



Institut  
Mines-Telecom

# Model-Based Design of Embedded Systems

Ludovic Apvrille,  
[ludovic.apvrille@telecom-paristech.fr](mailto:ludovic.apvrille@telecom-paristech.fr)

COMELEC Seminar, Paris, France



# Outline

## Introduction

Context

Model-Driven Engineering

## Synthetic overview of contributions

UML Profiles

Overview

TTool

## Focus on a few contributions

Partitioning

Handling security

Deployment

## Conclusions and perspectives

Conclusions

Perspectives



# Outline

Introduction

Context

Model-Driven Engineering

Synthetic overview of contributions

Focus on a few contributions

Conclusions and perspectives

# Designing Embedded Systems



How to Handle Complexity?

Modeling and verification!  
(But there are other options)

# Modeling is not Really a New Technique...

...and it is not limited to Software!



# Software Development Techniques for E.S.

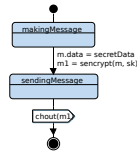
## Code-based approaches

- ▶ Extreme Programming
  - ▶ Strongly tested step-by-step code increments
- ▶ Agile Software Development
  - ▶ Focus on change in specification

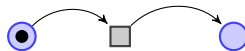


## Model-based approaches

- ▶ V-Cycle
  - ▶ KAOS, AADL, MDE, ...



- ▶ Formal models
  - ▶ B, LOTOS, Petri nets, ...



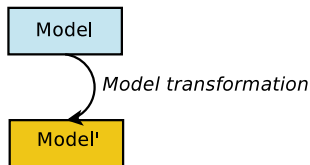
# Model Driven Engineering

## Definition

- ▶ Process based on abstract graphical representations for a given domain
- ▶ Intends to improve software engineering quality criteria
  - ▶ Reliability, extensibility, maintainability, . . .
- ▶ Should enhance team communication and documentation

## Abstraction levels

- ▶ Platform Independent Model, Platform Specific Model
- ▶ Model transformations



# UML Profiles

## Definition

- ▶ UML defined extension mechanisms to e.g.,
  - ▶ Define new operators
  - ▶ Provide a semantics
  - ▶ Give a methodology

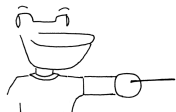
## Example of profiles

- ▶ Profiles defined by OMG (e.g., SPT, MARTE, SysML)
- ▶ Profiles defined by tool vendors (e.g. in Rhapsody, Artisan)
- ▶ User-defined and company-defined models



# UML Profiles and MDE

*UML profiles are a way to define domain-specific languages for MDE*



## Our contribution in MDE

Definition of UML profiles for modeling and verifying complex embedded systems

Definition of methodologies based on the V-cycle  
Definition of model transformations for simulation, formal verification and code generation purpose  
Implementation in a toolkit (TTool)

# Outline

Introduction

**Synthetic overview of contributions**

UML Profiles

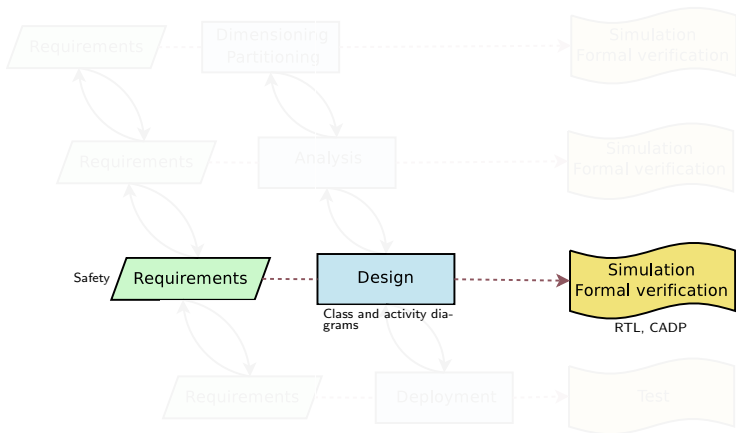
Overview

TTool

Focus on a few contributions

Conclusions and perspectives

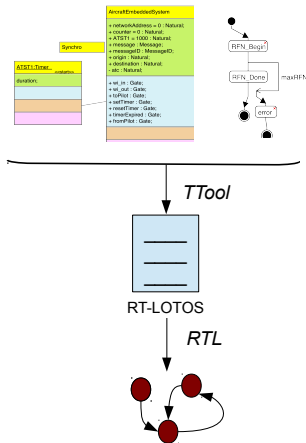
# TURTLE: A Formally Defined UML Profile



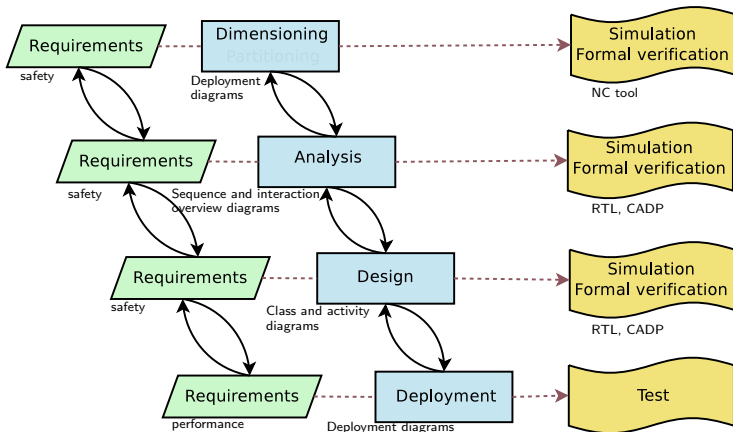
L. Apvrille, J.-P. Courtiat, C. Lohr, P de Saqui-Sannes , " TURTLE: A Real-Time UML Profile Supported by a Formal Validation Toolkit", IEEE Transactions on Software Engineering, Vol. 30, No. 7, pp. 473-487, July 2004

# TURTLE: A Formally Defined UML Profile (Cont.)

- ▶ Developed in the scope of my Ph.D.
- ▶ Partners: Thalès Alenia Space, LAAS-CNRS, ISAE
- ▶ Software design: class and activity diagram
  - ▶ Communication based on synchronous exchanges
  - ▶ Non-deterministic choices
  - ▶ Non-deterministic time intervals
- ▶ Model transformation to RT-LOTOS



# Extending TURTLE

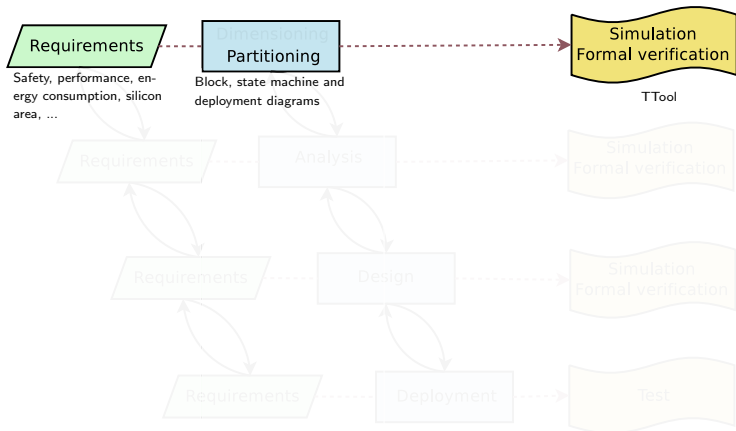


L. Apvrille, P de Saqui-Sannes, F. Khendek "TURTLE-P: A UML Profile for the Formal Validation of critical and Distributed Systems", SoSym (Software and System Modeling) Journal, Springer, Pages: 1-18, July 2006

## Extending TURTLE (Cont.)

- ▶ Developed in the scope of my post-doctorate and in LabSoC
- ▶ Collaboration with ISAE on Dimensioning stage
  - ▶ Network calculus toolkit for computing the Worst Case
  - ▶ Use case provided by Airbus
- ▶ Collaboration with ISAE and Concordia University for analysis and deployment stages
- ▶ Other partners / projects: LAAS-CNRS, UDCast, European project Maestro, ANR project Safecast, DoceaPower
- ▶ 1 Ph.D. completed (Benjamin Fontan, ISAE)

# DIPLODOCUS: HW/SW Partitioning



D. Knorreck, L. Apvrille, R. Pacalet, "Formal System-level Design Space Exploration", Concurrency and Computation: Practice and Experience, John Wiley and Sons, Ltd, 2012.

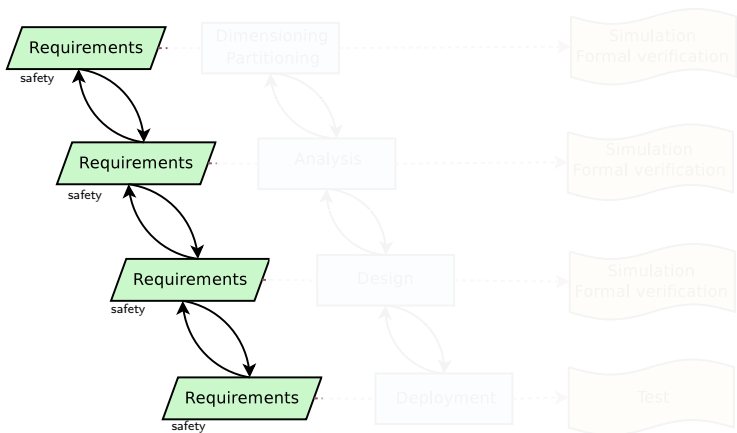
## DIPLODOCUS (Cont.)



- ▶ High abstraction level ("system-level")
- ▶ Can be used for Design Space Exploration
- ▶ Developed at LabSoC
- ▶ Partners: Texas Instruments, Freescale, European project EVITA, European project SACRA, LIP6
- ▶ 2 Ph.D. completed (Chafic Jaber, Daniel Knorreck), 3 on-going Ph.D. (Jair Gonzalez-Pina, Fériel Ben Abdallah, Andrea Enrici)
- ▶ Applied to:
  - ▶ Multimedia system, e.g., partitioning of smartphone platforms
  - ▶ Telecommunication systems, e.g., partitioning of LTE



# TEPE: Requirements and Property modeling

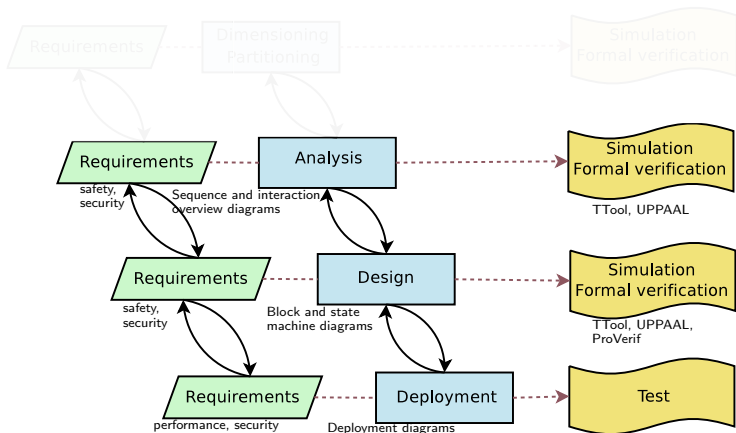


Daniel Knorreck, Ludovic Aprville, Pierre de Saqui-Sannes, "TEPE: A SysML Language for Time-Constrained Property Modeling and Formal Verification", ACM SIGSOFT Software Engineering Notes, Vol. 36, No 1, pp. 1-8, January 2011.

## TEPE (Cont.)

- ▶ Requirement modeling within SysML requirement diagrams
- ▶ Graphical modeling of safety properties using SysML Parametric Diagrams
  - ▶ Reachability, liveness
- ▶ Handle logical and temporal constraints
- ▶ 2 Ph.D. completed (Benjamin Fontan, Daniel Knorreck)
- ▶ Collaboration with ISAE

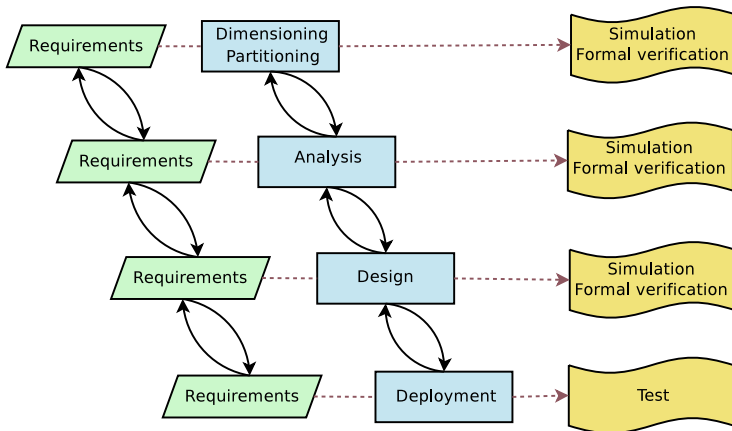
# AVATAR: Taking Into Account Security



# AVATAR

- ▶ Main idea: safety and security property proofs at the push of a button
- ▶ 2 Ph.D. completed (Muhammad Sabir Idrees, Gabriel Pedroza)
- ▶ Defined in the scope of the European project EVITA, Collaboration with ISAE and Eurecom
  - ▶ Used to model and prove security properties of cryptographic protocols for automotive systems
- ▶ Most TURTLE users have switched to AVATAR
- ▶ Widely used for educational purpose (Universities, training in companies, tutorials, etc.)

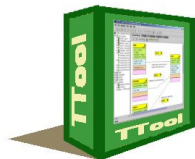
# Overview of Contributions



# TTool: A Multi Profile Platform

## TTool

- ▶ Open-source toolkit mainly developed by Telecom ParisTech / COMELEC
- ▶ Multi-profile toolkit
  - ▶ DIPLODOCUS, AVATAR, ...
- ▶ Support from academic (e.g. INRIA, ISAE) and industrial partners (e.g., Freescale)



## Main ideas

- ▶ Lightweight, easy-to-use toolkit
- ▶ Simulation with model animation
- ▶ Formal proof at the push of a button



# Outline

Introduction

Synthetic overview of contributions

**Focus on a few contributions**

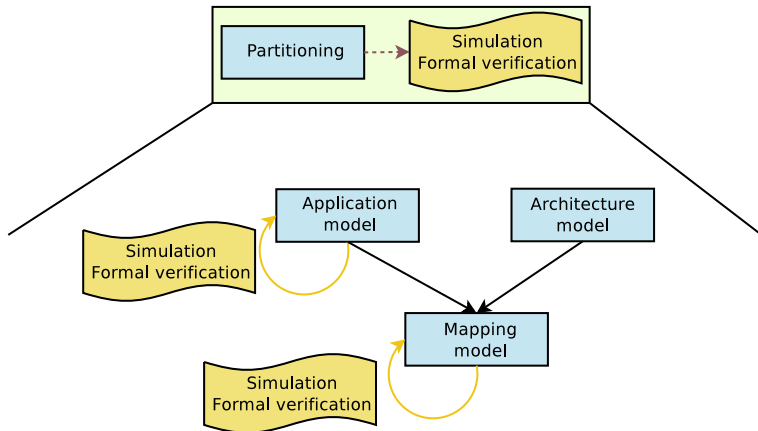
Partitioning

Handling security

Deployment

Conclusions and perspectives

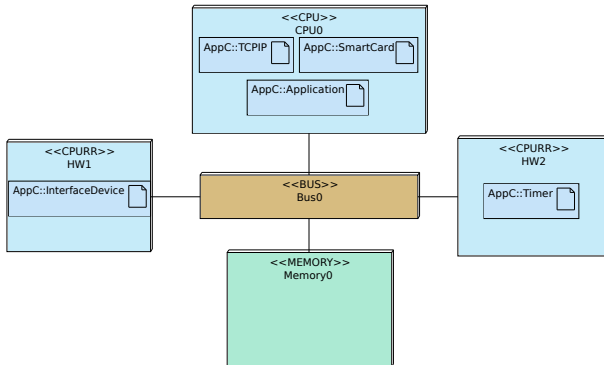
# Partitioning with the Y-Methodology





## Mapping

- ▶ Tasks are mapped on execution nodes (e.g., CPUs, HWAs)
- ▶ Channels are mapped on communication and storage nodes

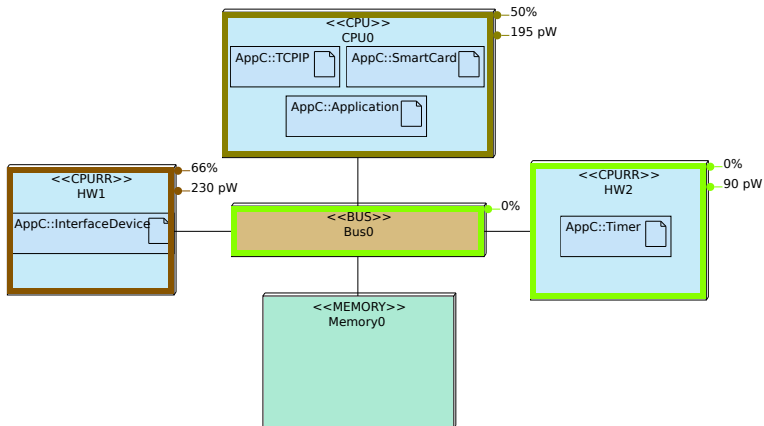


# After-Mapping Simulation

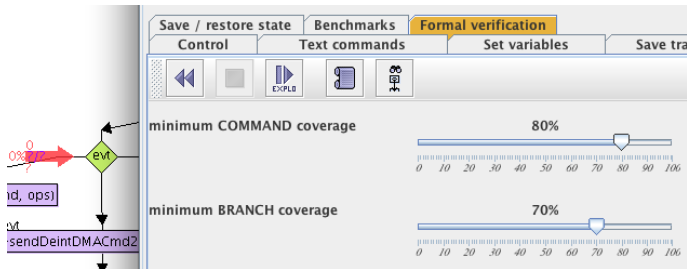
- ▶ TTool built-in simulator
- ▶ **Extremely fast**
- ▶ Diagram animation
- ▶ Step-by-step execution, breakpoints, etc.

The screenshot displays the TTool simulator interface. The top window shows the project name 'TechTool/TURTLEModeling/Diaf/IC/SmartCardProtocol\_1udo\_TEPE\_01.xml'. Below the menu bar, there are tabs for 'AppC', 'Mapping1', and 'Mapping2'. A toolbar contains various icons for simulation control. The main area shows a state machine diagram with nodes: 'req activation0', 'evt reset0', 'evt >answerToReset0', 'evt pTS0', 'out pTSConfirm0', 'data exchange', and a loop 'for(i=0,j=nbOfComputedPack...'. A green arrow labeled '0%' points to the 'out pTSConfirm0' node. A 'ch1 fromDtos' block is at the bottom. On the right, a 'Commands' panel includes 'Save / restore state', 'Benchmarks', 'Form', 'Control', and 'Text commands' sections, with a list of system components like CPU2 (3), Bus0 (2), Memory0 (1), AppC\_\_Applicat, and AppC\_\_fromAtoT.

## After-Mapping Simulation (Cont.)



# After-Mapping Coverage-Enhanced Simulation



Possibility to select a given part of the model to be explored

- ▶ Minimum percentage of operators coverage
- ▶ Minimum percentage of branch coverage

Implementation: TTool built-in model-checking techniques

## After-Mapping Formal Verification

- ▶ TTool built-in model checker can compute all possible execution paths
- ▶ Graph analysis and visualization

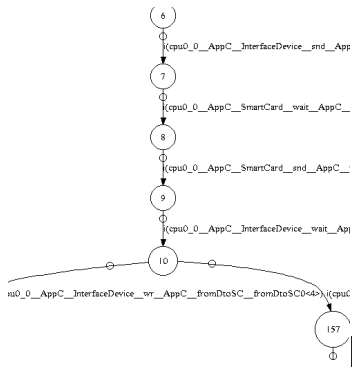
Analysis on the last DIPLODDUCS.RS

Deadlocks Shortest Paths Longest Paths

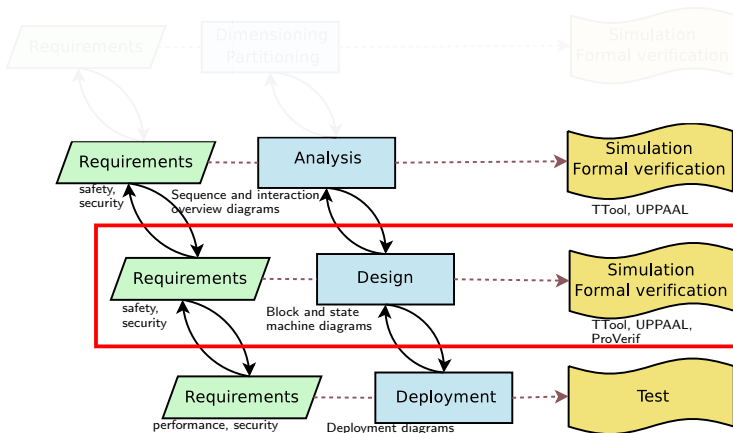
General info. Statistics

Transition	Nb	
allCPUsTerminated<24>	1	(176, 177)
allCPUsTerminated<26>	1	(172, 173)
allCPUsTerminated<38>	4	(81, 82), (98, 99), (115, 116)
allCPUsTerminated<40>	4	(77, 78), (94, 95), (111, 112)
allCPUsTerminated<43>	1	(64, 65)
allCPUsTerminated<45>	1	(60, 61)
allCPUsTerminated<47>	1	(138, 139)
allCPUsTerminated<49>	1	(134, 135)
allCPUsTerminated<53>	1	(47, 48)
allCPUsTerminated<55>	1	(43, 44)
cpu0_0_AppC_Application_sn...	8	(46, 47), (63, 64), (80, 81), (9
cpu0_0_AppC_Application_sn...	8	(40, 41), (57, 58), (74, 75), (9
cpu0_0_AppC_Application_sn...	8	(33, 34), (50, 51), (67, 68), (8
cpu0_0_AppC_Application_sn...	8	(37, 38), (54, 55), (71, 72), (8
cpu0_0_AppC_Application_wa...	8	(32, 33), (49, 50), (66, 67), (8
cpu0_0_AppC_Application_wr...	8	(38, 37), (53, 54), (70, 71), (8
cpu0_0_AppC_InterfaceDevice...	1	(10, 157)
cpu0_0_AppC_InterfaceDevice...	1	(0, 1)

Close



# Security-Oriented Design with AVATAR



## Design: Features for Security

**Initial knowledge:** Introduced as a common knowledge of the system, or of a specific session of the system:

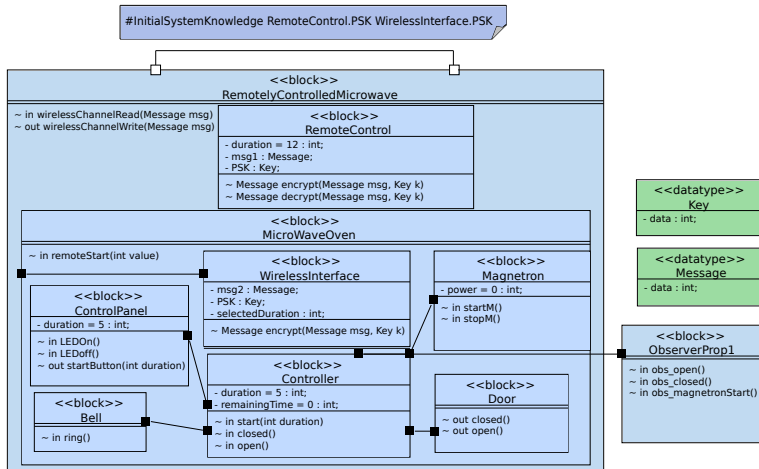
**#InitialSystemKnowledge** *Alice.sk Bob.sk*

**Cryptographic functions:** Predefined in each AVATAR block: *MAC(), encrypt(), decrypt(), sign(), verifyMAC(), verifySign()...*

**Communication Architecture:** Common public channels can be defined in blocks. Attackers can eavesdrop public channels

**Attacker model:** Taken from the underlying security framework *ProVerif*

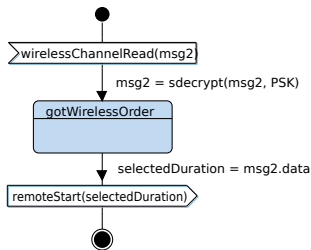
# Design: Architecture





## Detailed Design

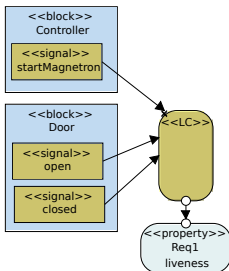
- ▶ Block's behaviour is described in terms of SysML State Machine Diagrams
- ▶ Non deterministic choices, non deterministic temporal operators



# Property Modeling

## Safety properties

- ▶ Customized Parametric Diagrams (TEPE)
- ▶ Reachability, liveness

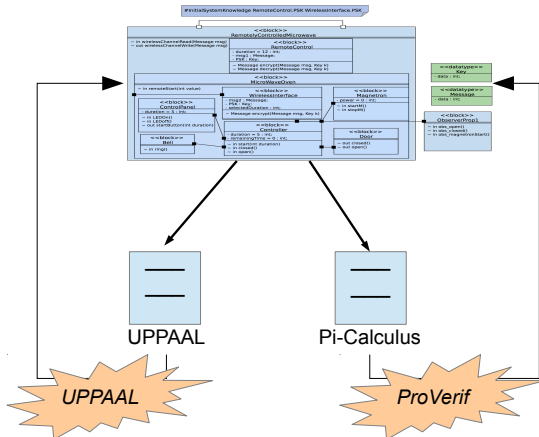


## Security properties

- ▶ Based on basic pragmas
  - ▶ Confidentiality of a block attribute
  - ▶ Authenticity of interconnected block signals

```
#Confidentiality RemoteControl.duration
#Authenticity RemoteControl.SendingRemoteOrder.msg1
WirelessInterface.gotWirelessOrder.msg2
```

# Model Transformation for Formal Verification



## Formal Verification

- ▶ Push button approach, both for safety and security properties!

### Safety properties

- ▶ UPPAAL based

Verify with UPPAAL: options

- Search for absence of deadlock situations
- Reachability of selected states
- Liveness of selected states
- Custom verification

Custom formulae =

- Generate simulation trace
- Show verification details

Session id on launcher=1  
Sending UPPAAL specification data

Reachability of: ObserverProp1.state0: Error  
-> property is NOT satisfied

All Done

### Security properties

- ▶ ProVerif based

Execution

- Execute ProVerif as

- Show output of ProVerif

Confidential Data:

-----  
duration

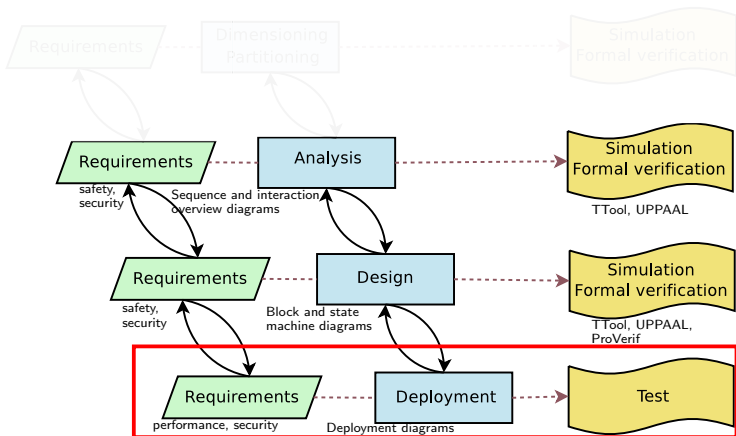
Non Confidential Data:

-----

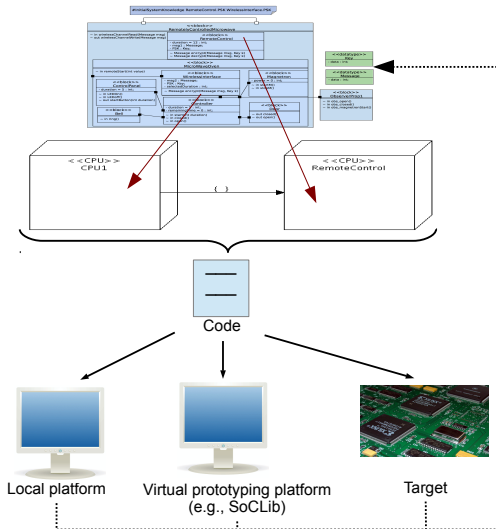
Satisfied Authenticity:

-----  
WirelessInterface\_gotWirelessOrder\_msg2\_data

# Deployment



## Deployment (Cont.)

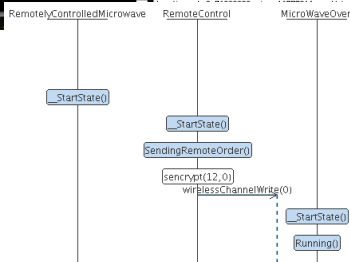


# Prototyping with SoCLib

```
vcity
BT> No request selected -> looking for one!
BT> Counting requests
BT> Starting loop
BT> Send sync
BT> Send sync not executable
BT> Counting requests: 0
BT> No pending requests
BT> Adding pending request in outMaitqueue
BT - Controller -> Waiting for request!
BT - Controller -> Releasing mutex
mj name = Magnetron
BT> Trace function
BT - Magnetron -> -> (====) Entering state + WaitForStart
BT> Trace function
BT - Magnetron -> Locking mutex
BT - Magnetron -> Mutex locked
BT - Magnetron -> Going to execute request
BT> No request selected -> looking for one!
BT> Counting requests
BT> Starting loop
BT> receive sync
BT> Counting requests: 1
BT> At least one pending request: 1
BT> selectedIndex: 1
BT> Getting request at index: 1

Terminal — xterm — 88x24
SystemC 2.2.0 --- Oct 10 2009 07:49:18
Copyright (c) 1996-2006 by all Contributors
ALL RIGHTS RESERVED
Packaged for MacOS by Logic Poet: http://www.logicpoet.com

Initializing memories with 5a
caba-vgr-mutekh_kernel_tutorial SoCLib simulator for MutekH
Initializing memories with 5a
Initializing memories with 5a
[GDB] SOCLIB_GDB env variable may contain the following flag letters:
X (dont break on except), S (wait connect on except),
C (functions branch trace), Z (functions entry trace), D (gdb protocol debug),
W (dont break on watchpoints), T (exit simulation on trap), F (start frozen)
-> See http://www.soelib.fr/trac/dev/wiki/Tools/GdbServer
[GDB] listening on port 2346
Loading at 0x60000000 size 1048576: .text .rodata .except .contextdata .data .bss
Warning: locale not supported by Xlib, locale set to C
Loading at 0x002000 size 4096: nothing
Loading at 0 size 4096: nothing
Loading at 0xbfc00000 size 4096: .boot
Loading at 0xfffff800 size 128: nothing
Loading at 0x00000000 size 16777216: nothing
```





# Outline

Introduction

Synthetic overview of contributions

Focus on a few contributions

**Conclusions and perspectives**

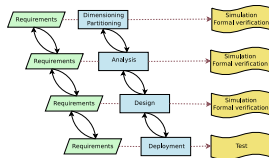
Conclusions

Perspectives

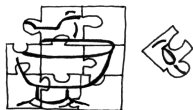


## Contributions

- ▶ Global approach for complex embedded systems based on Model Driven Engineering techniques
  - ▶ Safety-oriented Dimensioning, Analysis, Design, Deployment: TURTLE
  - ▶ Partitioning: DIPLODOCUS
  - ▶ Requirements and properties: TEPE
  - ▶ Safety and security: AVATAR
- ▶ Toolkit (TTool)
- ▶ Collaboration with academic and industrial partners



## And So?



Embedded System

=

*Tightly coupled hardware and software integration*

MDE is still a software-centric approach!

- ▶ Definitely not tightly coupled hardware and software methodology!
- ▶ Hardware is seen like a resource, i.e. it has no behaviour
- ▶ Same remark applies to intermediate layers (e.g., OS, middleware)

# Hardware in MDE

## A few techniques to handle hardware

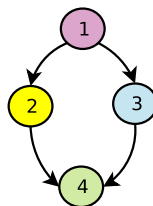
- ▶ UML Deployment diagrams
- ▶ SysML / MARTE allocation mechanisms
- ▶ DIPLODOCUS mapping
- ▶ Platform Independent Model, Platform Specific Model



# A Graal MDE

## Methodological steps

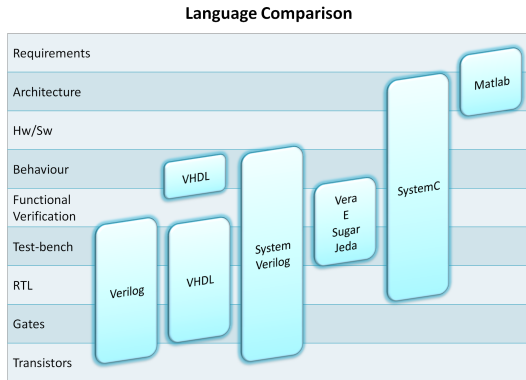
1. System partitioning
2. Software-centric model-based methodology
  - ▶ Including "low level" layers, e.g. drivers, OS, middleware
3. Hardware-centric model-based methodology
4. Model-based hardware and software integration



MDHSE: Model Driven Hardware and Software Engineering  
Meta-modeling for describing languages for all stages

## MDE for Hardware Design: A Few Issues

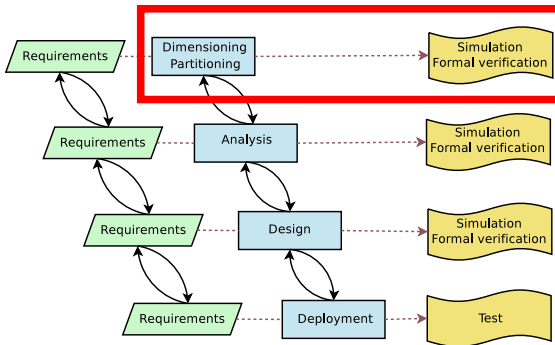
- ▶ Abstraction levels?
  - ▶ Software: PIM, PSM
  - ▶ Hardware: Transaction level (TLM, CABA), RTL
- ▶ Time handling?
  - ▶ Time model of MARTE?
- ▶ Is UML adapted as a modeling language? If yes, is it adapted to all abstraction levels?
- ▶ Underlying simulation technologies, i.e., model transformation to SystemC, System Verilog, SocLib?



ESA Microelectronics

## What About Security?

- ▶ AVATAR handles security from Analysis to Deployment
- ▶ Dimensioning, partitioning are not (yet) handled



## Questions?



- ▶ <http://perso.telecom-paristech.fr/~apvrille/>
- ▶ <http://ttool.telecom-paristech.fr/>