial or otherwise, of the other. The two models aim to describe the same system but use two distinct modeling languages.

- To prove vertical coherence:

  – First case: between two models, one of which was obtained by decomposition of the other one in order to describe a more detailed level of the same enterprise part from a given viewpoint, using the decomposition rules of the same modeling language.

  – Second case: between models based on different modeling languages but all to describing the same view point.

- To ensure, when possible, the completeness of a viewpoint representation.

- To prove, if they are specified, that the functional and non-functional requirements coming from the final user(s) of the system are covered, that is to say the model respects these requirements.

- To establish, when possible, or to improve the relevance of a viewpoint representation by means of the system itself.

- To qualify the model for internal use or re use in a future project, and to make possible to accredit the model by an external authority

- To facilitate and then to improve the efficiency of modeling tasks: reducing loss time, and eliminating errors and omissions. Indeed actors must be able to modify a model if, for example, V&V tasks conclude that its lacks definition. Actors also need to have increasingly precise information on real requirements as a given step continues. Lastly they have to take into account the enterprise environment or even technological evolution. So, actors must be able to:

  – To ensure the model evolves more rapidly and more coherently. To do this, a model needs to remain relevant, coherent and consistent throughout the re-modeling task.

  – To test and ensure alternative solutions without excessive modeling work and to show that the expected levels of performance, stability and integrity [93] can be achieved by the system.

  – To have a better understanding of possible emerging situations and risks due to the unpredictability of interactions between the system and the other systems composing the entire enterprise.

Of course, the constraints remain the same for any project in an enterprise: results must be achieved at minimum cost and as far as possible by using actors' existing competencies, knowledge and know-how .

Covering these requirements in a much more appropriate way requires the definition of research axes. These should focus on three main objectives:

- **Conceptual**:

  – How will it be possible to improve informal or semi formal approaches by developing dedicated mechanisms based on standardized, pre formatted and if possible semi-formal concepts and tools for enterprise modeling?

  – How is it possible to use formal approaches gaining advantages with regard to the current VVQC situation without the trouble and limitations of use induced by these approaches?

- **Technical**:

  – What kind of toolbox can emerge from a consensus between academic and industrial partners?

  – How should this toolbox be used, adapted and made available for the actors involved in a project, whatever their role, the project objectives or the enterprise type?

- **Usage**: training and achieving VVQC tools, tasks and methods

  – How to train future actors (engineers, researchers, technical staff, and also decision makers) to consider verification, validation and if possible qualification and certification goals as usual and relevant for their work, as is done in other domains.

  – When and how to formalize the VVQC processes and VVQC maturity level of an enterprise, as is now proposed in the System Engineering domain by the INCOSE [xxxx]

In this context, the Working Group “Verification, Validation and Accreditation of Enterprise Models” from Technical Committee TC5.3 and “Enterprise Networking and Integration” from the IFAC board aim to promote and to improve VVQC tools, concepts and approaches:

- **Conceptual objective**: the aim of this sub-group is to adapt and to formalize a set of concepts coming from other scientific domains and applied for a long time [32,33,34,35]. This implies defining the formal foundations of a set of relevant and important modeling languages such as UEML in its future version 2.0 [10]. This also requires defining and promoting a set of mechanisms and tools for supporting VVQC tasks when using this new modeling language. This has to be done using a mixture of semi-formal and formal approaches.
- **Technical objective**: a complete response to the conceptual aspect needs to be defined. However the sub group intends to produce an overview of existing tools which can be employed all along a project taking into account the project objectives, constraints and actor's profile (competencies, role).
- **VVQC usage**: this sub-group intends to define a guide for VVQC practitioners in industry working on information system building projects. This guide will have to be generalized in the future for other kinds of projects.

More precisely, this sub-group wants to promote several research axes which focus on merging formal and non formal tools and techniques. This requires:

- Developing property concepts for expressing functional and non functional requirements. A property may be proved on the model and then used to detect modeling errors, mistakes, lack of coherence, of relevance or dysfunctions. This involves being able to describe what a property is and how it is possible to prove it on the model as proposed by [95,96].
- To formalize and to integrate formal rewriting mechanisms with dedicated ontologies including analysis mechanisms such as Deductive Enterprise Model (DEM) proposed in [96].
- To define and to formalize partition mechanisms for checking the coherence between models coming from different points of view, or coming for the decomposition of another one. Considering for example a behavioral decomposition, the behavior of a model from the higher level must be equivalent to the behavior of the set of models coming from the refined level.
- To define and formalize rewriting mechanisms enabling the checking of the coherence of the resulting models after a horizontal refinement whatever the original and target modeling languages used during this rewriting process i.e. translation rules which can be formalized between two languages.
- To create new modeling languages or to extend the formalization of existing ones and of modeling frames integrating formal proof tools and concepts. Among other things, this will improve the model checking possibilities without any translation in other formalisms leading to a loss of sense or waste of time: the modeling language becomes able directly to support formal analysis techniques.
- Taking into account recent research works focusing on the unification of concepts required for enterprise modeling, to propose a common set of rules or to formalize existing behavioral rules so as be able to execute a model and measure results staying independent from any interpretation. In particular, the Multi Agent Systems paradigm must be introduced in this domain to define new simulation mechanisms. In particular, human resources behavior must be modeled in a more precise manner to enable unambiguous simulation of the global behavior of a complex socio technical system i.e. the enterprise system.
- To propose formalization languages and rules improving standards in order to be more formally specified and more suited to formal proof techniques.
- To define, formalize and standardize the use of knowledge reference models and knowledge representation approaches. This particularly concerns improving the ontology concept which is now one of the numerous areas of interest in research

5. Conclusion

This aim of this paper is to identify some areas of research that need to be taken into account for the next decade. Some of them have already been explored lasting recent years. However, it is now necessary to share and discuss these works and to evolve much more towards a common viewpoint of VVQC in enterprise projects.

The authors would firstly like simply to initiate this discussion and secondly to highlight the work on formal techniques, models, languages and tools that has been done and their application in industry to other kinds of complex system engineering approaches. The goal is not to replace existing VVQC approaches when they are already in use, but rather to merge formal and non formal approaches in a common framework and toolbox.

6. Bibliography

[1] A.Molina, H.Panetto, D.Chen, F.Vernadat, L.Whitman, Enterprise Integration and networking: Milestone Report, TC 5.3 Enterprise Integration and Networking, ICEIMT2004 proceedings, International Conference on Enterprise Integration and Modeling Technology,

Canada, 2004

[2] R.Cabral, G.Doumeingts, L.Man-Sze, K.Popplewell, Enterprise Interoperability: Research Roadmap, Working document, Version 2.0, 2006

[3] K.Tae-Young, L.Sunjae, K.Kwangsoo, K.Cheol-Han, A modelling framework for agile and interoperable virtual enterprises, Journal Computers in Industry, 2006

[4] L. Ferrarini, C. Veber, A. Lüder, J. Peschke, A. Kalogeras, J. Gialelis, J. Rode, D. Wünsch, V. Chapurlat, Control Architecture for Reconfigurable Manufacturing Systems: the PABADIS'PROMISE approach, ETFA 2006

[5] F.B.Vernadat, Enterprise Modelling and Integration: Principles and Applications, Chapmann & Hall, 1996

[6] INCOSE, System Engineering (SE) Handbook Working Group, System Engineering Handbook, A « How To » Guide For All Engineers, 2004

[7] GERAM: Generalised Enterprise Reference Architecture and Methodology Version 1.6.1, IFIP–IFAC Task Force on Architectures for Enterprise Integration, 1999

[8] J-L.Le Moigne, La modélisation des systèmes complexes, Paris, Bordas, Dunot, 1990 [in French]

[9] Fox, M.S., Gruninger, M. Enterprise Modelling, AI Magazine, AAAI Press, Fall 1998, pp. 109-121

[10] Deliverable D3.1: Requirements analysis: initial core constructs and architecture, Unified Enterprise Modeling Language UEML Thematic Network - IST-2001-34229 (see www.ueml.org), 2003

[11] ISO 8402: Quality management and quality assurance: Vocabulary, Second edition 1994-04-01, International Standard Organization, 1994

[12] O.Balci, Verification, Validation and Accreditation, Proceedings of the 1998 Winter Simulation Conference, D.J. Medeiros, E.F. Watson, J.S. Carson and M.S. Manivannan, eds.

[13] NASA, VV&A Recommended Practices Guide: Glossary, 2001

[14] D.Chen, F.Vernadat, Enterprise interoperability: a standardization view, Handbook on architectures of information systems, Springer, 2003

[15] R.A.Stegwee, B.D.Rukanova, Identification of different types of standards for domain specific Interoperability, Standard Making: A Critical Research Frontier for Information Systems, MISQ Special Issue Workshop, 2004

[16] EICTA Interoperability white paper, 2004 (see www.etsi.org/sos_interoperability/Background_papers/EICTA_white_paper_on_interoperability.pd)

[17] Y.Kalfoglou, M.Schorlemmer, Formal Support for Representing and Automating Semantic Interoperability, 1st European Semantic Web Symposium (ESWS'04), pages 45-61, Heraklion, 2004

[18] Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries, Institute of Electrical and Electronics Engineers, New York, 1990

[19] J.Bézivin, O.Gerbé, Towards a Precise Definition of the OMG/MDA Framework, ASE'01, November 2001.

[20] MDA, Model Driven Architecture (MDA), Architecture Board ORMSC, Joaquin Miller and Jishnu Mukerji Eds., 2001

[21] D.Chen, F.Vernadat, Standards on enterprise integration and engineering – A state of the art, In International Journal of Computer Integrated Manufacturing (IJCIM), Volume 17, n°3, April-May 2004, pp.235-253

[22] P.Bernus, Enterprise models for enterprise architecture and ISO9000:2000, Annual Review in Control, 2003, n°27, p211-220, Pergamon Press

[23] F.Yergeau, T.Bray, J.Paoli, C.M.Sperberg-McQueen, E.Maler, Extensible Markup Language (XML) 1.0 (Third Edition), W3C Recommendation, 2004 (see http://www.w3.org/TR/2004/REC-xml-20040204/)

[24] Y.F.Hwang, D.C.Rine, Verification framework and algorithms for integrating information distribution systems, Journal Information and Software Technologies, Elsevier, 2006

[25] K.Benghazi Akhlaki, M.I.Capel Tuñon, J.A.Holgado Terriza,L.E Mendoza Morales, Formal Specification of Real-Time Systems by Transformation of UML-RT Design Models, Procedding of Modelling, Simulation and Verification of Enterprise Information Systems MSVVEIS, Satellite event of ICEIS, International Conference on Enterprise Information systems, Paphos, Chypre, May 23 to 26, 2006

[26] Casualty Actuarial Society (CAS), Overview on Enterprise Risk Management, ERM Comittee, 2003

[27] W.M.P. Van der Aalst, Workflow Verification: Finding Control-Flow Errors Using Petri-Nets-Based Techniques, Business Process Management: Models, techniques and Empirical Studies, W. Van der Aalst, J. Desel, A. Oberweis (Eds.), Springer, 2000

[28] ARCHWARE, European Architecting Evolvable Software project, IST 2001-3236, 2004

(see hhtp://www.archware.org/)

[29] Leymonerie F., Blanc Dit Jolicoeur L., Cîmpan S., Braesch C., Oquendo F. Towards a business process formalization based on an architecture centered approach, ICEIS 2004, Portugal, 2004

[30] D.E.O'Leary, Enterprise Resource Planning Systems: Systems, Life cycle, Electronic commerce, and risk, Cambridge University press, UK, 2000

[31] M.Koubarakis, D.Plexousakis, A formal framework for business process modelling and design, Information System Journal, n°27, pp 299-319, Pergamon Press, 2002

[32] G.Love, G.Back, Model Verification and Validation for Rapidly Developed Simulation Models: Balancing Cost and Theory, white paper of the Project Performance Corporation (see http://www.ppc.com/), 2000

[33] H.S.Jagdev, J.Browne, P.Jordan, Verification and validation issues in manufacturing models, Computers in Industry n°25, pages 331-353, 1995

[34] RPG, V&V Techniques, RPG reference Document (see http://vva.dmso.mil/), DMSO (Defence Modelling and Simulation Office), 2001

[35] O.Balci, W.Ornwsby, Expanding our horizons in verification, validation and accreditation research and practice, 2002 Winter Simulation Conference, E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes Eds. 2002

[36] ISA Enterprise - Control system Integration, Instrument Society of America, Part 1, ISA-ds95.01, Draft 14, 1999

[37] C4ISR Architecture Framework, Version 2.0, December 1997

[38] D.Chen, B.Vallespir, G.Doumeingts, Designing Manufacturing Systems: Contribution to the development of an Enterprise Engineering Methodology (EEM) within the frame of GERAM, IFAC'2002 World Congress, Barcelona, Spain, 2002

[39] J.Sekkerman, Extended Enterprise Architecture Maturity Model (E2AMM), 2003

[40] EN ISO/DIS 19440, Enterprise integration: Constructs of enterprise modelling, Draft version, Genève, 2004

[41] J.Schekkerman, Enterprise Architecture Validation, Achieving Business-Aligned and Validated Enterprise Architectures Institute For Enterprise Architecture Developments Report, 2003

[42] D.Kelton, D.A.Sadowski, R.P.Sadowski, Simulation with Arena, McGraw-Hill, 2001

[43] E.J.Derrick, O.Balci, A visual simulation support environment based on the Domino conceptual framework, Journal of Systems and Software, vol 31, Issue2, December 1995

[44] R.Ahmed, S.Robinson, Simulation in Business and Industry: How Simulation Context Can Affect Simulation Practice. Spring Simulation MultiConference, Business and Industry Symposium, Norfolk, USA, 25-29 March 2007

[45] A.Greasley, Effective uses of business process simulation, Proceedings of the 2000 Winter Simulation Conference, J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick eds.

[46] S.Robinson, Successful Simulation: a Practical Approach to Simulation Projects. McGraw-Hill, Maidenhead, England, 1994

[47] NASA Formal Methods Specification and Analysis Guidebook for the Verification of Software and Computer Systems, Volume II: A Practitioner's Companion (see http://eis.jpl.nasa.gov/quality/Formal_Methods/document/NASA gb2.pdf)

[48] A.Van Lamsweerde, Formal Specification: a Roadmap, The Future of Software Engineering, A. Finkelstein (ed.), ACM Press, 2002

[49] D.Jackson, Lightweight Formal Methods, Proceedings of International Symposium of Formal Methods Europe, Berlin, Germany, March 12-16., 2001

[50] Bérard B., Bidoit M., Finkel A., Laroussinie F., Petit A., Petrucci L., Schnoebelen Ph. McKenzie P. Systems and Software verification: model checking techniques and tools, Springer (2001)

[51] M.Petit, Formal requirements engineering of manufacturing systems: a multi-formalism and component-based approach, Thèse de doctorat de l'Université de Namur, Belgique, Octobre 1999

[52] V. Chapurlat, B. Kamsu-Foguem, F. Prunet, Enterprise model verification and validation: an approach, Annual Review in Control, Volume 27, Issue 2, pages 185-197, 2003

[53] J.M.Spivey, The Z notation: a reference manual (2nd edition), Prentice Hall International, 1992

[54] J-R.Abrial, The B-Book: Assigning programs to meaning, Cambridge University Press, 1996

[55] Vienna Development Method: Specification Language (VDM-SL), Part1: Base language, Internat. Standard ISO/IEC 13817-1, 1996

[56] G.Smith, The Object-Z Specification Language. Advances in Formal Methods. Kluwer Academic Publishers, 2000 (ISBN 0-7923-8684-1)

[57] J.Hoenicke1, P.Maier, Model-Checking of Specifications Integrating Processes, Data and Time, AVACS – Automatic Verification and Analysis of Complex Systems, Report n°5, Board of SFB/TR 14 Eds.

[58] G.Morel, J-F.Petin, P.Lamboley, Formal specification for manufacturing systems automation, 10th IFAC INCOM Symposium, Sept.20-22, 2001, Vienna, Austria

[59] C.Schlenoff, M.Gruninger, F.Tissot, J.Valois, J.Lubell, J.Lee, The Process Specification Language (PSL): Overview and Version 1.0 Specification, NISTIR 6459, National Institute of Standards and Technology, Gaithersburg, MD, 2000, (see http://www.mel.nist.gov/psl/pubs.html)

[60] J.Hilger J., J-M.Proth, Production Management Language, IFAC'90, Tallinn, Russia, 1990

[61] E.Dubois, P.Dubois, Albert II: une approche à la conception de cahiers de charges pour des systèmes réactifs et coopératifs, Techniques et Sciences Informatique (TSI), vol. 17, no 2, Février 1998

[62] M.Petit, E. Dubois, Defining an ontology for the formal requirements engineering of manufacturing systems, in Enterprise engineering and Integration: Building international consensus (K.Kosanke and J.Nell Eds.), Springer Verlag, Berlin, 1997

[63] E.Dubois, M. Petit, Using a formal declarative language for specifying requirements modelled in CIMOSA in Integrated Manufacturing Systems Engineering, P.Ladet and F.Vernadat Eds., Chapman & Hall, London, 1994

[64] E.Canuto, F.Donati, M.Vallauri, Manufacturing algebra: a new mathematical tool for discrete-event modelling of manufacturing systems, Systems: theory and practice, Advances in computing sciences, R.Albrecht editor, Springer Wien New-York, pp 269 to 312, 1998

[65] F.Donati, E.Canuto, M.Vallauri, Advances in manufacturing algebra: discrete-event dynamic models of production processes, IFAC Manufacturing System: Modelling, Management and Control, Vienna, Austria, 1997

[66] R.Alur, T.A. Henzinger, Logics and Models of Real Time: A Survey, Real-Time: Theory in Practice, REX Workshop, LNCS 600, pp. 74-106, 1991

[67] E.Emerson, Temporal and modal logic, Handbook of Theoretical Computer Science, vol. B. MIT Press. Editeur: J. van Leeuwen ISBN 0262220393, pp. 955-1072

[68] C.A.R. HoareCommunicating Sequential Processes (CSP), Juin 2004, (see http://www.usingcsp.com/)

[69] S.Budkowski, P.Dembinski, M.Diaz, ISO Standardized description technique ESTELLE (see http://www-lor.int-evry.fr/idemcop and http://www-lor.int-evry.fr/edt/), 1997

[70] The SPIN Primer and Reference Manual, Addison Wesley, ISBN 0-321-22862-6 (see also aussi http://spinroot.com/spin/whatispin.html), 2005

[71] H.Garavel, Caesar Reference Manual, version 3.7 (see http://www.inrialpes.fr/vasy/Publications/Garavel -90-a.html), 1990

[72] ARCHWARE, European Architecting Evolvable Software project, IST 2001-3236, 2004 (see hhtp://www.archware.org/), 2004

[73] R.Milner, The Polyadic Pi-Calculus: a tutorial (This version is available at http://www.lfcs.inf.ed.ac.uk/reports/91/ECS-LFCS-91-180/), 1991 to 2003

[74] S.Easterbrook, An Introduction to Formal Modeling in Requirements Engineering, 10th Joint International Requirements Engineering Conference, in Essen, Germany, Septembre 2002

[75] Formal verification tools overview web site (see http://anna.fi.muni.cz/yahoda/), 2003

[76] M.Saaltink, The Z/EVES 2.0 User's Guide, TR-99-5493-06a (see http://www.ora.on.ca/z-eves/), 1999

[77] Stanford TEmporal Prover (voir le site http://www-step.stanford.edu/), version de 1996

[78] N.Bjorner, Z.Manna, H.Sipma, T.Uribe, Deductive Verification of Real-Time Systems Using STeP, Technical Report STAN-CS-TR-98-1616, Computer Science Department, Stanford University, 1998

[79] D.Diallo, Assistance à la validation par paraphrasage de spécifications formelles écrites en B, Thèse de Doctorat de l'Université de Nantes, 2000 [in French]

[80] K.L.McMilan, The SMV System for SMV Version 2.5.4: SMV Manual (see http://www-2.cs.cmu.edu/~modelcheck/smv.html), 2000

[81] S.Owre, N.Shankar, J.M.Rushby, D.W.J.Stringer-Calvert, PVS Language Reference, Version 2.4, Computer Sciences Laboratory, SRI International (see http://pvs.csl.sri.com/), 2001

[82] J-M.Faure, L-J.Lesage, Methods for safe control systems design and implementation, 10th IFAC Symposium on Information Control Problems in Manufacturing, INCOM'2001, Vienna (Austria), September 2001

[83] N.Völker, B.J.Krämer, Automated verification of function block-based industrial control systems, Science of Computer Programming, Volume 42, Issue 1, January 2002, pages 101-113

[84] J-M.Roussel, J-M.Faure, J-J.Lesage, A.Medina, An algebraic approach for dependable logic control systems design. International Journal of Production Research; vol. 42, n° 14,

pp: 2859-2876, July 2004

[86] The Coq Proof Assistant - Reference Manual, Version 8.0, Janvier 2005 (see http://coq.inria.fr/)

[87] W.McCune, OTTER 3.3 Reference Manual, ANL/MCS-Technical Memorandum n°263, Mathematics and Computer Science Division, 2003

[88] M.Kaufmann, J.S.Moore, ACL2 Reference Manual, Applicative Common Lisp, Department of Computer Sciences, University of Texas at Austin (see the last hypertext version available at http://www.cs.utexas.edu/users/moore/acl2/v3-0/ACL2-TUTORIAL.html), 2006

[89] The HOL System, Cambridge Research Center of SRI International (see http://hol.sourceforge.net/), 2005

[90] T.Nipkow, L.C.Paulson, M.Wenzel, Isabelle: A Proof Assistant for Higher-Order Logic, Springer-Verlag, 2005 (see also http://www.cl.cam.ac.uk/Research/HVG/Isabelle/)

[91] S.R.Rakitin, Software Verification and Validation for Practitioners and Managers. Artech House Publishers, 2001

[92] S.King, J.Hammond, R.Chapman, A.Pryor, Is proof more cost-effective than testing? IEEE Transactions on Software engineering, Vol 26, N°8, August 2000

[93] Méthode SAGACE: le systémographe, CEA, Version 1.0, 1999 [in French]

[94] Kamsu-Foguem B., Chapurlat V., Prunet F. Enterprise Model Verification: A graph-based approach, In Proceedings of CESA'2003, Computing Engineering in Systems Applications, Lille, France, 2003

[95] Accelera Formal Verification Technical Committee (FVTC), PSL Property Specification Language Reference Manual, Version 1.1 (see http://www.eda.org/vfv/), 2004

[96] M.S. Fox, M.Grüninger, On Ontologies And Enterprise Modelling, Proceedings of the International Conference on Enterprise Integration Modelling Technology97, Springer-Verlag, 1997