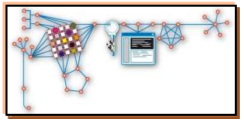
DE LA RECHERCHE À L'INDUSTRIE

DEALING WITH COMPLEX-SYSTEMS DESIGN:

THE MODEL-DRIVEN PARADIGM.



Huitièmes Journées Informatique de l'IN2P3-IRFU

Sébastien Gérard

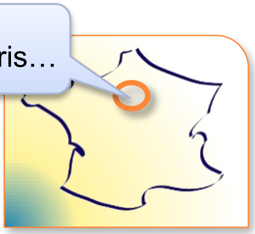
CEA LIST / LISE

sebastien.gerard@cea.fr

BEFORE STARTING…

WHERE AM I COMING FROM?

20 km south of Paris…



CEA LIST

- Part of the Department of Research and Technology of the French governmental agency for nuclear and alternatives energies (CEA)
- Focused on developing innovative technologies for smart and complex systems
- Staff ~ 750 persons.

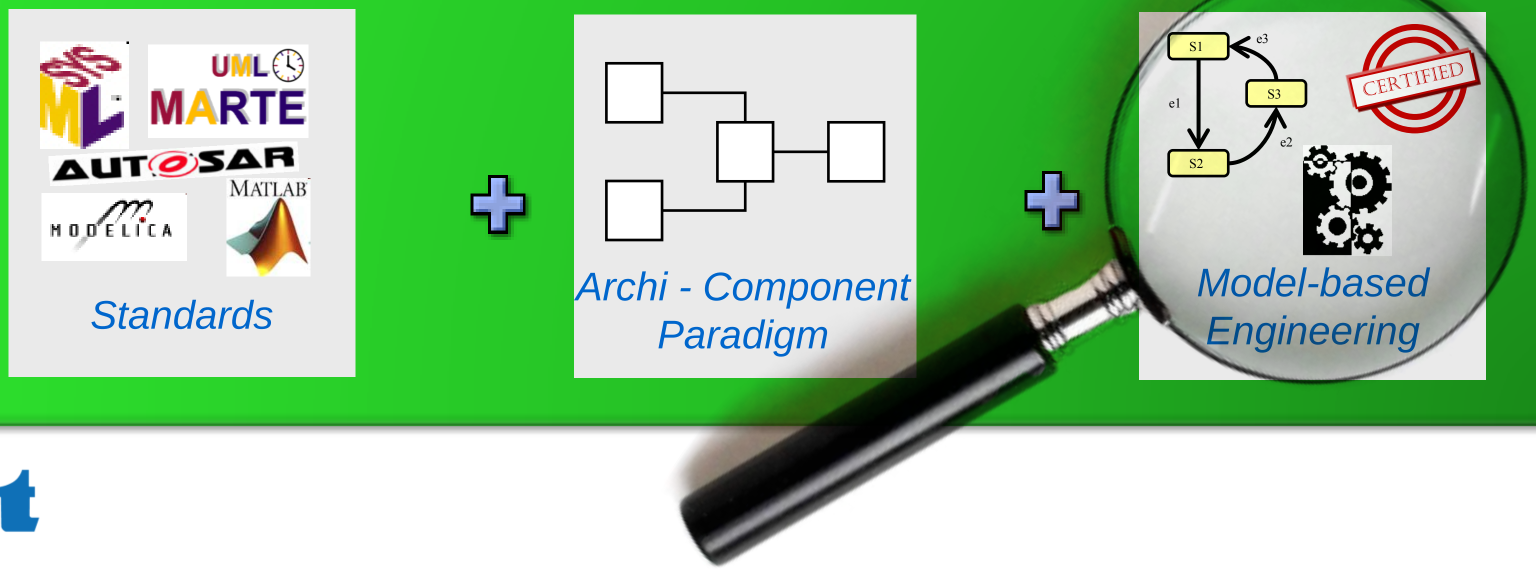
LISE Labs.

- Laboratory of model-driven engineering for embedded systems
  - A laboratory of ~30 persons (including 22 permanent members)
- Our domain
  - Specification, Design and Validation of Complex Critical Software-intensive System
- **Main research topics includes:** *Model-driven engineering for complex systems, Modeling language engineering, Formal verification and validation, Automatic test-generation, Model execution/compilation and code generation, Model monitoring, Model-based analysis (e.g., safety and performance), etc.*

THESIS FOR CHALLENGING THE DEVELOPMENT OF COMPLEX SYSTEMS

Going further for developing modern complex systems requires new advanced and innovative methods and tools!

Standards-based Architecture-centric Model-driven



Standards

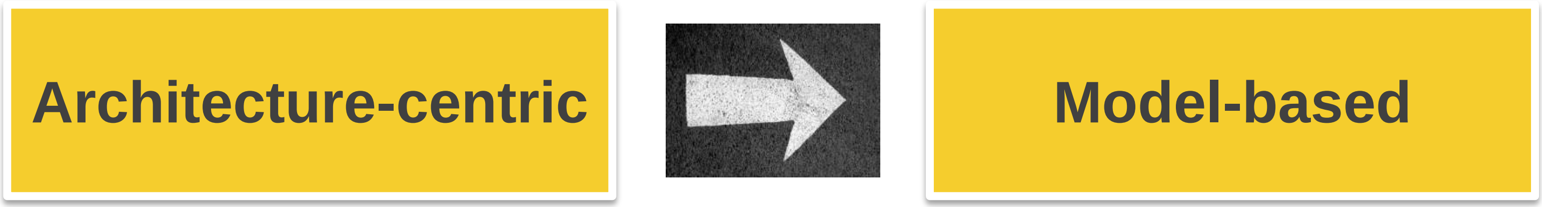
Archi - Component Paradigm

Model-based Engineering

SUMMARY

Architecture-centric design has opened the door to the need/use of modeling languages:

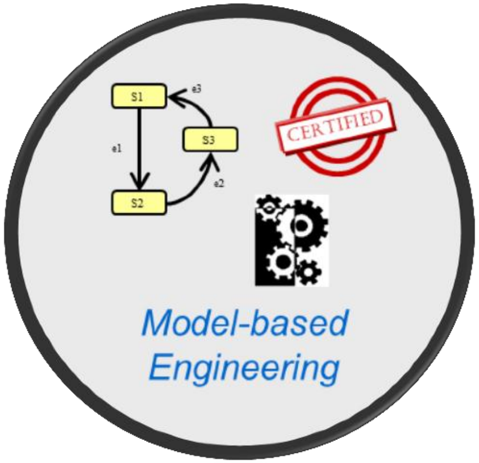
- Enabling the expression of the concepts of architecture description, decomposition, abstraction and view.
- And enabling to establish explicit relationships between elements at different abstraction levels and projected in different views.



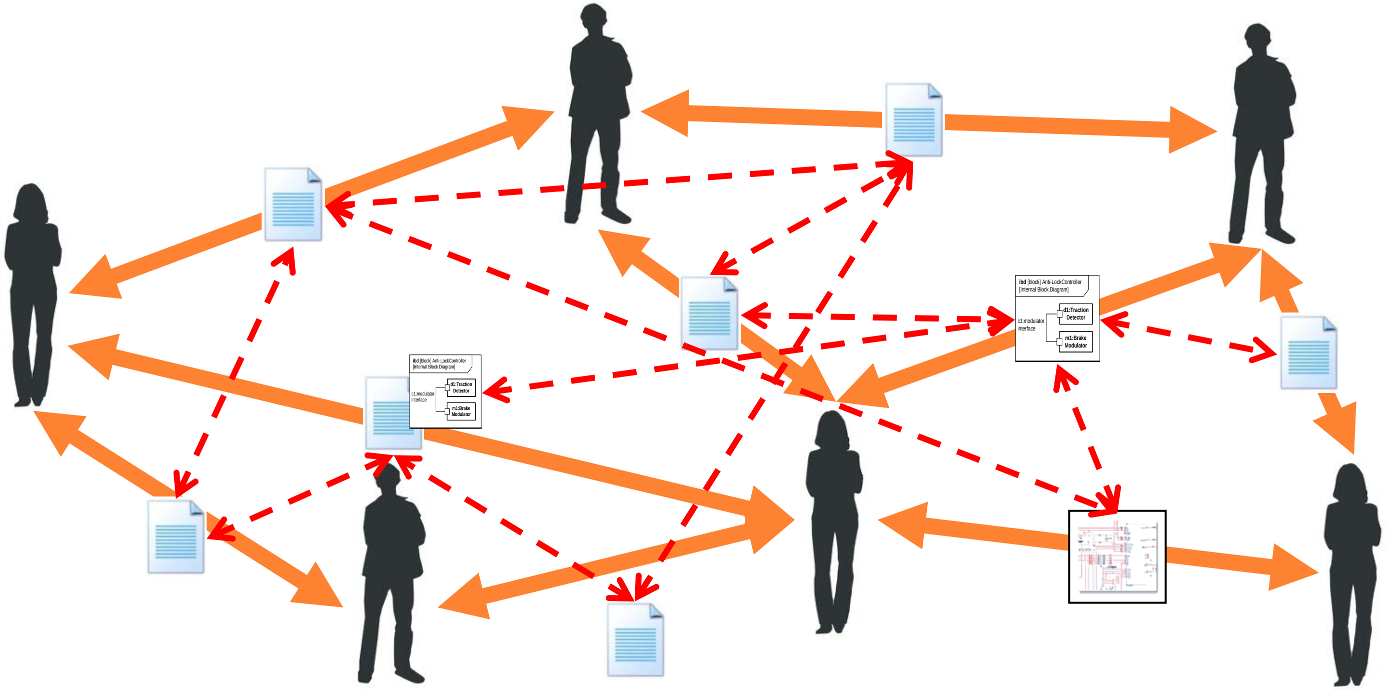
Architecture-centric

Model-based

WHAT IS MBE?



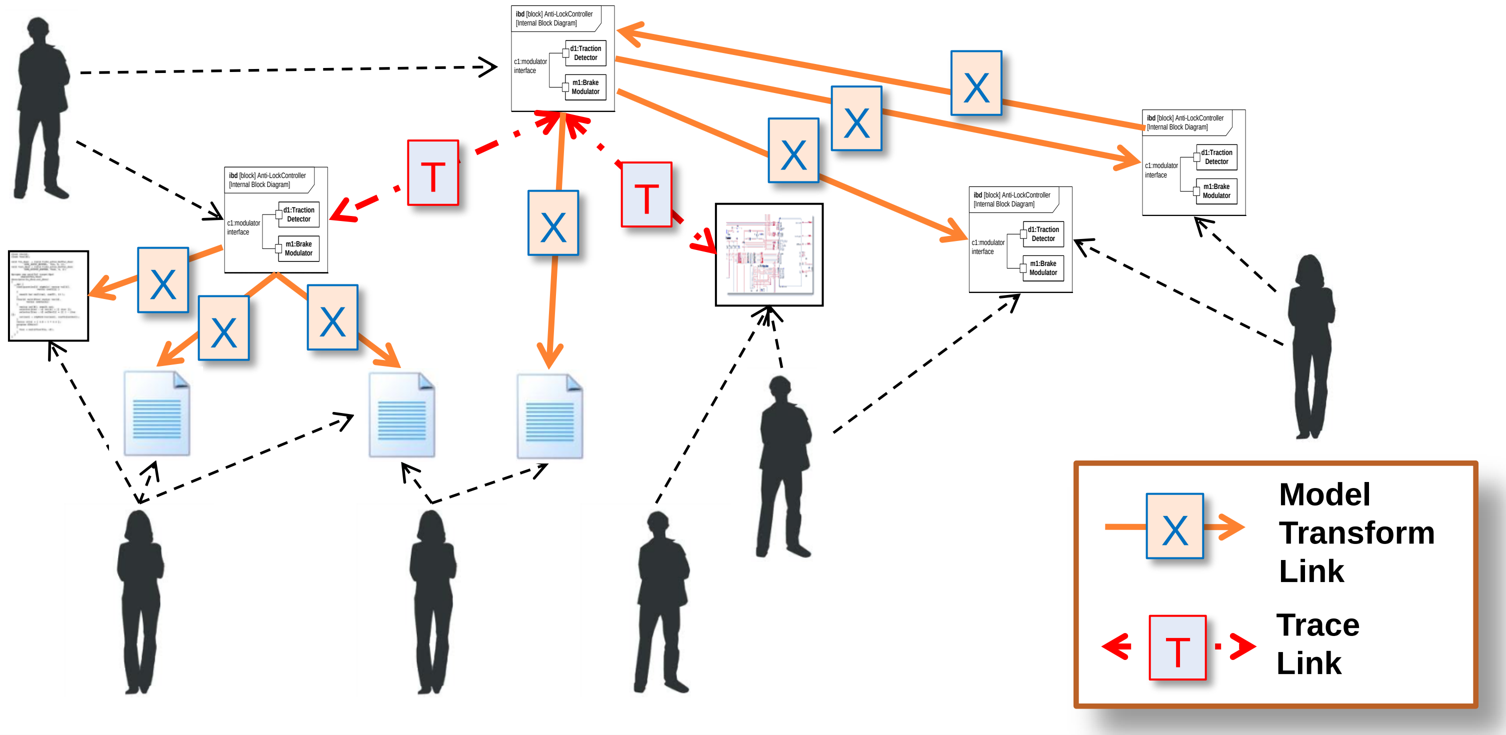
TRADITIONAL DEVELOPMENT APPROACH



Documents, schematics, models everywhere…
… connected informally through text and people’s memories

Unreliable / Inefficient / Non-scalable

MODEL-BASED ENGINEERING APPROACHES



Models, Models, Models Everywhere…

… but connected formally using computers and networks.

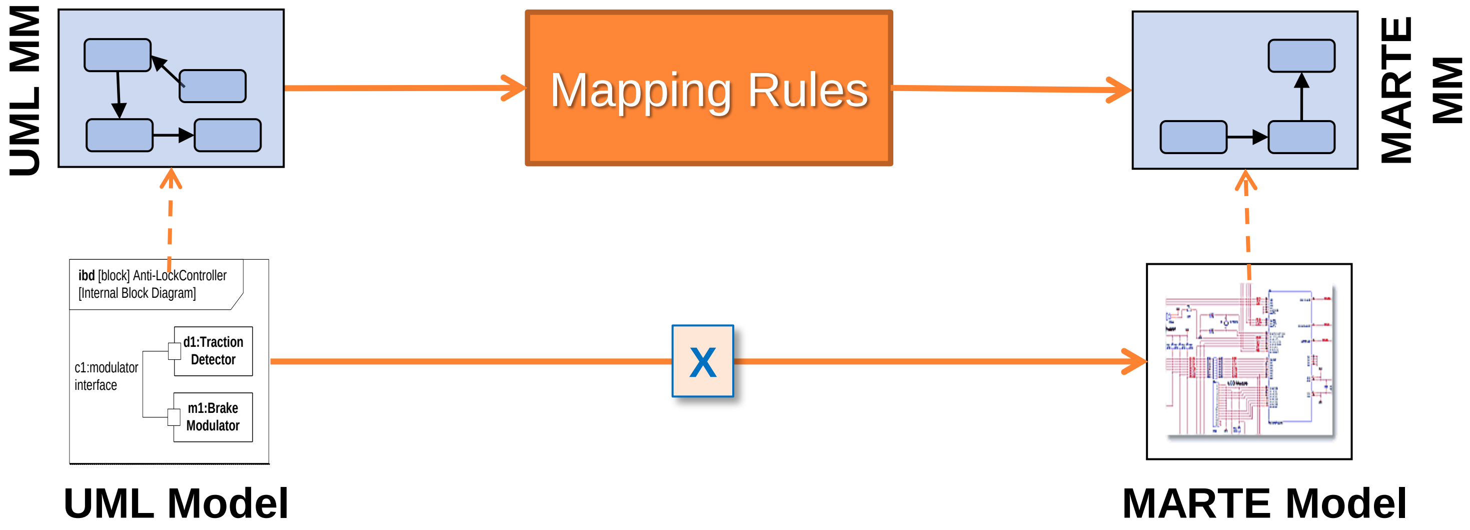
More efficient, More reliable, and More scalable.

FOOTNOTE ON MODEL TRANSFORMATIONS

Deriving a new model(s) from an existing one, possibly several, based on pre-defined automatable conversion rules:

- For different viewpoints (e.g., tester’s viewpoint, user’s viewpoint)
- For different levels of detail (e.g., detailed to abstract)
- For generating text documents from models

E.g., extracting the software scheduling information from a UML model and converting them into an equivalent MARTE model



EXPECTED ROLES OF MODELS

To communicate

- … their understanding and design intent to others

➔ Models are documentations

To specify

- ... the implementation of the system

➔ Models are specifications

To understand

- … the interesting characteristics of an existing or desired (complex) system and its environment

➔ Models are analysis tools

To predict

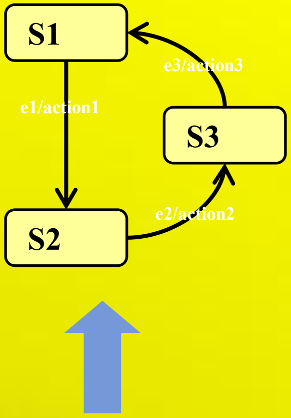
- … the interesting characteristics of the system by analyzing its model(s)

➔ Models are design assistants

TWO MAIN PRINCIPLES BEHIND MBE

Realm of ***modeling languages***

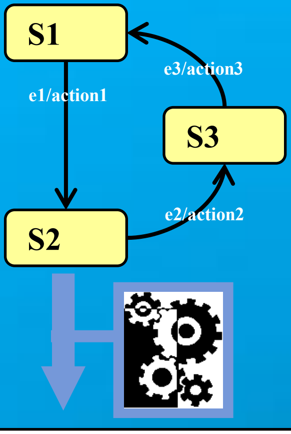
ABSTRACTION



```
SC_MODULE(producer)
{
sc_outmaster<int> out1;
sc_in<bool> start; // kick-start
void generate_data ()
{
for(int i =0; i <10; i++) {
out1 =i ; //to invoke slave;}
}
SC_CTOR(producer)
{
SC_METHOD(generate_data);
sensitive << start;}};
SC_MODULE(consumer)
{
sc_inslave<int> in1;
int sum; // state variable
void accumulate (){
sum += in1;
cout << "Sum = " << sum << endl;}
```

AUTOMATION

Realm of tools



```
SC_MODULE(producer)
{
sc_outmaster<int> out1;
sc_in<bool> start; // kick-start
void generate_data ()
{
for(int i =0; i <10; i++) {
out1 =i ; //to invoke slave;}
}
SC_CTOR(producer)
{
SC_METHOD(generate_data);
sensitive << start;}};
SC_MODULE(consumer)
{
sc_inslave<int> in1;
int sum; // state variable
void accumulate (){
sum += in1;
cout << "Sum = " << sum << endl;}
```

(Slide courtesy to Bran Selic, Malina Sofware)

Are we sure, MBE does work?

SOME PROOFS OF SUCCESS APPLYING MDE

J. Hutchinson, et al., "Empirical Assessment of MDE in Industry," ICSE 2011 (*).

- Systematic study of the effectiveness of model-based methods in for software development in industry.

J. Hutchinson, et al., "Model-Driven Engineering Practices in Industry," ICSE 2011 (*).

- Systematic study of the level of use of model-based methods in for software development in industry.

P. Mohagheghi and V. Dehlen, "Where is the Proof? – A Review of Experiences from Applying MDE in Industry," ECMDA 2008 (*).

- Review of available publications on industrial application of MBE in industry.

T. Weigert and F. Weil, "Practical Experiences in Using Model-Driven Engineering to Develop Trustworthy Computing Systems," IEEE SUTC 2006.

- Summary of systematic use of MBE in Motorola with evaluation.

The Middleware Co., "Model-Driven development for J2EE Utilizing a Model Driven Architecture (MDA) Approach," 2003.

- A systematic comparative study of traditional vs. model-based development on a software project.

Cloutier and M. Bone, "Compilation of SysML RFI – Final Report", Stevens Institute of Technology, 2010.

- Systematic study of the use and effectiveness of model-based methods in systems engineering in industry.

(*) = Sources that include extensive references to other surveys and experience reports

MAIN RESULTS FROM THESE EXPERIENCES

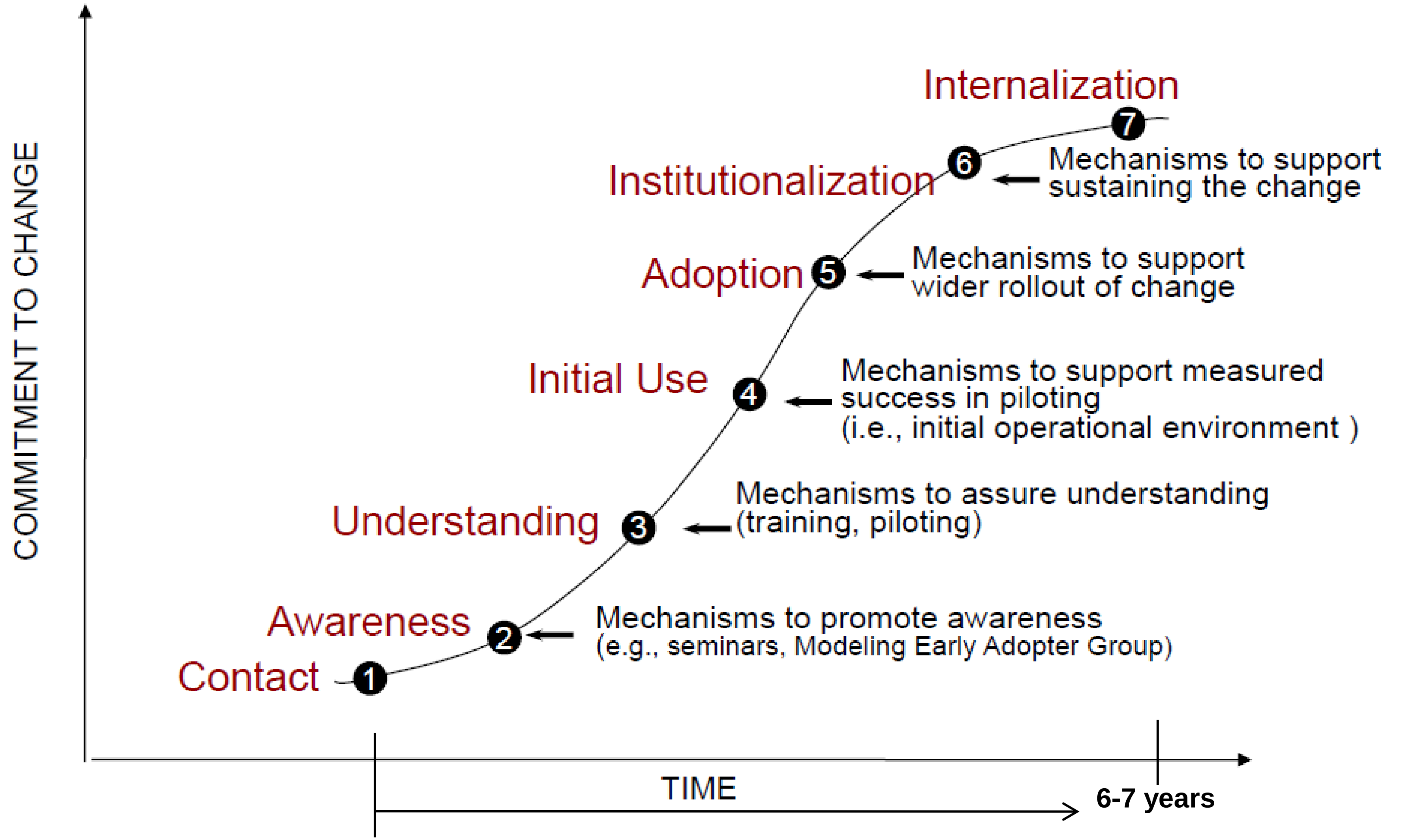
Diverse and widespread industrial experiences with MBE has demonstrated that it is effective in:

- Increasing productivity, product quality and complexity management.
- And also improving maintainability!

However, these studies also show that the introduction of MBE into a legacy organization must be gradually and systematically planned and executed or it will either disappoint or fail!

Primary hurdles to successful adoption of MBE:
Inadequate corporate commitment, Inexperience of development staff, Technology boycott by development staff, Inadequate tools and languages, Unrealistic expectations (overly ambitious first project), Cost of (re-)training and Cost of (re-)tooling.

NASA-JPL APPROACH TO INFUSION OF MBE


Internalization
7
6
Mechanisms to support sustaining the change
Institutionalization
Adoption
5
Mechanisms to support wider rollout of change
Initial Use
4
Mechanisms to support measured success in piloting (i.e., initial operational environment)
Understanding
3
Mechanisms to assure understanding (training, piloting)
Awareness
2
Mechanisms to promote awareness (e.g., seminars, Modeling Early Adopter Group)
Contact
1
COMMITMENT TO CHANGE
TIME
6-7 years

Adapted from Out from Dependency: Thriving as an Insurgent in a Sometimes Hostile Environment,
SuZ Garcia and Chuck Myers, SEPG Conference, 2001

SOURCE: C. Lin et al., "Experiences Deploying MBSE at NASA JPL", GeorgiaTech Workshop, 2010.

STROOL WITHIN OMG MDA-RELATED STANDARDS

FOR MODELING, UML2 IS THE LEADING CHOICE!

The UML provides a large number of concepts covering a large number of requirements/concerns: UML is indeed a family of modeling languages!

The UML is very popular

- **UML is indeed widely educated & disseminated:**
  - Academics courses, tutorials, books, professional trainings, mentors, …
- **UML is also widely implemented by commercial and open-source tools**
  - www.eclipse.org/papyrus ;-)

The UML enables heterogeneous modeling processes

- **It is a de facto “pivot” languages enabling integration of various formalisms needed for specific concerns:**
  - For requirements: DOORS, Rectify, …
  - For simulation: Modelica, Simulink, …
  - For V&V: Event B, Timed Automata, …
  - For implementing: C++, SystemC, Java, Perl, …
  - etc.

PAPYRUS: THE OPEN-SOURCE UML TOOL

Papyrus is the official open-source Eclipse UML2 modeling tool:

www.eclipse.org/papyrus

- Papyrus provides a complete graphical editor for both UML and SysML standards based on the MDT::UML2 component for its repository.
- Papyrus addresses the two key features expected from a UML2 graphical editor: modeling and profiling.
- Papyrus is highly customizable and extensible enabling DSML definitions based on standard UML profiles!
- Papyrus provides a support to MARTE 1.1 (including a rich text editor for VSL).

FROM UML TO DSML…

Originally intended for modeling software-intensive systems:

- UML models capture different views of a software system (e.g., data structure, run-time behavior, packaging and deployment)
- Inspired primarily by the concepts from object-oriented languages (class, operation, object, etc.) but now supporting functional-oriented design style.

However, the general nature of its concepts make UML2 suitable for extensions to specific modeling domains.

- Domain Specific Modeling Language by profiling the UML2!
- E.g., MARTE and SysML.

FOOTNOTE ON DSML & UML PROFILES (1)

Meta-modelling via MOF

- **For heavyweight extension mechanisms**
- **Ensures full manipulation of MMs**
  - Add, remove meta-classes and relationships between

Meta-models profiling

- **For lightweight extension mechanisms**
- **Adaptation of existing meta-models…**

  …no modifications of existing concepts!
  - e.g., UML MM constraints may not be suppressed, but additional one (compatible with existing one) may be added.
- **May extend any standard MM of the OMG**
  - e.g., UML profiles, BPMN profiles…

Profile vs. MOF → ?

- **Depend on your project context:** *e.g., scope of the extensions, Tooling constraints, Engineer level of experiment/education, Cost constraints.*

FOOTNOTE ON DSML & UML PROFILES (2)

UML profile definition

- **“A special kind of package containing stereotypes, modeling rules and model libraries that, in conjunction with the UML metamodel, define a group of domain-specific concepts and relationships.”**

Minimal benefits using UML profiles are:

- **Correctly defined profiles allow direct and effective reuse of the extensive available support provided for UML.**
  - E.g., Tools, methods, experts and trainings.
- **DSMLs based on UML profiles share a common semantic foundation which can greatly reduce the language fragmentation problem related to DSML-based approaches.**
  - Note: Ongoing formalisation of the UML at the OMG level!

FOOTNOTE ON DSML & UML PROFILES (3)

Possible rationale for defining a UML profile

- **Define a domain specific terminology, i.e. a domain specific notation instead of the plain UML2 notation.**
- **Complete/specialize the UML2 semantics for dealing with:**
  - UML Semantics Variation Points,
  - For clarifying ambiguous definition,
  - For specializing an existing semantics aspect of UML2.
- **Define usage constraints of the UML2 in order to drive/limit its usage**
  - e.g., for defining a domain specific methodology limiting the scope of UML.
- **Define new meta-information for annotating a model for a given purpose**
  - e.g. for code generation purpose, for enabling model-based analysis such as quality performance analysis, etc.

BUT WHY MARTE IN ADDITION TO UML?

Remember why UML was borne ?

- **Too many approaches, modelling languages & tools…**
  - Need to train engineer to a lot of different tools and languages

Need to unify all languages around a unique, common and shared language: UML

“not replace them, just aggregate, integrate and support them”

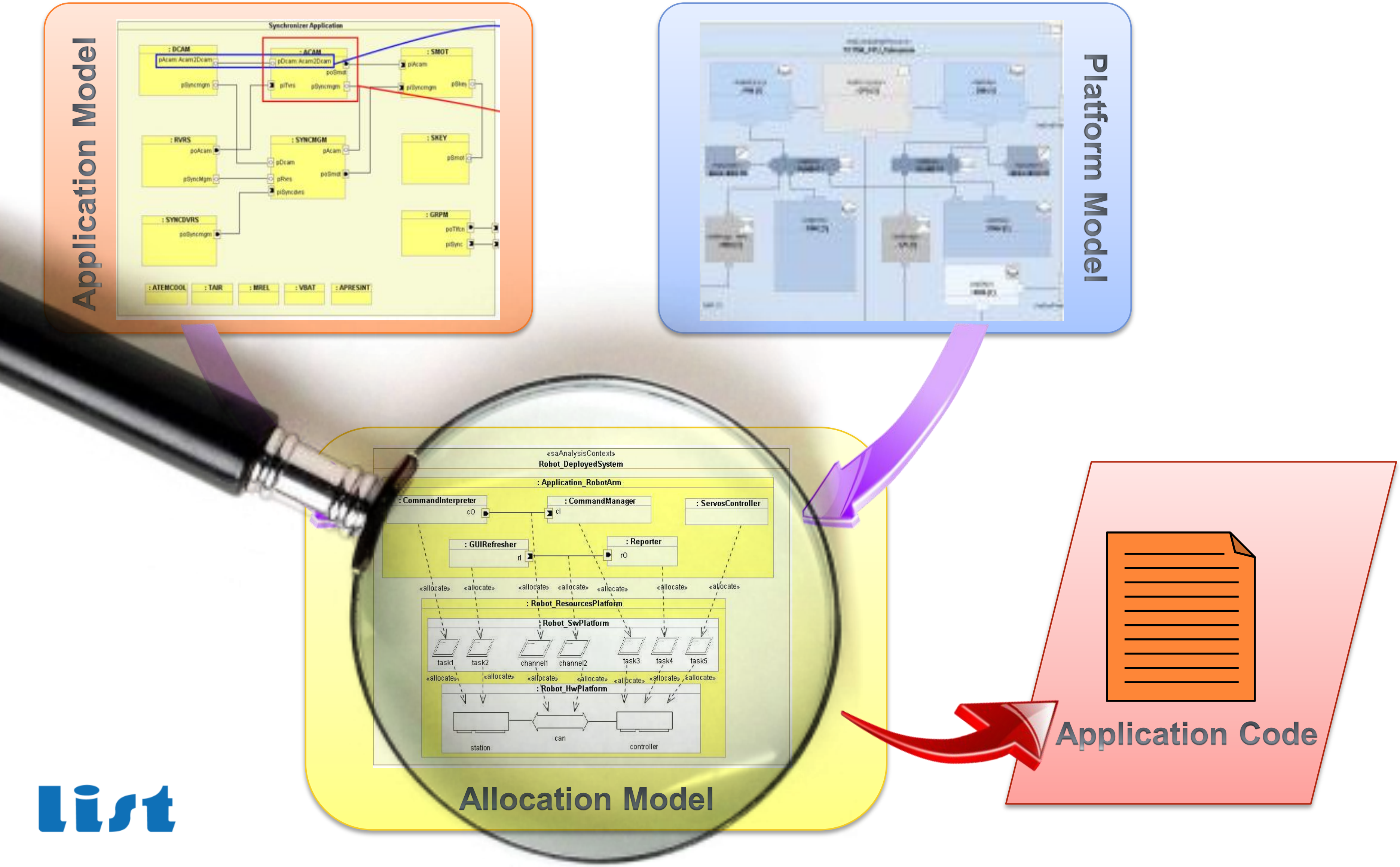
For RTE systems, it is a similar situation.

- **Too many specific approaches, languages and tools…**
  - Sometimes redundant and with few capabilities of interoperability
- **Often complex access to related advanced-tecnologies**
  - Difficulty (and then costly) for obtaining and managing required engineer expertise

Need to unify all languages around a unique, common and shared language: MARTE

“not replace them, just aggregate, integrate and support them”

MARTE SUPPORT FOR THE "Y-CHART"



Application Model

Platform Model

Allocation Model

Application Code

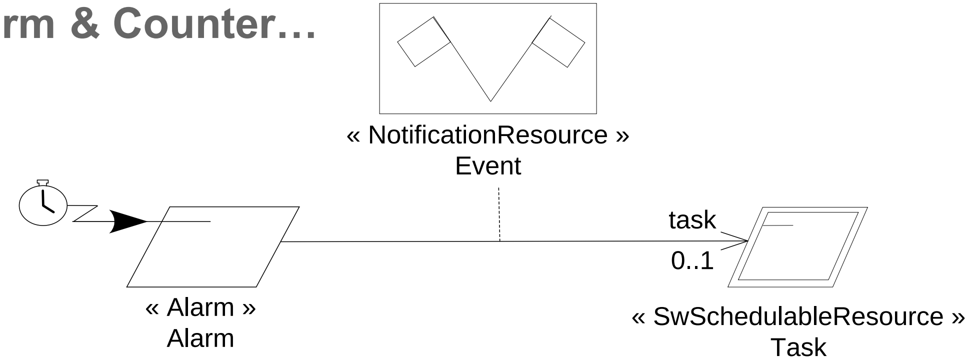
SOFTWARE PLATFORM-BASED MODELLING

SRM: Define constructs for modelling multitask design

- Real-Time Operating Systems (e.g. POSIX, OSEK/VDX and ARINC 653)
- Real-Time language libraries (e.g. ADA)

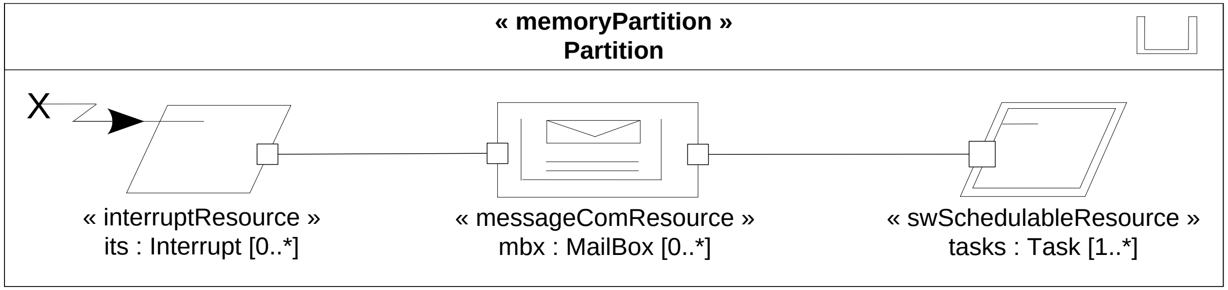
Ex.1: Concurrent execution mechanisms

- Task, Interrupt, Alarm & Counter…



Ex2.: Synchronization mechanisms

- Events, Mutual Exclusion Access Mechanisms…

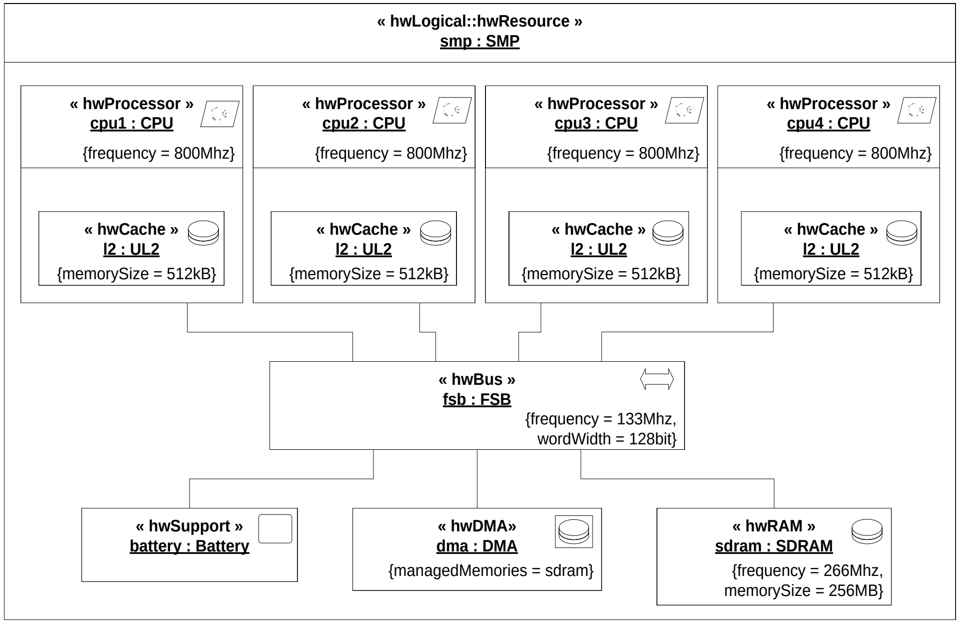


HRM: For describing structure of hardware ptf

- Different abstraction levels: e.g. for processor simulation, power consumption calculation and WCET analysis.
- Two sub-views:

Logical view (functionality)

Physical view (layouts)



« hwLogical::hwResource »
smp : SMP

« hwProcessor »
cpu1 : CPU
{frequency = 800Mhz}

« hwProcessor »
cpu2 : CPU
{frequency = 800Mhz}

« hwProcessor »
cpu3 : CPU
{frequency = 800Mhz}

« hwProcessor »
cpu4 : CPU
{frequency = 800Mhz}

« hwCache »
l2 : UL2
{memorySize = 512kB}

« hwCache »
l2 : UL2
{memorySize = 512kB}

« hwCache »
l2 : UL2
{memorySize = 512kB}

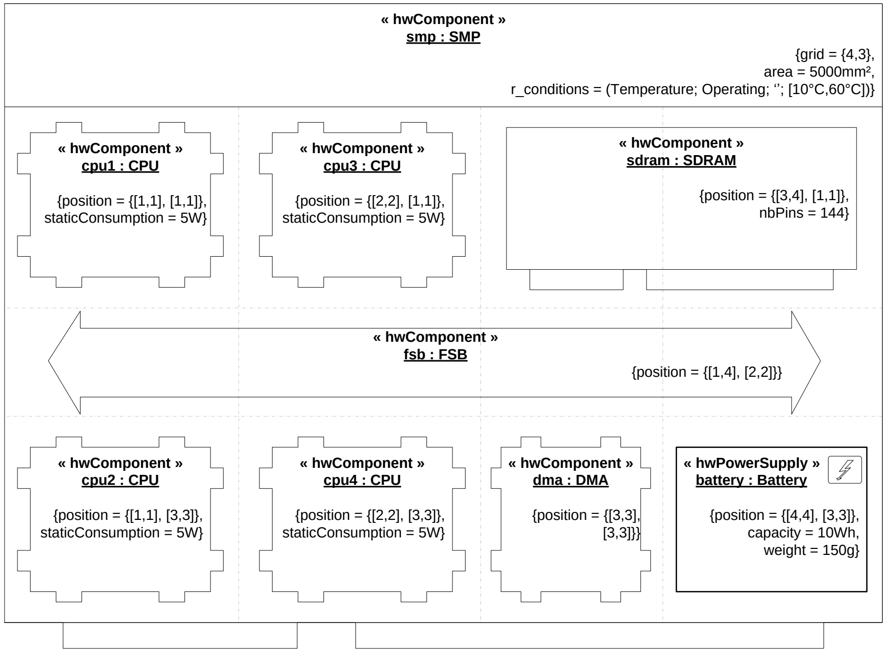
« hwCache »
l2 : UL2
{memorySize = 512kB}

« hwBus »
fsb : FSB
{frequency = 133Mhz,
wordWidth = 128bit}

« hwSupport »
battery : Battery

« hwDMA»
dma : DMA
{managedMemories = sdram}

« hwRAM »
sdram : SDRAM
{frequency = 266Mhz,
memorySize = 256MB}



« hwComponent »
smp : SMP
{grid = {4,3},
area = 5000mm²,
r_conditions = (Temperature; Operating; ‘’; [10°C,60°C])}

« hwComponent »
cpu1 : CPU
{position = {[1,1], [1,1]},
staticConsumption = 5W}

« hwComponent »
cpu3 : CPU
{position = {[2,2], [1,1]},
staticConsumption = 5W}

« hwComponent »
sdram : SDRAM
{position = {[3,4], [1,1]},
nbPins = 144}

« hwComponent »
fsb : FSB
{position = {[1,4], [2,2]}}

« hwComponent »
cpu2 : CPU
{position = {[1,1], [3,3]},
staticConsumption = 5W}

« hwComponent »
cpu4 : CPU
{position = {[2,2], [3,3]},
staticConsumption = 5W}

« hwComponent »
dma : DMA
{position = {[3,3],
[3,3]}}

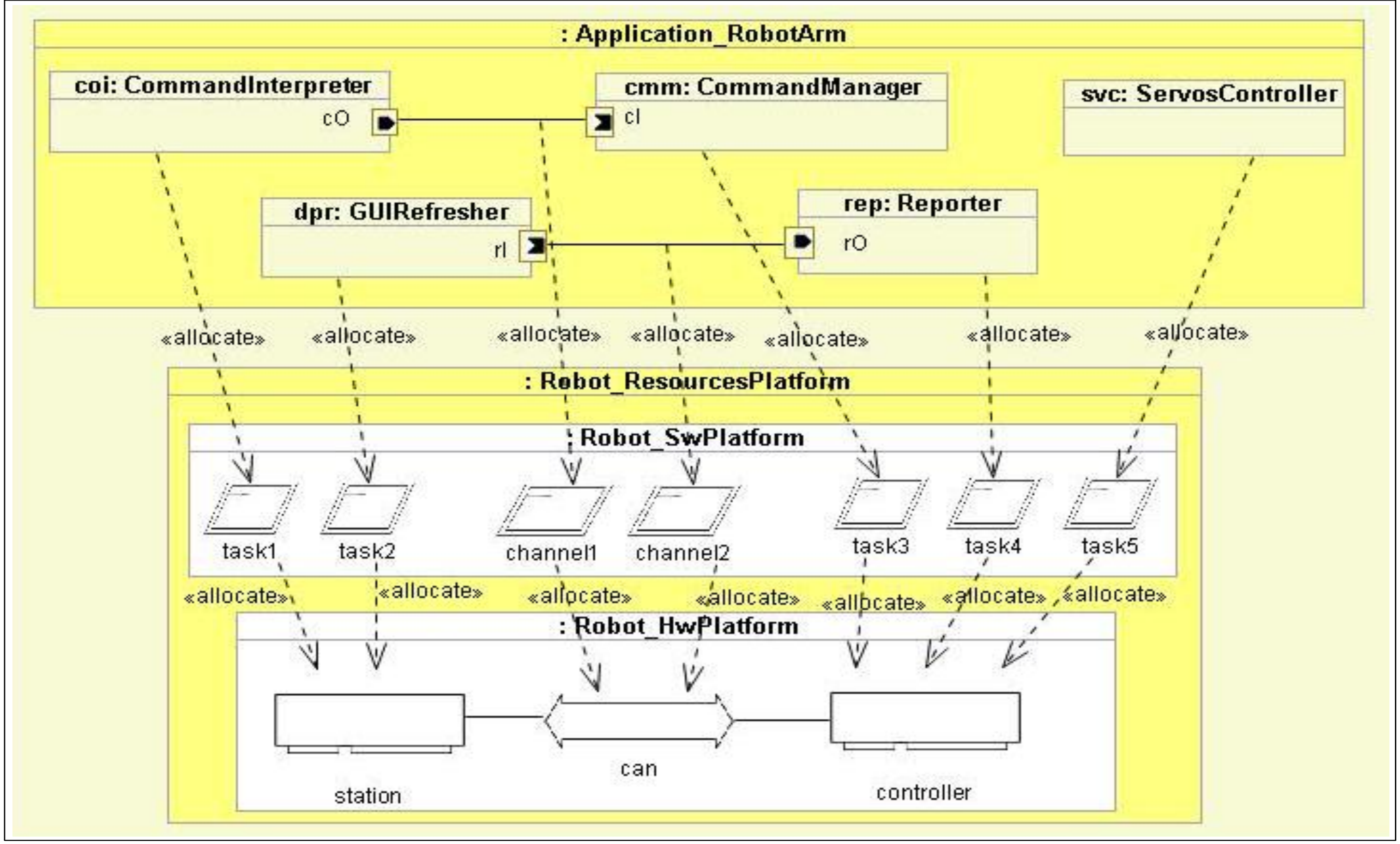
« hwPowerSupply »
battery : Battery
{position = {[4,4], [3,3]},
capacity = 10Wh,
weight = 150g}

EXAMPLE OF CONFIGURATION MODELLING USING A COMPOSITE STRUCTURE

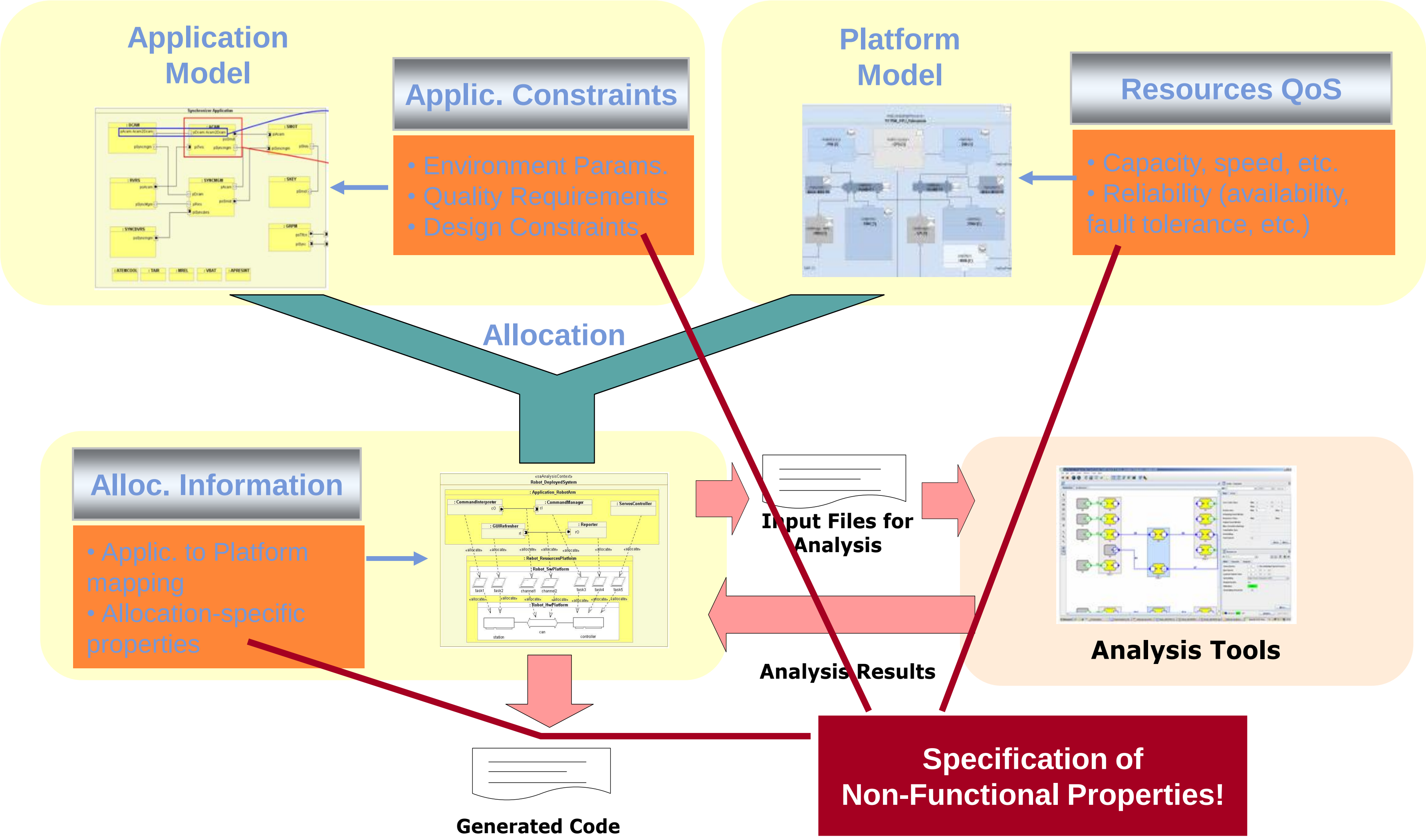
« configuration »

{mode= DegradedMode}

DegradedMode_SystemConfiguration



“Y-CHART” APPROACH AUGMENTED FOR SUPPORTING MODEL-BASED ANALYSIS



Application Model

Applic. Constraints

- Environment Params.
- Quality Requirements
- Design Constraints

Platform Model

Resources QoS

- Capacity, speed, etc.
- Reliability (availability, fault tolerance, etc.)

Allocation

Alloc. Information

- Applic. to Platform mapping
- Allocation-specific properties

Input Files for Analysis

Analysis Tools

Analysis Results

Generated Code

Specification of Non-Functional Properties!

SUMMARY

MARTE IS TO THE RTES DOMAIN AS UML TO THE SYSTEM & SOFTWARE DOMAIN:

A FAMILY OF LARGE AND OPEN SPECIFICATION FORMALISMS!

TOWARDS MORE FORMAL MDE

TOWARDS MORE FORMAL MDE @ OMG

fUML (v1.0, http://www.omg.org/spec/FUML/)

- Foundational UML (fUML) is an executable subset of standard UML that can be used to define, in an operational style, the structural and behavioral semantics of systems.

Alf (v1.0)

- Textual surface representation for UML modeling elements with the primary purpose of acting as the surface notation for specifying executable (fUML) behaviors within an overall graphical UML model.
- Also provides an extended textual notation for structural modeling within the fUML subset.

UML2.5 NEW

- Complete revision of its text description to simplify its presentation and disambiguate as much as possible its semantics.

Precise semantics of UML Composite Structures RFP NEW

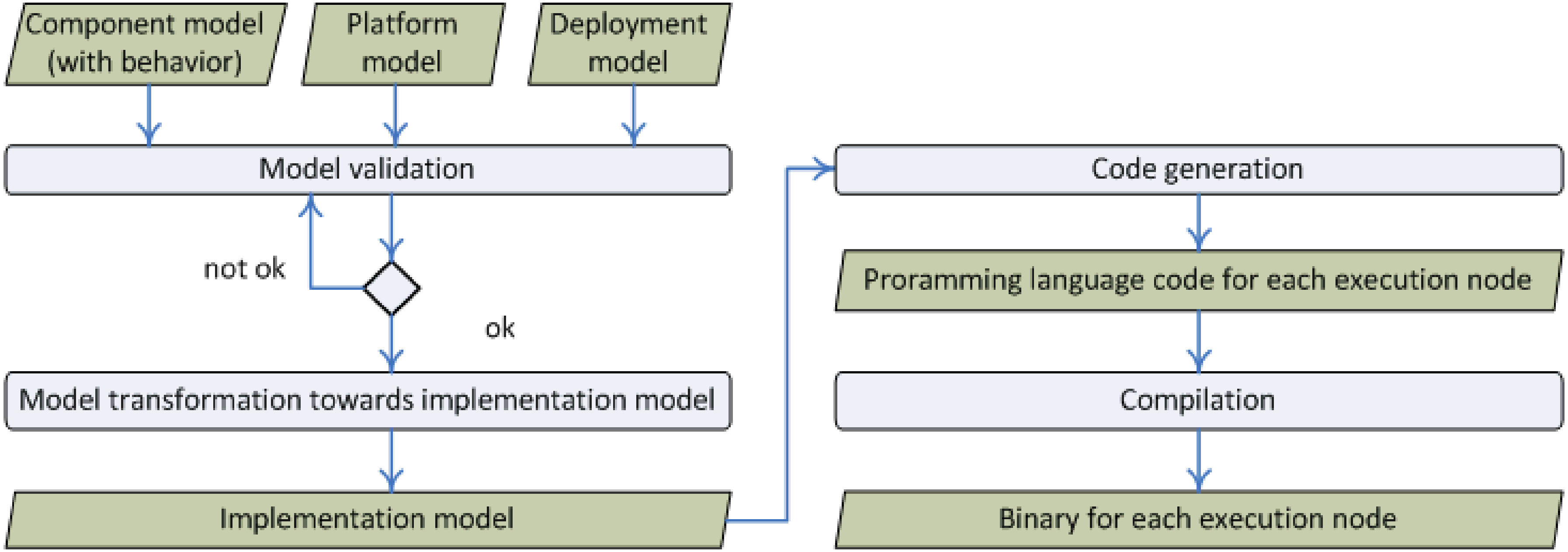
- Solicit a new specification defining a precise semantics for UML composite structures and their extensions.
- Containing two dedicated appendix for both MARTE and SysML.

EXECUTIVE MODELS: FROM MODEL TO CODE…

Based on standard OO-UML to code generators

Open and flexible framework for CBSE

- Open support for advanced communication and computation models
- Support for application deployment: instantiation, configuration & allocation


Component model (with behavior)
Platform model
Deployment model
Model validation
not ok
ok
Model transformation towards implementation model
Implementation model
Code generation
Proramming language code for each execution node
Compilation
Binary for each execution node

C++ CODE GENERATION

Support for two modeling paradigms

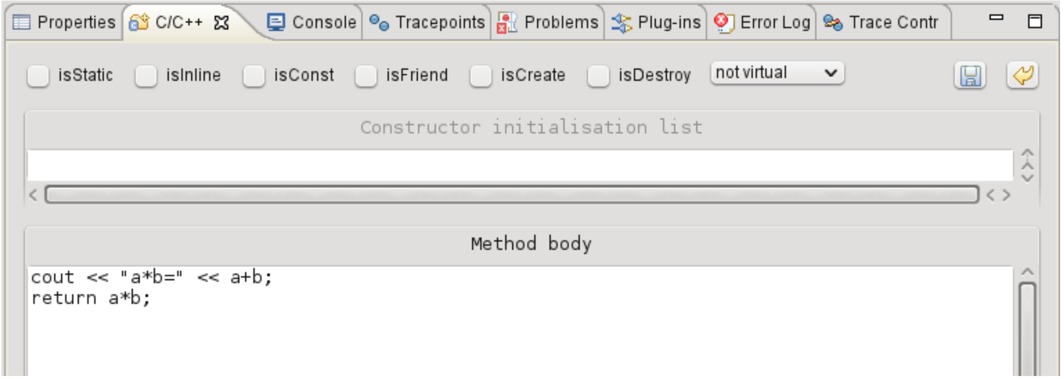
- Object-oriented (OO) model denoted by:
  - Class diagrams and state-machine diagrams
- Component-based (CB) model denoted by:
  - Composite-structure, class and state-machine diagrams

C++ “optimization” of a UML model

- Dedicated UML profile for C++
  - Current support for C++ 03 (*C++11 in preparation: e.g., scoping of enumerations*)
- UML towards C++
  - C++ like associations: support for pointer, ref, typdefs, friend, …
  - Model library with C++ primitive types (int, float, int32_t, …)
  - Enables manual #include directives (chiefly managed automatically)
  - UML packages => C++ namespace & file system hierarchy
  - Template support

A SPECIFIC PAPYRUS CUSTUMIZATION FOR C++

Dedicated and simple UI for managing C++ annotations onto UML models



☐ isStatic ☐ isInline ☐ isConst ☐ isFriend ☐ isCreate ☐ isDestroy not virtual

Constructor initialisation list

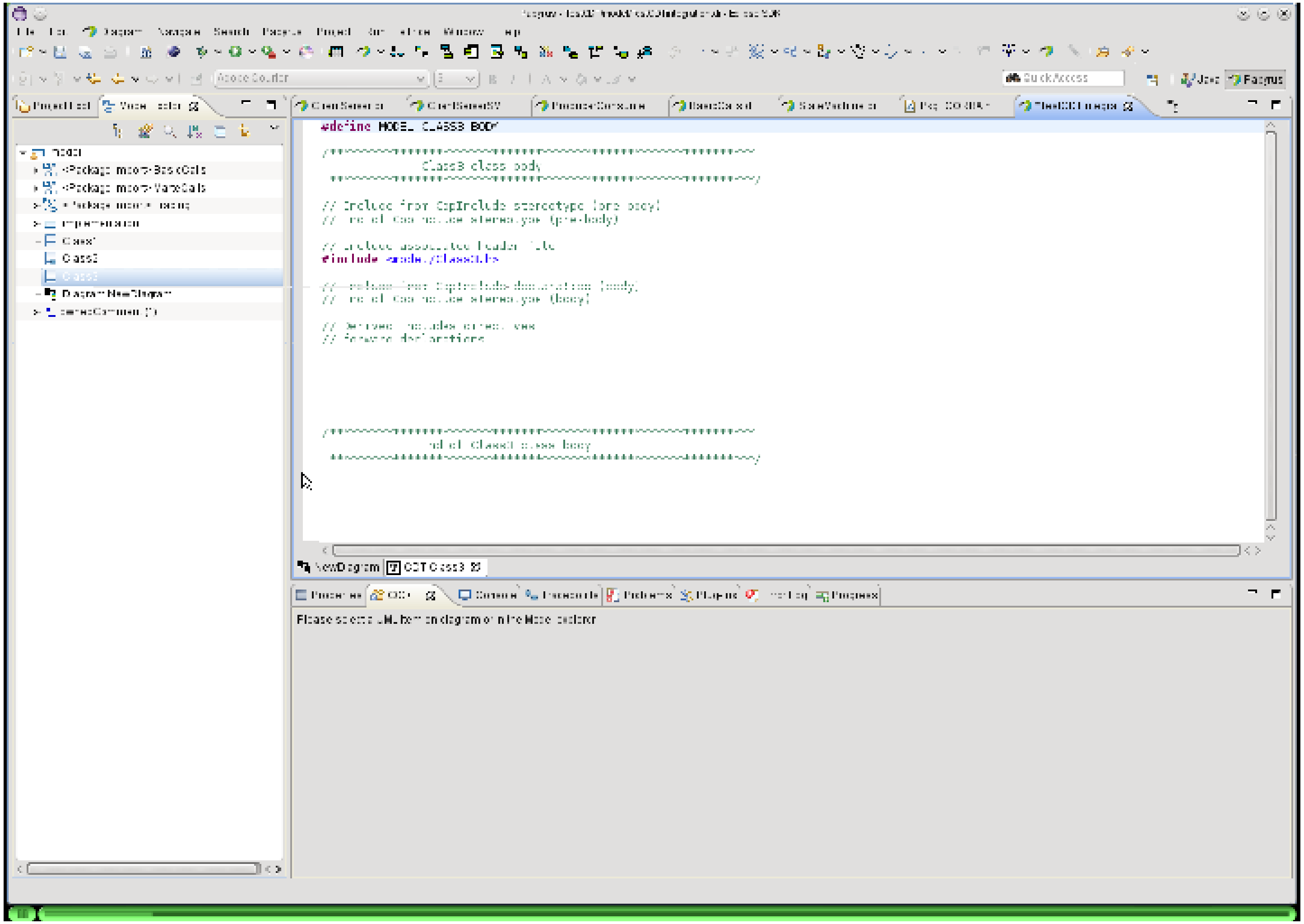
Method body

```
cout << "a*b=" << a+b;
return a*b;
```

Integration of CDT into Papyrus to manipulate the generated C++

- Keep synchronized both model and code views!

DEMO ON UML MODEL ⇔ C++ CODE: GENERATION & SYNCHRONIZATION



```
#define MODEL_CLASS3_BODY

/************************************************************
              Class3 class body
 ************************************************************/

// Include from CppInclude stereotype (pre-body)
// End of CppInclude stereotype (pre-body)

// include associated header file
#include <model/Class3.h>

// Include from CppInclude declaration (body)
// End of CppInclude stereotype (body)

// Derived includes directives
// forward declarations



/************************************************************
              End of Class3 class body
 ************************************************************/
```

Please select a UML item on diagram or in the Model Explorer

THANK
YOU

Acknowledgments
to Bran Selic
and the LISE team.

www.eclipse.org/papyrus

Commissariat à l’énergie atomique et aux énergies alternatives
Institut Carnot CEA LIST
Centre de Saclay | 91191 Gif-sur-Yvette Cedex
T. +33 (0)169 077 093 | M. +33 (0)688 200 047

Direction	DRT
Département	DILS
Laboratoire	LISE

Etablissement public à caractère industriel et commercial | RCS Paris B 775 685 019