

Industrial Validator Whitepaper Thales

Nature:	Report		
Dissemination Level:	Public		
Distribution to Participants	Additional Distribution		
ALL INTERESTED companies represented	interested_all@interested-ip.eu www.interested-ip.eu		
DocID:	INTERESTED_whitepaper_Thales_final	Creation Date:	09/07/2010

Project	INTERESTED	Contract Number	214889
Author	P. Chaumette, M. Faugère, J-Y. Friant	Organisation	Thales

Project	INTERESTED	Contract Number	214889
Internal Reviewers	M. Barreteau	Organisation	Thales

1 Table of Contents

1	Table of Contents	2
2	Motivation for the Thales Validator	3
2.1	Overview	3
2.2	Market Requirements.....	3
2.3	The Need	4
2.4	Cost-Benefits Estimation.....	4
3	Technical implementation of Thales Validator	5
3.1	Current approach.....	5
3.1.1	Description.....	5
3.1.2	Weakness of the current approach	6
3.2	INTERESTED approach	6
3.2.1	Description.....	6
3.2.2	Model based design and validation approached for System Engineering.....	8
3.3	Metrics	10
3.3.1	Effort.....	10
3.3.2	Quality.....	11

2 Motivation for the Thales Validator

2.1 Overview

The objective of this document is to present the context in which Thales Rail Signaling will use and exploit the model based approach, involving tight intertwinement between design and validation steps based on strong language semantics for system engineering. The workflow defined and tested within the INTERESTED project will be used to measure the quantitative and qualitative improvement of the approach according railway system safety critical application development cycle.

Today, the Scade Suite is currently used for application software development; however the system and subsystem level specifications are based mainly on documentation produced during a classical approach with traditional documentation production.

The intent is to integrate system level modeling and validation capabilities, requirements management and documentation generation on top of software level modeling and validation capabilities based on a time-triggered approach, covering the whole development cycle with a model based approach.

This model based approach over the whole development cycle will allow - using modeler with well-defined system semantic and associated checkers - to perform system and subsystem level description and validation. The goal is to take advantage of this to enhance the documentation production process in terms of quality, design time and coherency, but also to move from a document to a model centric process.

The defined and evaluation metrics represents the benefits in terms of development time, model and documentation quality improvement of the generalized modeling and validation process based on strong and formal languages at system engineering level.

Even if the model based approach carries potential benefits as mentioned above, the development of safety critical application of the railway domain often comprises large implication of customer; this is especially true for development plans and documentation acceptance. This means that, in order to bring the most benefits, the customer as well must adhere to the model-based approach. Therefore, industrialization of tools used by the INTERESTED project is mandatory prior to industrial adoption of the workflow defined by the project.

2.2 Market Requirements

Today, the system engineering activities are mainly covered manually. Every document and especially schematics made using MS Office™ or MS Visio™, uses, at best, conventions that are specific to each designer. As the drawing tools are not backed up by a model, semantic checks or representation rules cannot be enforced automatically (and therefore reliably). Any document produced need to pass through the corresponding stages:

- Document design
- Technical review
- Safety review
- Customer review
- Authority review

Each of this stage is conducted by different teams in different organizations. Also the document acceptance is a key factor in the ability to deploy the system in operation. The productivity of this process impacts greatly the project deadlines.

As the documents are used for safety certification, they require a especially high level of quality: this quality includes complete consistency between documents describing different levels or aspects of the system; it also includes exhaustive traceability along the documentation and software components chains. Ease for determining impacts of a change, and to maintain consistency of all documentation with the product is a critical factor during the stages that precede the deployment of the system. Use of semantic free drawing tool has the advantage of enabling the schematics to be easily tailored to focus communication on specific point, but makes the maintenance of consistency of documentation a costly and lengthy task.

2.3 The Need

The main need is to move from a document centric to a model centric approach at system engineering stage. The designer should not have to define its own semantic and formalism to describe system behaviors, properties, structure, architecture and interfaces.

Document design and review become more efficient if the technical content obey to given semantic and specific design and naming convention rules, if the document is automatically generated from a common and unique model staying as basis for model and analysis.

The need is to:

- Avoid the overhead associated with the description format definition for system level documentation
- Reduce the freedom of interpretation by the reader
- Operate semantic check
- Operate coherency check at each description level
- Be able to identify differences between different versions to conduct impact analysis
- Generate part of the documentation from the system model
- Allocate requirements to architecture elements
- Decompose and link interfaces definition from top to bottom system/software level
- Decompose and link time domain related elements from top to bottom system/software level
- Share the tool will the different reviewers to allow navigation in models to support corresponding document review

TOPCASEDⁱ and similar tools are more and more deployed to support model based system and software engineering process, a homogeneous development environment with seamless transition between design layers from system engineering level down to software level, and adapted and customized for formal time driven design and validation semantics becomes an unavoidable need.

Model based system and software design, process, formal validation and associated shall:

- facilitate communication between the different design and validation actors and teams,
- ensure consistent communication support through the use of strong formalisms able to capture functional and technical information,
- help for requirements validation,
- ensure consistency among several input
- allow model simulation and formal validation
- allow full document generation (based on the common repository which is the model)
- allow correct software synthesis according models of computation and target platform features

2.4 Cost-Benefits Estimation

The metrics used to quantify the benefits of the integrated SysML design and Scade based validation tool chain, will be firstly a quantitative approximation based on a effort on man-month necessary to provide a System Engineering model –based design. This modeling activity will integrate model reworking due to modeling language particular semantics as well as formal validation steps with the Scade tool Suite.

A second metric will be more qualitative, based on number of remarks produced by the reviewers after the different document reviews.

3 Technical implementation of Thales Validator

3.1 Current approach

3.1.1 Description

Currently, the document production for system design is structured with the corresponding layers:

System level

- Requirements specification
- Architecture and interfaces specification

Subsystem level

- Requirements specification
- Architecture and interfaces specification

Application software level / subsystem

- Requirements specification
- Architecture and interface specification
- Component requirements and architecture specification
- Component design specification

The actual system engineering production is using conventional document design approach based on Word and Visio.

The software engineering development is produced using Scade Suite, and thus is model centric, rather than document centric as figured out Figure 1.

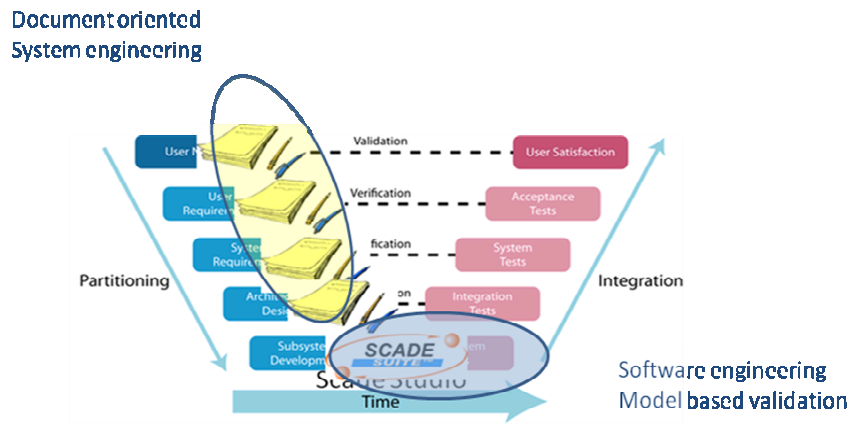


Figure 1: Actual document centric process

Model based design for application software development is already in place with the Scade suite tools set. Considering the design phase, the following data are showing the contribution level of model based approach during design phase.

Design phase	Model based design contribution to documentation generation with automatic generation.	Automated coherency check and design rule verification
Software requirements specification	10%	80%
Software architecture	20%	80%
Software Component Requirements and Architecture Specification	50%	60%
Component design specification	100%	90%

The software requirements specification phase uses model data to specify software interfaces at application level. Sharing models between software parts allows checking consistency of interface, but without dynamic aspect of the interface behavior. The contribution is low (10%) because the static description of interface is a minor (even if essential) part of software requirement (mainly free text at this level). Static description of interfaces in a data flow design (implied by SCADE), leads to high level of coherency check (80% of the 10% modeled).

For the software architecture level, the number of exchanged data flows increases, leading to a higher contribution.

For software component requirement and architecture phase, a partial model is prototyped. Therefore, a much higher contribution of model-based approach is already obtained. Nevertheless, the automated check level is lower due to the textual description of treatment which represents a large amount of work, but which is not using a model representation, and cannot be checked. Special rule checker tools have been developed to push up the automated checking of design rules.

For component design phase, the 100% figure corresponds to most applicative software components, which are entirely developed using SCADE.

3.1.2 Weakness of the current approach

The current approach presents many drawbacks:

The formalism is highly dependent of the producer. This dependency can be damped with guidelines and generic document templates but not to a level allowing controlled production.

The current approach has many weaknesses:

- Lack of formalism generates freedom of interpretation and therefore divergence of understanding between redactors and readers: a formal, semi-formal or at least standardized way of describing requirements and architecture would reduce freedom of interpretation.
- Plain office (word processor, drawing tool) with human language redaction prevents implementing tooled checks (semantic checks especially): consistency relies on redaction quality and peer review mainly.
- The same lack of formalism that prevents tooled consistency check at a given level also prevents tooled checks between different level, for instance between interface requirements and interface implementation.
- Between different versions, differences are limited to documents differences: this means that substance and form are mixed (and produce the same kind of difference). Impact analysis may often be difficult to conduct.
- The source of information being the final document, repetition of information is difficult to avoid and costly to maintain (this could be avoided by generating part of the documentation from the system model)
- Traceability implies allocation of requirements to architectural elements. As DOORS™ is used to hold traceability information at system/subsystem level, this in turn implies capturing architectural elements in DOORS™ database. This leads to double capture of architecture (architecture documentation and traceability database), which means additional workload and risk of inconsistency.
- Interfaces exist in multiple forms (top level documents, lower level documents, code, even traceability database), with different levels of detail, but not tooled mechanism ties this multiple descriptions. This leads to additional workload and risks of inconsistency.
- Timing requirements and constraints are spread in documents of various level and purpose, and there are no reliable ways to collect this information.
- Documentation review is made difficult by the absence of navigation support, especially across different documents (opposite to what happen when navigating within a model).

3.2 INTERESTED approach

3.2.1 Description

To collect realistic cost-benefits, the INTERESTED workflow have been tested on a representative Thales application, already design and validated in Scade Suite. The chosen application is a subset of the Thales Rail Signalling System, the Communication-Based Train Control (CBTC), soon to be installed in the Paris line 13 metropolitan railways, and

presented high safety critical aspects. The aim of this system is to control train speed and signalisation in order to maximise vehicle frequencies satisfying safety and security rules. A special focus on the versioning control subsystem has been studied.

Based on Visio documentation, System Engineering activities have been reworked in a model-based approach (in SysML) coupled with a formal validation and refinement process based on synchronous paradigms under the Scade Suite tool.

To evaluate the benefit of such a tool interaction in a model driven context, a specific process has been set-up as presented Figure 2 using the Artisan Studio-SCADE Suite gateways.

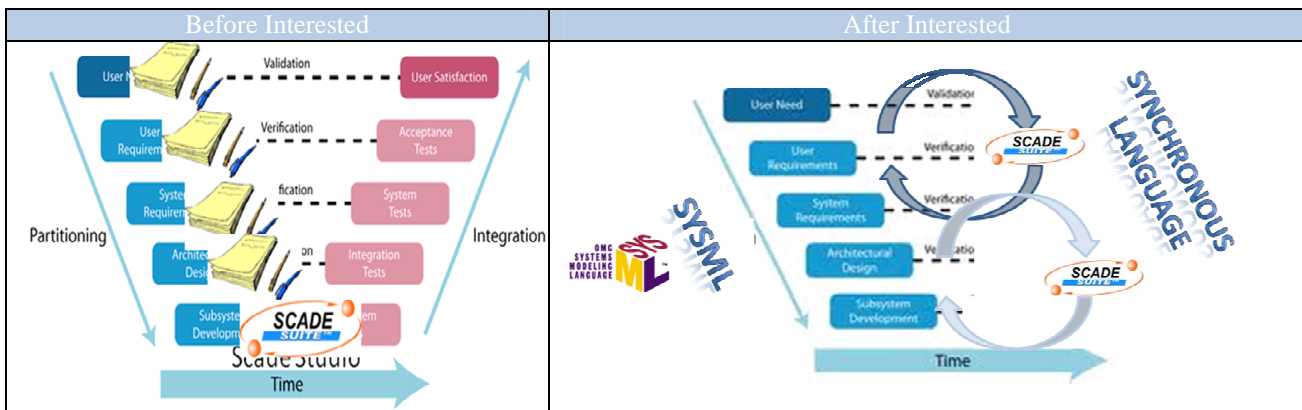


Figure 2: System engineering process with integrated Artisan-Scade gateways

The process is incremental and iterative. System design will be made in SysML world, using SysML formalism, allowing the expression of following structural system features:

- System decomposition in subsystems,
- Interfaces specifications,
- Data type definition.

Some subsystem behaviours will be refined and validated in Scade, leading to the interfaces (specified in SysML) validation for a given abstraction layer. Refinement can go then further in SysML, leading to more detailed system decomposition and interface specifications that will again be validated in Scade.

In practice, this round trip means that

1. In a first step, a first model will be designed with Artisan Studio, leading to the definition of the system, the subsystems, the interfaces and data type and structures; some model parts will then be imported into SCADA Suite;
2. In a second step, the model will be completed / checked / simulated from a functional and logical point of views in Scade; syntactic and semantic automatic checks will be made;
3. In a third step, the refined model issued from Scade will be re-introduced into Artisan Studio to be refined, completed / corrected / modified and re-imported back to SCADA for further modifications / completions until converging towards stable interfaces.

The INTERESTED process will differ from the Thales Rail Signalling traditional one, in the sense that:

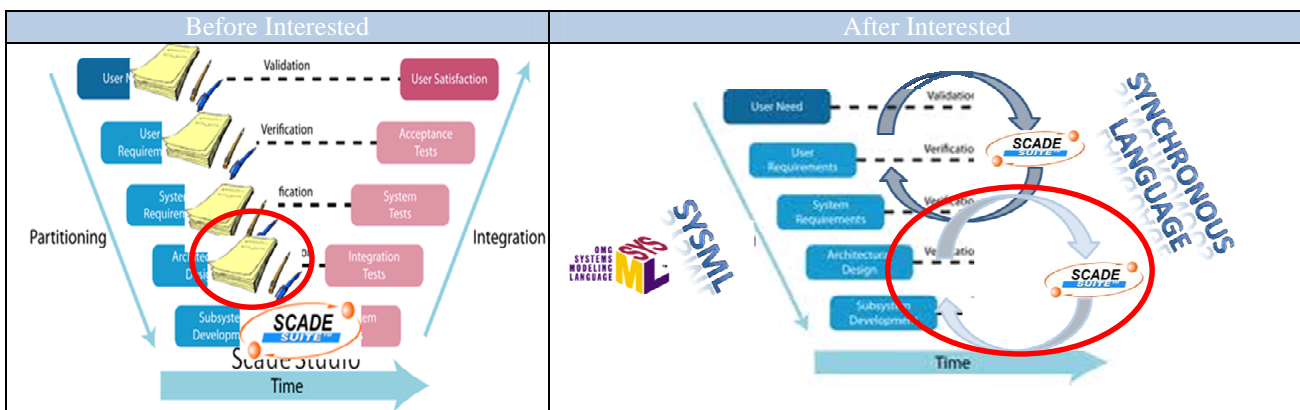
- A common model language with well defined semantic will be used (SysML) (two languages SysML and Scade/Lustre in our case but can be seen as a same language at system engineering level)
- A design language providing static and dynamic and graphical design capabilities
- A design language providing well defined granularity abstractions
- A generic design language supported by many tool vendors
- A centralized data base concentrating all the information on a unique place
- A centralized data base for generating consistent documentation and code
- Syntactical correctness and validation process at system engineering level validating types and interface design and port connection mismatch detection (type, in/out)
- Management and integration of analysis and simulation results

- Tight traceability and consistency management between system and software engineering levels including behaviours model validation based on synchronous paradigm
- Separation of concern (control flow/data flow) based on strong semantics

In this evaluation, the extra cost relying on design and validation language and associated tool design mastering, multi layer model based approach design and refinement process has been abstracted away.

3.2.2 Model based design and validation approached for System Engineering

The aim is to evaluate the generalization of a model based approach based on a time triggered semantic at system engineering level with a tight coupling between system and software engineering layers. The syntactical and semantical validation is provided by the use of a formal language and validation tool (based on synchronous paradigms) coupled to the SysML language.



The red circle on the left represents the focus described in 3.2.2.1; the one on the right represents the focus described in 3.2.2.2.

3.2.2.1 Evaluation of a model based approach for System Engineering

Mastering already the software engineering process with formal design and validation under Scade Suite, the goal is to provide a top level design and validation environment on top of this software engineering layer, composed of a SysML design tool (Artisan Studio) coupled to a formal environment design and validation environment (Scade Suite).

We expected following benefits in term of design quality and document production improvements, provided by:

- The use of a unified, complete and standard modeling language: a unified semantic provides to the designer a frame for designing expression without the need to define him -self the semantic fitting is needs.
- The use of a common language enhanced understanding of documentation between individuals, teams, and companies, without ambiguities
- The use of a complete language supporting development cycle from requirement specifications until code generation, supporting
 - Structural, behavioral specification, functional and non functional capabilities
 - Graphical and textual capabilities
- The use of a common referee to exchange information between
 - members and activities
 - abstraction levels: transfer of interface definition at system level to application software development environment for subsequent definition and refinement
- Unified approach with continuous learning phase implying productivity enhancement in understanding the documents

- Reduced amount of remarks related to quality, coherency and understanding of system specification

Extra cost have been identified at the beginning of this evolution phase towards model centric approach, in terms of effort and money due to SysML modeling language, activities and tool mastering, as well as in a modeling tool investments.

Recommendations for tool providers

- Having a common unified repository is a good point, but this is not enough. A tooling process is needed, supporting specific domain information and abstraction layers
- Modeling Guidelines are required
- Domain and process specific abstraction levels and modeling granularity have to be set-up

3.2.2.2 Evaluation of a coupled formal and non formal design and validation process for System Engineering

SysML, combined with UML is a unified, common and complete modeling language, and currently the most popular modeling language used in industry. But this language is not formal at the moment, it support many variation points that have to be fixed to allow simulation capabilities. Coherence checks are usually not supported by modeling tools.

At the opposite, Lustre (Scade Language) supported by Scade Suite in a formal language, allowing validation and simulation capabilities without being ambiguous. Besides being based on synchronous paradigm with unique assignment principles, the language and tool environment allow correct and validated design, simulation, and code generation is proceeded by a certified code generator.

Scade Suite has been used as semantical checkers for SysML. This validation capability will be only valid for common SysML and Lustre features, i.e. structural, interface, interface types and port connections validation features.

We expected following benefits in term of design quality and document production improvements, provided by:

- Reduced errors during design phase with semantic and coherency check support for designer
- Incremental functional design and validation
- Interface consistency checks

Recommendations for tool providers

Beside the fact that there were gateway maturity and coverage limitations, it seems important to make recommendations on gateway functionality improvement to better match Thales and more generally industrial needs.

- Better run-trip process definition and integration in a development process
Different run-trip scenarios can be imagined at different development process levels (i.e. focus on requirements, systems decomposition or subsystem analysis, and code generation). These run-trip processes shall be specified, formalized, allowing the designer to better integrate one of them in its own development process with a run-trip activity. During this evaluation, gateways have been more designed in a refinement process than in a run-trip process: models are imported following different semantics, and will be completed and refined without the possibility to come back to the initial model syntax and semantic in an integrated way.
- Finer granularity of the Artisan2Scade gateway: it shall be possible to import all, or a selected subset of "Scade" stereotyped SysML blocks into Scade.
- Check and error shall be detected and raised up before transformations / imports.

- SysML and Scade semantic alignment shall be provided by the Scade profile. Scade and SysML do not have the same semantic: Scade is based on a strong semantic relying on synchronous language properties and hypothesis like the unique assignation property. UML/SysML language relies on much more looser and larger semantics, where message broadcasts and multiple receptions are possible for example. It is necessary to restrict (in the profile) the SysML semantics to the Scade one by the way of patterns and modelling limitation.
- Full synchronisation between blocks, interfaces and data types shall be provided (deletion, addition, modification), parts, without having to re-import the model in another repository or without having to close one of the tool.

What has been appreciated:

- Navigation and object localization between Scade Suite and Artisan Studio;
- Modification propagation between Artisan and Scade (Addition / deletion of subcomponents, ports, types; modification of (components, ports, types) names; modification of the packaging (for subcomponents and types).

3.3 Metrics

Thales uses two different metrics, a qualitative and a quantitative one, to estimate the cost/benefits in adopting the INTERESTED tool chain.

The first metric is an effort relying metrics, allowing to quantify how “easier” and “faster” the modelling and validation activities will be improved while moving from a document centric to model centric approach.

The second metrics is a qualitative one, used to give a feedback on the quality and correctness of the design and documentation.

3.3.1 Effort

The two major benefits on productivity rely on one hand on the use of the model based approach, with all the model based engineering benefits like:

- common language to design system and software concepts,
- structural and behavioral capabilities,
- specific diagram semantics,
- unique repository,
- well defined SysML/UML semantics

The second benefits rely on the formal Scade/Lustre semantics, allowing to validate syntactically and semantically a given model. At a first level, the Scade tool can be considered as the UML/SysML model checker, validating syntactically the SysML model (type mismatch, unused types, etc...)

Based on those two inputs the absolute gain in effort has been calculated.

Metrics	Gain (in time)
Total effort including design and reviews	25%
Diagram design effort	10%

The gain obtained in diagram design effort is not so important than the one obtained on model correction; this is especially due to the fact that Word and Visio provide a large freedom on design, that is not allowed in UML/SysML. Diagram semantic is well defined, and could not be modified.

3.3.2 Quality

SCADE provides a complete tool suite covering all system development lifecycle, from requirement traceability with Doors, system design, formal verification and simulation, qualified code generation and model test coverage capabilities. All these functionalities are adapted and optimized for time-triggered approach. Artisan-SCADE gateway will enhance SCADE with UML/SysML specific features and capabilities helping the designer to specify the system components and associated interfaces from the requirements, using specific activity and collaborative concepts.

Rational Rhapsody Designer for Systems Engineers provides for system engineers a collaborative development environment with simulation for early requirements, architecture and behavioural validation, helping communication of complex requirements and trade-off analysis of complex systems. As SCADE, it provides full lifecycle traceability and analysis from requirements to design, customisable automatic documentation capabilities, simulation that executes the model to help architecture validation; static model checking analysis helps improve design consistency.

The functional coverage is the same, but the theoretical foundations behind SCADE, are much stronger, especially concerning time for SCADE, than for UML/SysML. SCADE provides formal capabilities, allowing to reason without ambiguity on logical and temporal logic. Use the gateways, the two repositories can be seen as a unique one.

Metrics:	Gain (in time)
Remarks produced by reviewers	25%

Based on one hand on SysML semi formal language and large development cycle coverage, as Scade/Lustre strong formal language and validation capabilities, the models are correct, the associated code and documentation also, generating a global quality improvement of 25% on the first review. Errors remaining are conceptual ones, rather than syntactic ones.

ⁱ www.topcased.org/