	<b>Rapport de fin de projet DOMINO</b>	Date : 5/05/2009
		Réf.: ANR-FORM-080124-01-01
		Nombre de pages : 7

## Projet ANR-06-TLOG-011

### Rapport final du projet DOMINO

#### A. Identification

Acronyme du projet	DOMINO
Titre du projet	DOMaINes et prOcesus méthodologique
Coordinateur du projet (société/organisme)	Christian Percebois Institut de Recherche en Informatique de Toulouse
Période du projet (date début – date fin contractuelle)	mars 2007 à juin 2009
Site web du projet, le cas échéant	<a href="http://www.domino-rntl.org">http://www.domino-rntl.org</a>

Rédacteur de ce rapport	
Civilité, prénom, nom	M. Christian Percebois
Téléphone	05 61 55 74 23
Adresse électronique	percebois@irit.fr
Date de rédaction	5 mai 2009


Liste des partenaires à la fin du projet (société/organisme et responsable scientifique)	Airbus, Jean-Charles Dalbin CEA LIST, Hubert Dubois CNES, Erwann Poupart ENSIETA, Philippe Dhaussy INRIA, Benoît Baudry IRIT, Christian Percebois Soft-Maint, Laurent Sabatier
--	--

#### E. Mémoire scientifique

The DOMINO approach explicitly aims to increase the level of trust in software systems developed with a mode-driven approach. DOMINO proposes a set of concepts and techniques to domain experts so that they can define and establish their own model-based process to leverage models and model transformations and build trustable systems. We present two complementary aspects: the validation of models produced at the different steps of the process and the development of trustworthy components that completely or partially automate a development step. The concepts, methods and tools developed during our work have been applied to case studies in CNES and Airbus.

##### E.1 Expected benefits, related issues, state of the art

Model-Driven Engineering (MDE) improves the capitalization of design know-how, the reuse - on an abstract level - of development artefacts and the control of complexity by means of the unifying framework that the models create. However, the lack of robust tools for model manipulation and management, of well-defined domain specific languages and associated technologies, the issues related to separation of concerns in models and the lack of validation techniques of models and model transformations are critical barriers to a wide industrial adoption of MDE. Thus, current processes use conventional software

	<b>Rapport de fin de projet DOMINO</b>	Date : 5/05/2009
		Réf.: ANR-FORM-080124-01-01
		Nombre de pages : 7

engineering processes and tools and suffer from interpretation and implementation errors. In most cases, verification occurs after the development phase. As a consequence, verification, correction and maintenance require a large human effort, whereas MDE tends to automate critical development steps in order to increase quality by construction, thus decreasing the effort needed at the end of the development.

These activities should be reconsidered and more attention should be paid to the new possibilities in modelling and meta-modelling offered by MDE. Efforts should now be focused on the requirements modelling and on ensuring the continuity in the development process. In turn, the process can be implemented by means of model transformations and intermediate verifications.

It is in this context that DOMINO aims to increase the quality in software development by offering rigorous methods and associated tools to evaluate and improve trust in basic components. Below, we define trustworthiness as a set of guarantees for the component. Engineers can therefore make the component either totally or partially responsible for an activity in a model-based process. More specifically, our study involves an MDE development process. It approaches the questions of model quality improvement and the trust in the transformations of models that perform development stages automatically.

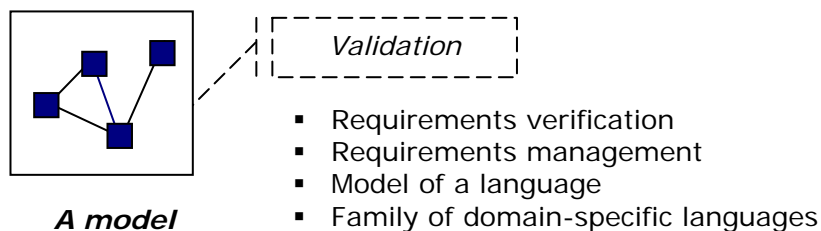
## E.2 Scientific and technical approach


Trust in a software system is supported both by the validation of the models built along the development cycle and the use of trustworthy MDE components. Modeling and checking techniques are specified for validation. We also propose a model for MDE components based on the integration of three characteristics: its specification, its implementation and the techniques of V&V.

Generally speaking, the notion of trust has many facets. It takes various forms depending on the level of abstraction considered and is strengthened either by construction, or by redundancy. For model transformation for instance, the construction refers to the application of techniques dedicated to the definition of transformations that can reduce the gap between requirements and implementation, thus reducing the risk of errors. Redundancy aims to keep track of requirements in relation with the implementation in order to prove and test the transformation.

### E.2.1 Validation of the models

This part of the study aims to ensure the relevance of the models produced during the development process. Validation comes down to seeking a certain form of completeness with respect to the system being modeled and thus to link the models to expected or perceived reality. The experiments on the two DOMINO case studies (CNES and Airbus) more specifically involved checking the requirements and the modelling of so-called domain-specific languages, as illustrated in the figure below.



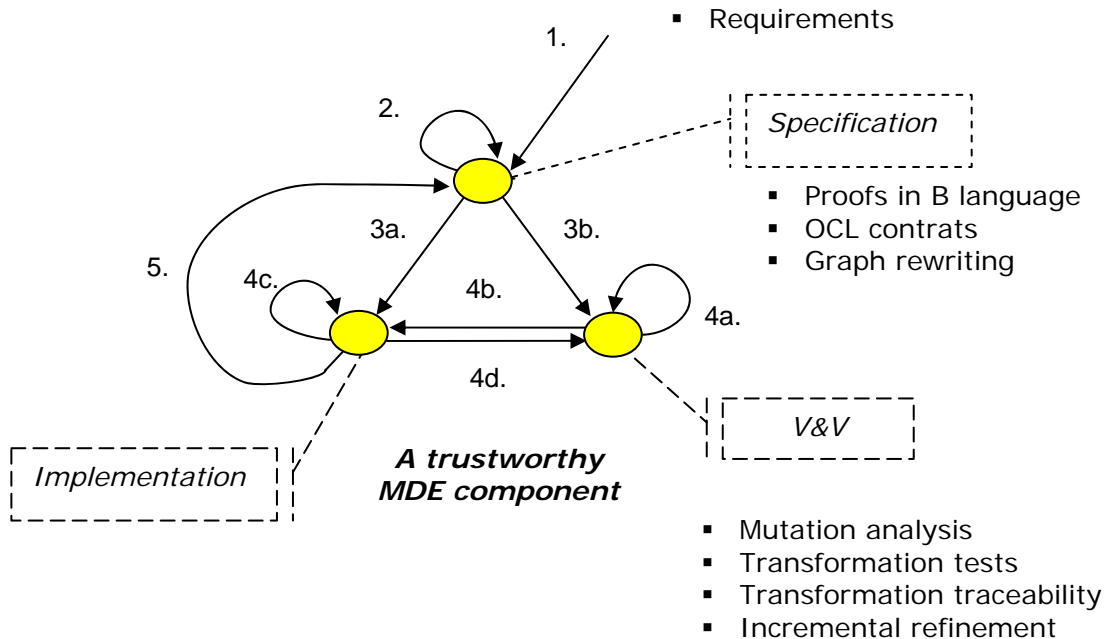
	<b>Rapport de fin de projet DOMINO</b>	Date : 5/05/2009
		Réf.: ANR-FORM-080124-01-01
		Nombre de pages : 7

- **Verification of requirements.** The requirements must be checked early in the development. To carry out such verifications, it is necessary, to express the requirements in terms of formal properties and, to formalize the context in which the model should be checked. A Context Description Language (CDL) was proposed in order to unify the description of the use-case contexts and of the properties. From this point, the concept of Proof Unit (PU) emerged. It refers to the model to be validated and to a CDL model that specifies both the requirements and the V&V context of the use-case. These units are translated within the *Observer-Based Prover* (OBP) tool and by the transformation of UML models into formal description that can be validated by formal V&V environments.
- **Requirements management.** The models must be validated with respect to the requirements. A requirements meta-model has been proposed to define the requirements and manage their traceability. The meta-model takes the form of a dedicated profile based on SysML and includes tooling for requirements management in the Papyrus environment. In addition to the annotation and the traceability links, the accent is placed on the coherence of the requirements model, the solution model, and the requirements V&V.
- **Model of language.** DOMINO contributes to the integration of Domain-Specific Languages (DSL) into MDE developments. We propose an intermediate modelling step to adapt design models before code generation, by establishing, in terms of models, a clear and unambiguous definition of the grammar of the target programming language. The integration of models based on the grammar of the languages enables software development on a precise definition of the links between the models and the programs. Two levels of assistance are proposed: one, which is original, in the BNF concerning the grammar of a language and another, which is more conventional, in the language model concerning the program models. It then becomes possible, at these two levels, to add domain specific properties.
- **Family of domain-specific languages.** Any given professional field usually has a family of DSL. Developers are supposed to speak the same single language. In practice however, each group of specialists rewrites the concept of that sector of activities in his own domain-specific language. In order to ensure the interoperability of the domain-specific components, we consider the formal semantics of each of the languages. Using results from category theory, we obtain semi-automatically, the definition of a language that can unify the family, as well as translators for the initial languages. The same theoretical results ensure that it is possible to transpose and to automatically prove a property from one DSL towards the unified language. This approach was validated for the operational procedure languages used in the space industry.

### E.2.2 Trust in MDE components


An important goal in DOMINO is to control the development of an MDE component that automates a step of the process in order to measure its level of trust. In this way, a component (represented by the triangle in the figure below), has three vertices: its specification, its implementation and its associated assets for V&V. DOMINO technologies aim to improve the specification and V&V protocols for MDE components and contribute to the overall consistency of the components' facets.

It can be noted that the triangular structure model can be used at each of the stages of the engineering process.



The following techniques, proposed by DOMINO for MDE components, were tested on the CNES and Airbus case studies. They apply distinct levels of formalization – from semi-formal, such as model transformation testing, to the formal level with the proof in B of the transformation and contracts. We then evoke more prospective studies concerning the rewriting of graphs and the refinement of models more akin to development proved by construction.

- **Proof in B.** We experimented the B language to formalize the meta-models and the transformation rules from SOLM to O2PL for the CNES case study. The formalism helps express the meta-models and the source and target models by data elements (sets, constants and variables) and by the predicates defining the properties of these elements (B invariants). Each transformation rule is modeled as an abstract B operation which defines a precondition and a substitution. The B precondition formalizes the condition for applying the transformation rule, while the substitution part formalizes the target elements. The transformation specified in B enables the use of the proof assistant to analyze and prove the transformation consistency with respect to the meta-models and transformation invariants.
- **OCL contracts.** These contracts are useful to test model transformations as they represent an executable version of the specification and enable errors to be detected at runtime. It is therefore possible to dynamically ensure that the component accepts the models that are processed and, in turn, produces correct models. With DOMINO, the contracts concern the syntax of the meta-models and source and target models of the transformation as well as its behavioral semantics. They can be written in OCL-Kermeta by using the aspects mechanism to dissociate the structural from the behavioral parts of the transformation, or in pOCL (*procedural Object Constraint Language*), an extension of OCL that supports simultaneous manipulation by several models and allows inter-model correspondences.
- **Mutation analysis.** Originally used for the qualification of a set of test models, we use mutation analysis to measure the level of trust of the transformation component. The technique is based on the creation of erroneous versions of the transformation to check how the test models behave. The erroneous versions, called mutants, are confronted with the contracts that implement an executable form of the specification

	<b>Rapport de fin de projet DOMINO</b>	Date : 5/05/2009
		Réf.: ANR-FORM-080124-01-01
		Nombre de pages : 7

embedded in the component. We adapt this technique for model transformation with new fault models that capture the errors specifically found in transformations. Such faults are related to the navigation and the filtering of the source and target models and on the creation of the target model.

- **Transformation tests.** Testing a model transformation consists in running the transformation with test data, and checking that the model that is produced is acceptable with respect to the transformation specifications (expressed in natural language, in terms of rules or in terms of contracts). In the DOMINO project, we focused on the automatic generation of test models, taking into account two essential issues in test data generation: these data must satisfy a large number of constraints coming from heterogeneous sources of knowledge – cover requirements, conform to the meta-model, satisfy pre conditions for the transformation – and these data are structurally complex - graphs of objects. The solution proposed in DOMINO is based on the expression of all constraints in a common formalism and on techniques for automatic constraint solving using SAT solvers.
- **Traceability of the transformations.** The eTraceTool platform of DOMINO captures traces of imperative model transformations. The transformation events are intercepted in a non intrusive way by means of aspects-oriented techniques and represented as a trace model which conforms to nested and arcs labeled traces meta-model. Then, traces model is isomorphic with the functional decomposition of the transformation. In a MDE way, if we tool all along a refinement chain with our traceability tool and repercussion transformation, we can obtain requirement traceability in exhibiting design choices on properties refined during the process. A model is generated for each execution of a traced transformation. The platform has been used in the refinement of properties defined in Context Description Language (CDL). A new transformation obtained by co-evolution thus allowed the automatic generation of the refined properties from the transformation of an abstract context and its CDL properties.

The following DOMINO techniques are part of the project's results, although they deserve deeper study before experiments.

- **Graph rewriting.** It is a unified approach for the category of attributed graphs depending on inductive types to define the structure of graphs and the associated attributes. In order to tend towards a program for operational model transformation, we consider two paths: proof-oriented implementation (Coq and other proof assistants) or implementation in a functional language (Caml, Ocaml and Haskell).
- **Incremental refinement.** We are working on definition of an incremental approach for the construction of complete, valid and deterministic behavioral specifications. The model would be constructed from a succession of models obtained through a series of transformations (addition or deletion of elements, reduction of the indeterminism, etc.) with associated verification procedures. We plan to study the applicability of this approach to UML behavioral models, in particular to state machines and sequence diagrams.

### E.3. Results

One of the DOMINO contributions for the industrial partners CNES and Airbus is the improvement of an MDE process integrating the activities of model transformation and validation.


DOMINO provides technologies that aim at building trust in automatic transformations. This should reduce human efforts reduce the amount of error-prone, tedious manual


activities of the development process. Moreover, they should reduce the amount of test and manual checks on MDE components while maintaining the necessary level of trust.

Expected MDE benefits are twofold: MDE provides the ability to build more automated design processes and know-how capitalization; it also provides an opportunity to capture well-formed (IEEE 1220: unambiguous, testable or measurable, etc.) and formal requirements. When requirements are modeled as formal properties, the subsequent design and validation phases are more effective, and trust may be provided by DOMINO formal technologies. Another MDE benefit is the ability to use, early in the requirement engineering process, high level concepts, easier to use for humans compared to those coming from lower level executable domain specific languages. DOMINO's DSL technologies are elements that may help to achieve this goal.


We validated some techniques for building trust in MDE components. The process iterates on the activities related to the three vertices. At each step, one can choose the technology that is the best suited to manage the current activity for the development of the component. Improving a summit with respect to the two others improves the overall consistency of a component. Thus, for instance, in the context of V&V by testing, fixing an error in the implementation leads to the creation of a new set of mutants. This forces to iterate on the generation of test models and the definition of contracts. Likewise, the improvement of the contracts involves testing the implementation again, since the contracts can be used as the oracle of any test case.

The tables below report the technical achievements for the CNES and Airbus case studies about the points mentioned above. The items in italics refer to academic studies related to case studies. The top part relates to the validation of models and below is related trustworthy components.

	CNES	<ul style="list-style-type: none"> <li>▪ A meta-model generator based on BNF rules applied to the meta-modeling of O2PL language</li> <li>▪ Proof of the <i>Guidance Navigation and Control (GNC)</i> property</li> <li>▪ <i>Definition of a language common to two dialects inspired from the Pluto standard</i></li> </ul>
	Airbus	<ul style="list-style-type: none"> <li>▪ A meta-model generator based on BNF rules applied to the meta-modeling of C language</li> <li>▪ Proof of the properties of the model of "COM/MON synchronization"</li> <li>▪ Management of the "COM/MON synchronization" requirements</li> </ul>

	CNES	<ul style="list-style-type: none"> <li>▪ Proof in B of the transformation of SOLM into O2PL</li> <li>▪ pOCL verification of the transformation Activity Diagram into O2PL</li> </ul>
	Airbus	<ul style="list-style-type: none"> <li>▪ OCL verification of the models at the input of the Airbus code generators</li> </ul>
	CNES and Airbus	<ul style="list-style-type: none"> <li>▪ <i>Co-evolution of business and properties models</i></li> </ul>

From the academic point of view, the study provided a better understanding and formalization of the notion of trust. The different propositions have been consolidated by the definition of a common framework of interaction between specification, implementation and verification of the component. As a multi-faceted concept, trust involves different formalisms which are complementary: requirements and traceability of the transformations, proof units and traceability, modeling formalism and the creation of a unifying language for a family of DSL.

	<p>Rapport de fin de projet <b>DOMINO</b></p>	Date : 5/05/2009
		Réf.: ANR-FORM-080124-01-01
		Nombre de pages : 7

#### E.4 Discussion and perspectives

- **Diagnosis and analysis.** The techniques developed by DOMINO are meant to design correctly and find defect in models or MDE components. However, if these techniques allow detecting anomalies early, they currently provide little assistance to find the source of the error and to fix it. Providing relevant and understandable feedback to the user, based on the analysis provided by DOMINO techniques, is an important perspective in order to make these techniques applicable in an industrial context. For example, while studying the applicability of OCL or proof units, the lack of efficient feedback has been identified as a major limitation by Airbus. Diagnosis information has to be provided as charts, text or logs, which are technological spaces very different from the modeling space. This shift between spaces is a major challenge for relevant diagnosis and efficient assistance to diagnosis.
- **Adaptable components.** The studies carried out during the DOMINO project also demonstrate the necessity to identify and quantify the trust that can be attributed to a MDE component, especially a model transformation that automates parts of software development. If this is put into practice, software engineers would therefore have access to libraries of MDE components and would choose the component(s) that best fit the process activities. On the other hand, it is necessary to take specific contexts into account when using a component as a part of a specific MDE process. This raises the challenge of the variability of components and the way they are used. This perspective is also about how to compose components in compliance with the global reference process. The use of such components can only be envisaged if the investment devoted to the requirement formalization actually leads to more trust and less tedious and error-prone development tasks. This will leave time for the unambiguous management of activities and interference-free interpretation and will lead to an efficient and robust model-based development.
- **Multi-domain collaborative development.** To ease the effective use of DOMINO technologies in an industrial context, it is necessary to take into account collaborative engineering with experts coming from different domains. Embedded systems such as space systems are often systems of systems that would benefit from multi-views capability for design, validation and also operation phases. However, multi-domain collaborative development of complex system models is not currently supported by model-based environments. Working on different but related models can be critical when considering dynamicity of models and domain-specific teams. In order to address complex inter-relationships and complex evolution cycles of the whole process, we need to address consistency checks between viewpoints at some specific synchronization periods of the development.