An Interactive System Level Simulation Environment for Systems-On-Chip

Daniel Knorreck,
Ludovic Apvrille, Renaud Pacalet

daniel.knorreck@telecom-paristech.fr

Outline

- The DIPLODOCUS Profile
- IDE: TTool
- Fast and Interactive Simulation Capabilities
- Conclusions and Future Work

The DIPLODOCUS Profile

Context: Design Space Exploration

- **Definition**:

  Process of **assessing several functionally equivalent implementations** of a system with the objective to identify an optimal solution with respect to given metrics

- Metrics could be:
  - performance related (end to end delay, compliance with deadlines,…)
  - power/energy consumption
  - cost (in terms of money, silicon area, dev. time)
- Carried out at **early design stages → only high level models of system exist**

DIPLODOCUS UML Profile I

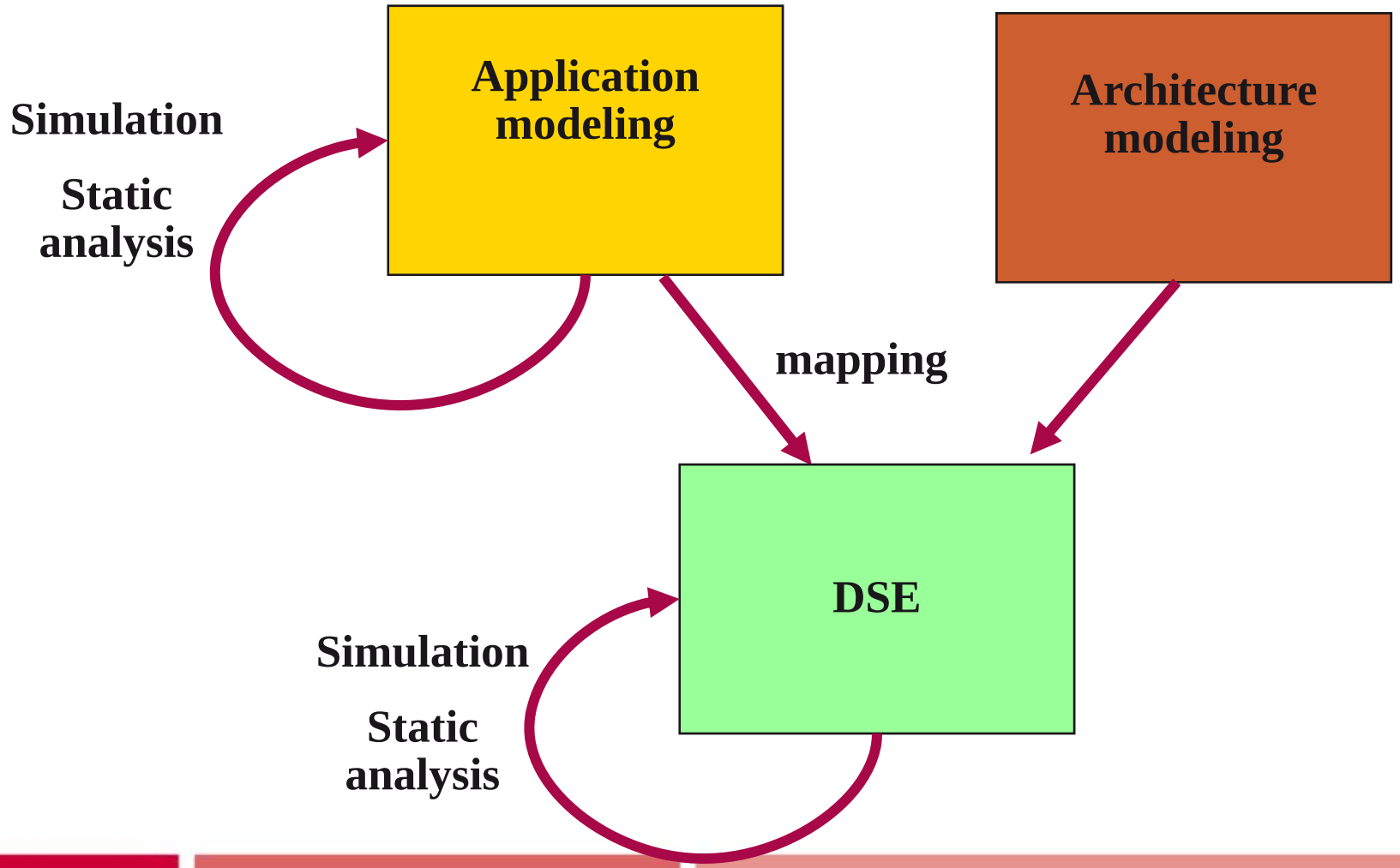
- Intended for **High Level Modeling** of **Systems-On-Chip**
- Introduces **abstraction** to deal with complexity
- Comprises **formal semantics** needed for formal analysis
- Abstraction level leverages **efficient System Level Simulation**
- **Major goal: Design Space Exploration**

DIPLODOCUS UML Profile II

- **Clear separation between**
  - Application
  - Architecture
  - Mapping
- **Data abstraction**
- **Abstract control flow** representation

- **The environment is based on**
  - UML as modeling language
  - LOTOS and UPPAAL for **formal analysis**
  - SystemC/C++ for **simulation**

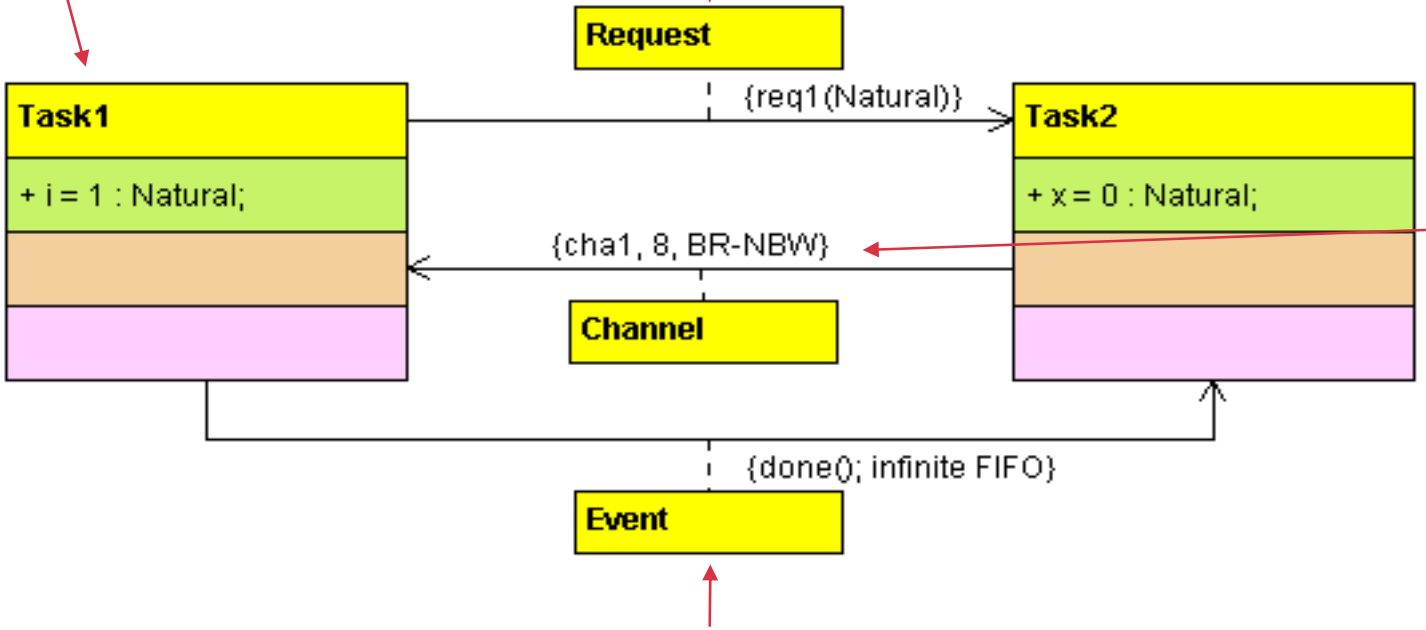
DIPLODOCUS Methodology



DIPLODOCUS: Task Diagram (App. Model)

Request. Use to spawn a task if an instance of that task is not currently executing.

Declaration of a task



Channel. Do not convey values: they are meant to model a number of exchanged samples.

Three channel types:

BR-BW: Blocking Read – Blocking write (Finite FIFO)

BR-NBW: Blocking Read – Non Blocking Write (Infinite FIFO)

NBR-NBW: Non Blocking Read - Non Blocking Write (= Shared Memory)

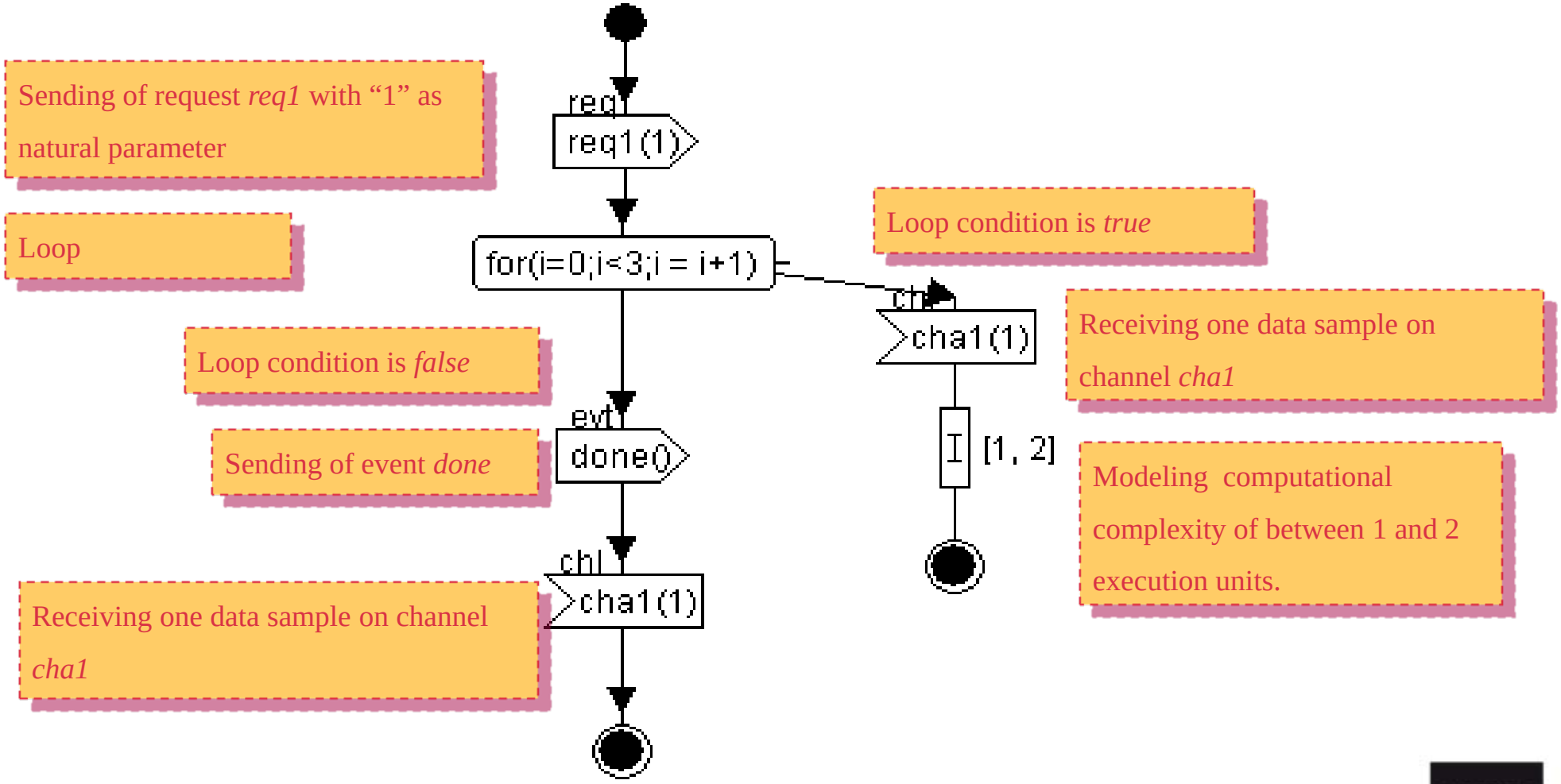
Event. Used for inter-task signaling.

Type may be **infinite FIFO** *or* **finite FIFO**. When a finite FIFO is full, the older event is erased. Events may carry values.

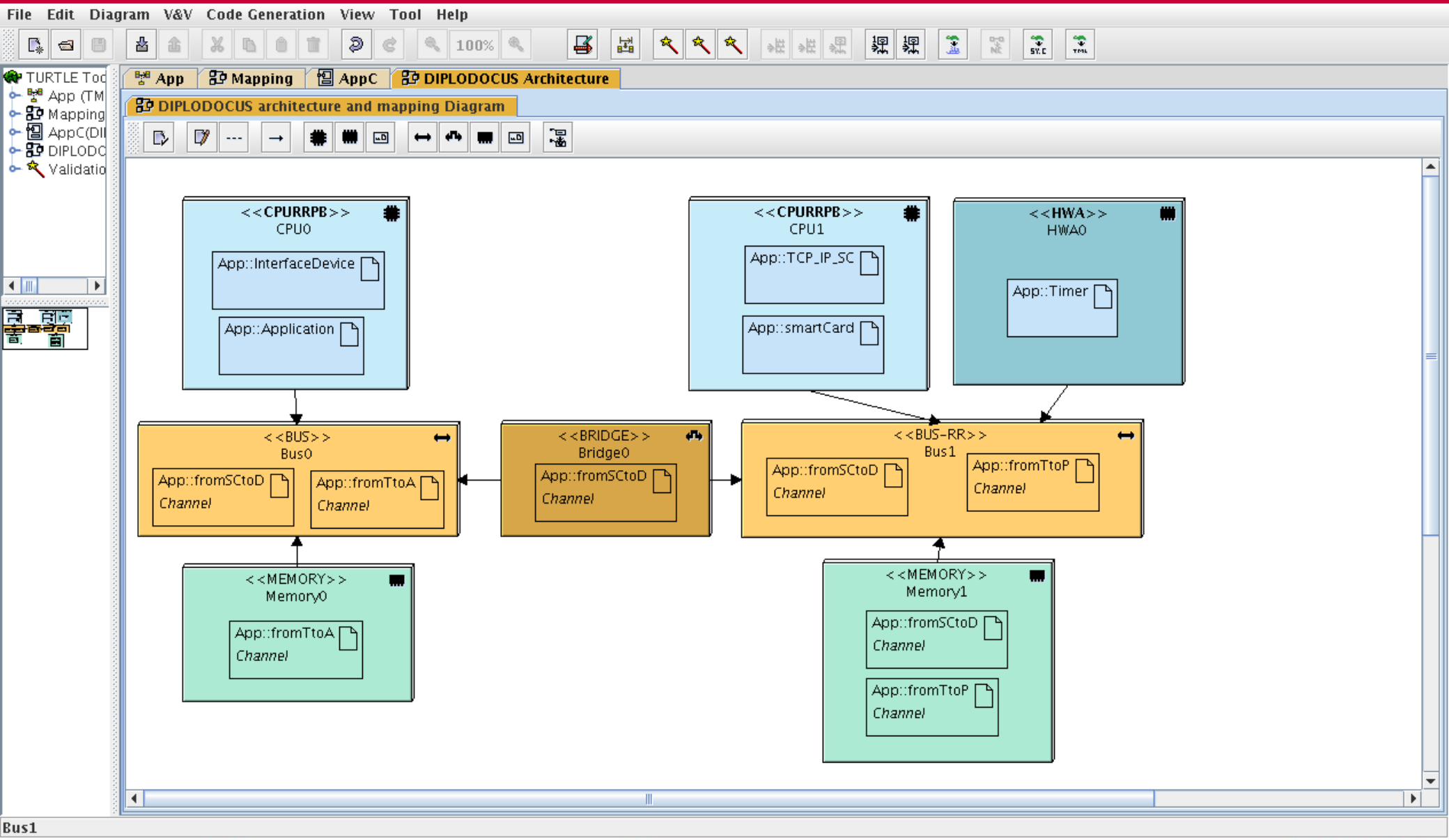
DIPLODOCUS: Task Behavior (App. Model)

- **A behavior must be provided for each task**
  - **UML activity diagram**
  - Usual **control operators**
    - Loops
    - Choices
  - **Channel operators**
    - Write x samples in a channel
    - Read x samples from a channel
  - **Events operators**
    - Send, receive an event
    - Test whether an event may be received
    - Select between events
  - **Requests**
    - Send a request

DIPLODOCUS: Task Behavior (App. Model)



DIPLODOCUS: Mapping (Architecture Model)



IDE: TTool

TTool

The TURTLE Toolkit

An open source toolkit provided by

INSTITUT TELECOM

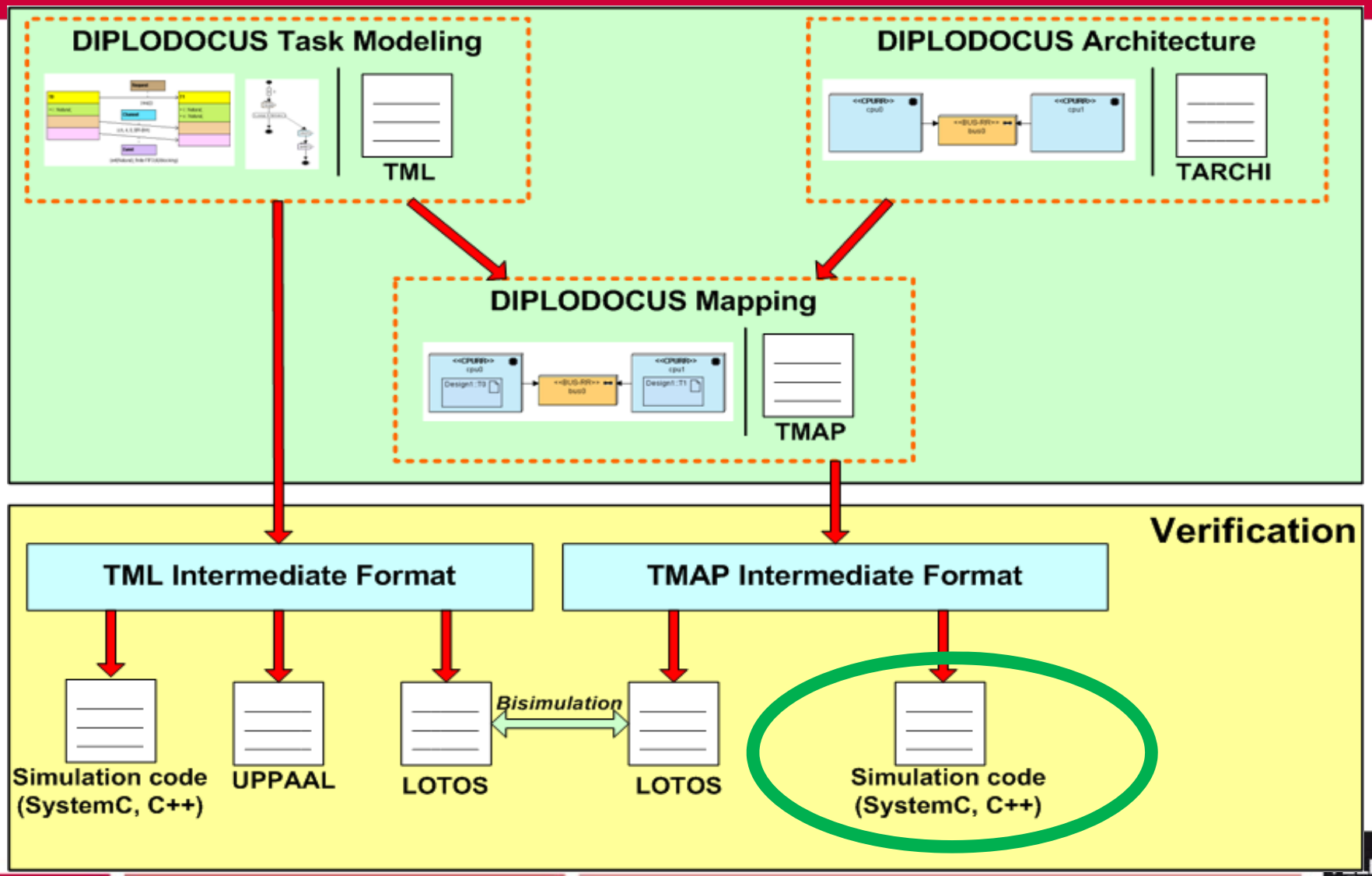
TELECOM ParisTech

TTool in a Nutshell

- **TTool enables you to**:

- **Use UML** to draw your applications, architecture and mapping
- **Verify** the **syntax** of your models
- Simply **Simulate** by executing your models without writing a single line of code
- **Perform formal proofs** at the push of a button, no expertise in temporal logics needed

Work Flow with TTool

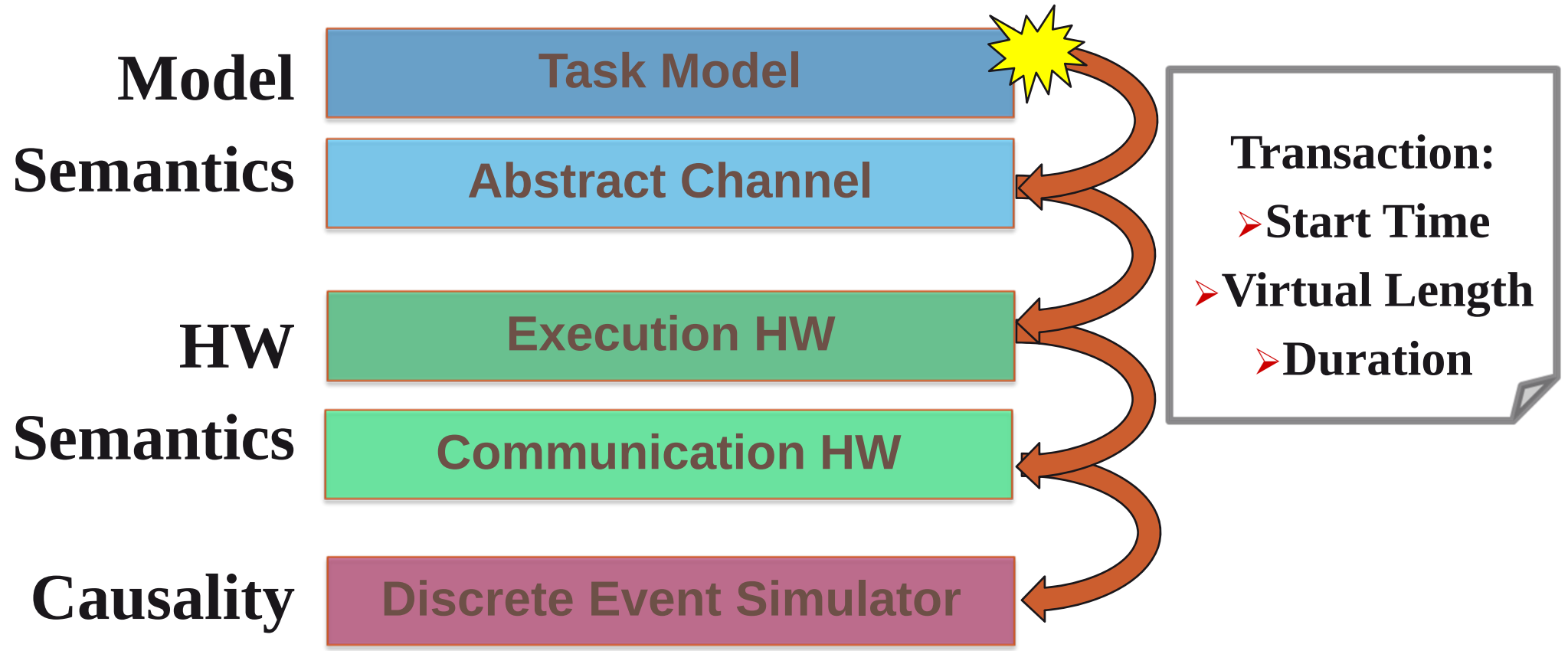


Fast and Interactive Simulation Capabilities

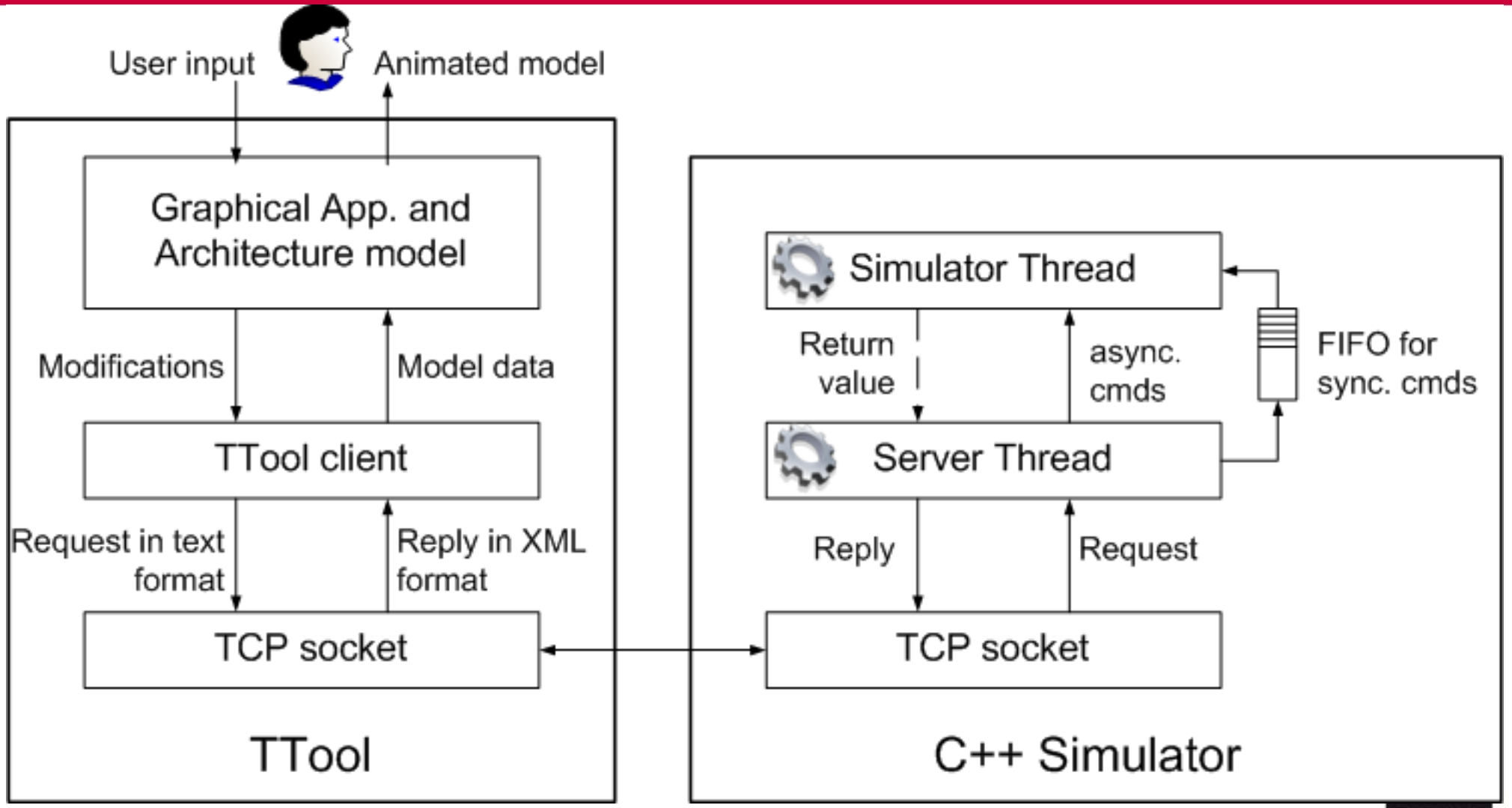
Simulation Strategy in a Nutshell

- Modeling methodology relies on both **control flow abstraction** and **data abstraction**.
- Simulation strategy should **leverage** this **high level** description **for performance** reasons.
- Coarse grained simulation strategy required based on **transactions spanning several clock cycles**.
- Thus, HW components have their own **local simulation time**.
- Synchronization of clocks is accomplished by passing transactions to involved components

Time stamp policy for Transactions



Interactive simulation environment



Simulation commands I

- **Several conditional run commands**
  - x time units, x transactions, x commands
  - Device based (CPU, Memory, Bus)
  - Application based (Task, Channel)
  - To next random choice (user may influence execution)
  - Until condition is fulfilled
- **Information retrieval about simulation**
  - Read characteristic state variables of hardware/application elements, benchmarks,…

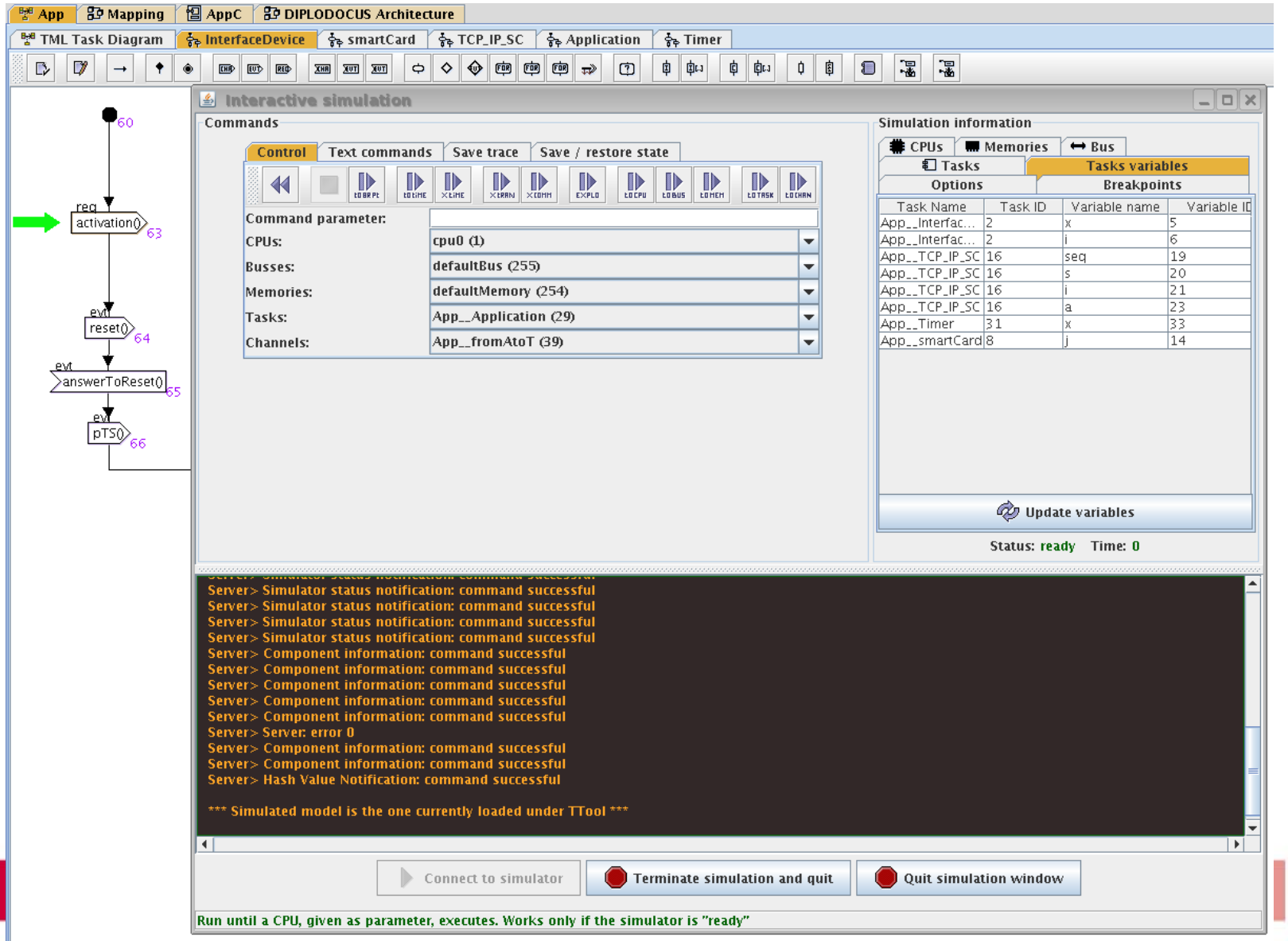
Simulation commands II

- **Manipulate application entities**
  - Write samples/events in channel
  - Set task variables
- **Generation of traces**
  - For output in HTML, VCD and text format
- **Save and Restore simulation states**
- **Break point management**
  - Add, remove, enable, disable breakpoints

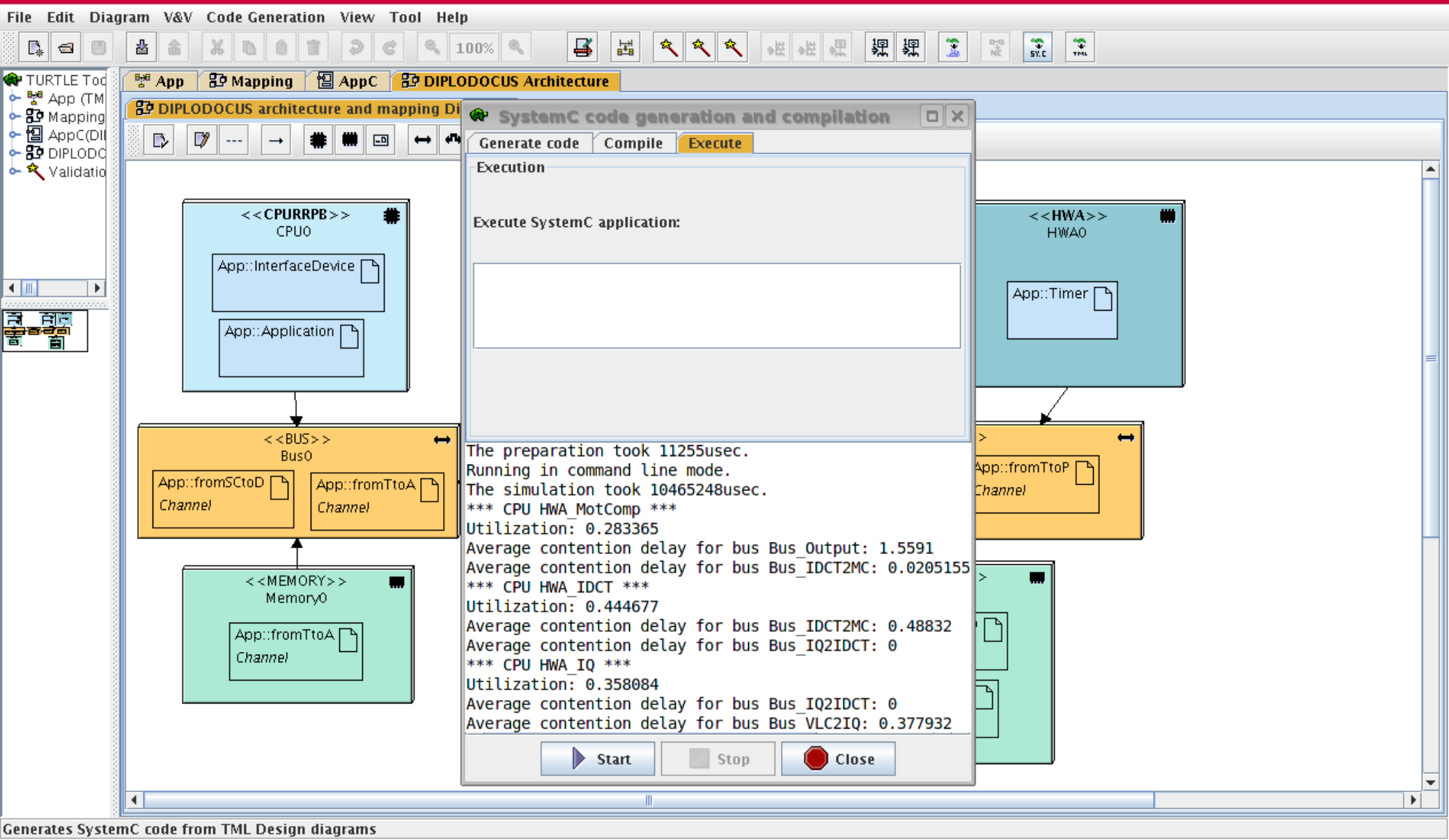
Conditional run commands

- Commands comprise a **condition expressed in terms of** local **variables** of a specific task, for example: x == y+1
- Condition is transmitted to the simulation environment
- **Simulator compiles condition**
- Code is **embedded as a shared library**
- Task commands which modify variables will invoke a condition function

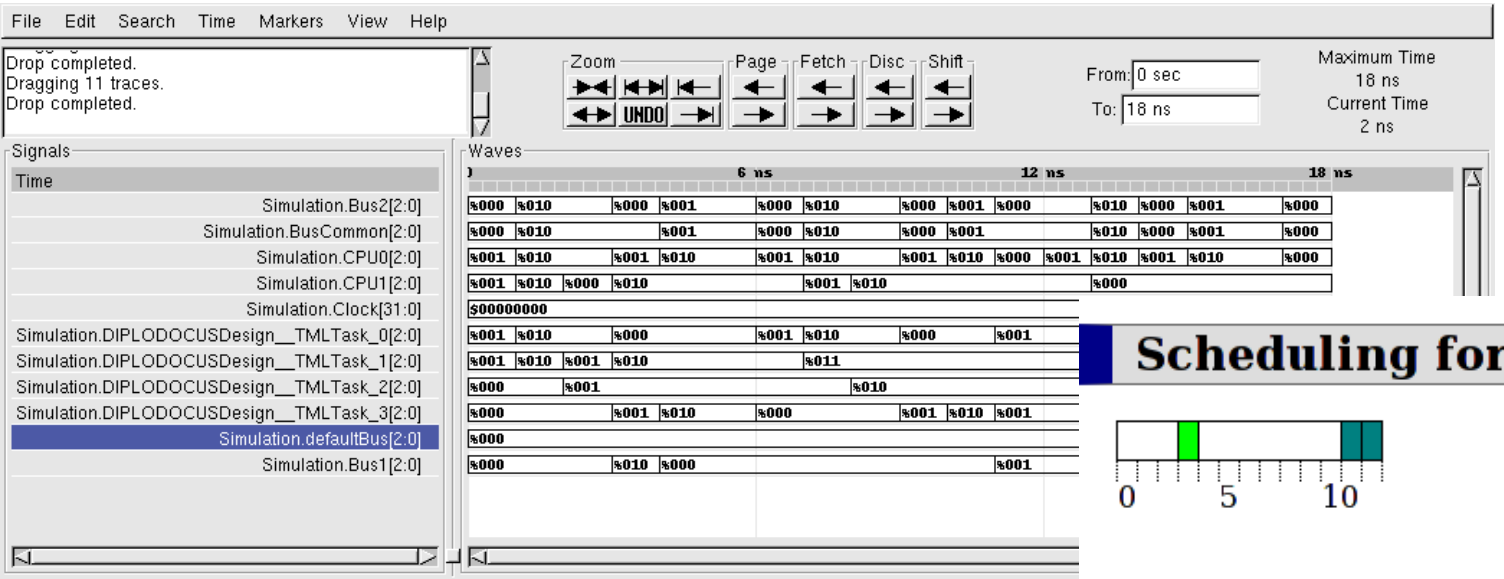
DEMO


App
Mapping
AppC
DIPLODOCUS Architecture
TML Task Diagram
InterfaceDevice
smartCard
TCP_IP_SC
Application
Timer
Interactive simulation
Commands
Control
Text commands
Save trace
Save / restore state
Command parameter:
CPUs:
cpu0 (1)
Busses:
defaultBus (255)
Memories:
defaultMemory (254)
Tasks:
App__Application (29)
Channels:
App__fromAtoT (39)
Simulation information
CPUs
Memories
Bus
Tasks
Tasks variables
Options
Breakpoints
Task Name
Task ID
Variable name
Variable ID
App__Interfac... 2 x 5
App__Interfac... 2 i 6
App__TCP_IP_SC 16 seq 19
App__TCP_IP_SC 16 s 20
App__TCP_IP_SC 16 i 21
App__TCP_IP_SC 16 a 23
App__Timer 31 x 33
App__smartCard 8 j 14
Update variables
Status: ready Time: 0
Server> Simulator status notification: command successful
Server> Simulator status notification: command successful
Server> Simulator status notification: command successful
Server> Simulator status notification: command successful
Server> Component information: command successful
Server> Component information: command successful
Server> Component information: command successful
Server> Component information: command successful
Server> Component information: command successful
Server> Server: error 0
Server> Component information: command successful
Server> Component information: command successful
Server> Hash Value Notification: command successful
*** Simulated model is the one currently loaded under TTool ***
Connect to simulator
Terminate simulation and quit
Quit simulation window
Run until a CPU, given as parameter, executes. Works only if the simulator is "ready"
60
req
activation()
63
evt
reset()
64
evt
answerToReset()
65
evt
pTS()
66

Simulation I - Outcomes

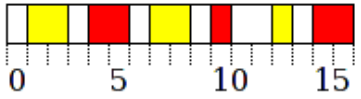


Simulation II - Outcomes

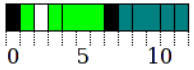


Scheduling for device: Bus1

Scheduling for device: Bus2

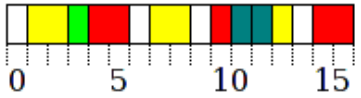


Scheduling for device: CPU1



	DIPLODOCUSDesign__TMLTask_2	DIPLODOCUSDesign__TMLTask_1
13	Action a=5	

Scheduling for device: BusCommon



Conclusions and Future Work

Conclusions

- **Implementation of a simulation environment**
  - Leverages efficiently the characteristics of the UML high level description by aggregating clock cycles
  - Interactivity allows for
    - Debugging applications
    - Accessing intermediate simulation results
    - Returning to previous system states
    - Enhancing the coverage of the simulation by exploring several possible executions

Future Work

- Verification of functional requirements during simulation (an appropriate language has already been proposed)
- Automatic and guided exploration of several alternative executions
- Tracking recurring system states of different executions
- Technical improvements of the simulator and refinement of semantics of HW nodes

Thank you for your attention!

Questions