

[www.thalesgroup.com](http://www.thalesgroup.com)

## Application of the trace driven process on a Software Radio case-study, experiments and preliminary results

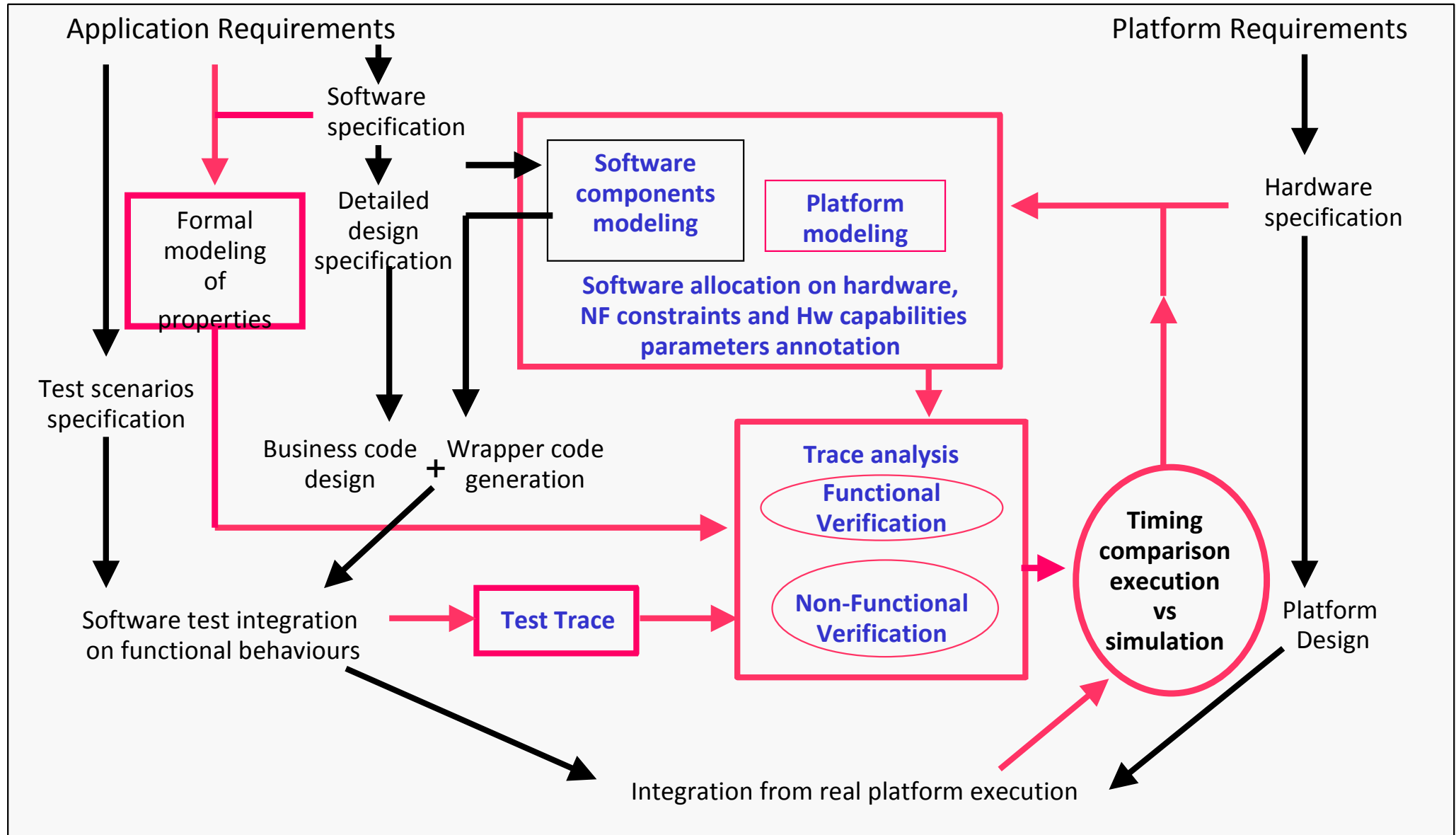
[Shuai.Li@fr.thalesgroup.com](mailto:Shuai.Li@fr.thalesgroup.com)  
Thales Communications & Security



- 1. Context and objectives**
- 2. Case-study application**
- 3. Application MARTE model**
- 4. Trace generation and analysis**
- 5. Future works**



## 1. Context and objectives



## 1. Context and objectives

### ***Functional and timing verification***

- ◆ Verify arriving order of events and their inter-dependencies
- ◆ Verify duration constraints between events

### ***Models and execution traces***

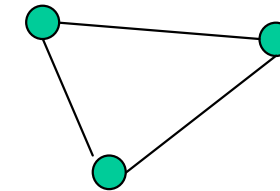
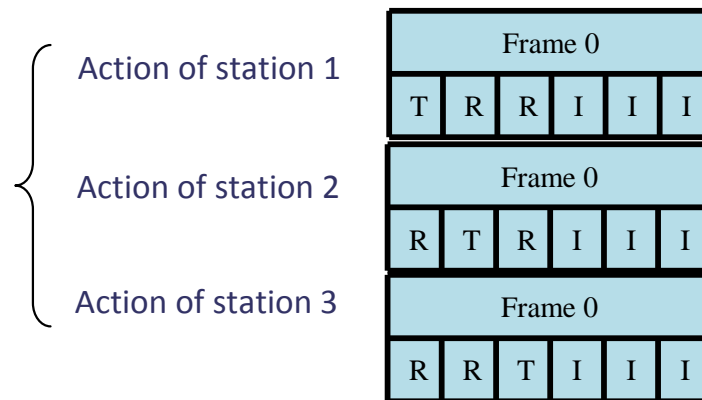
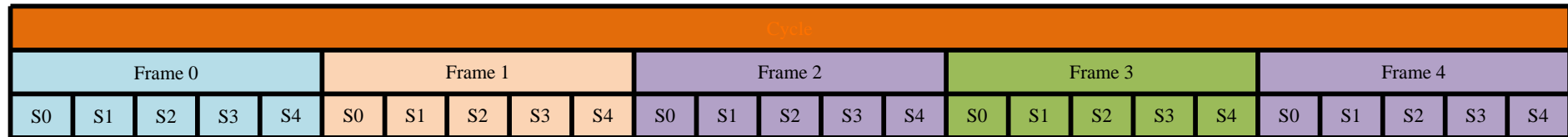
- ◆ Filter execution traces
- ◆ Compare with specifications automatically

### ***Integration and ease-of-use***

- ◆ Integrate into current design flow without modifying (too much) system engineer's habits
- ◆ Automate the process through tools development



## 2. Case-study application

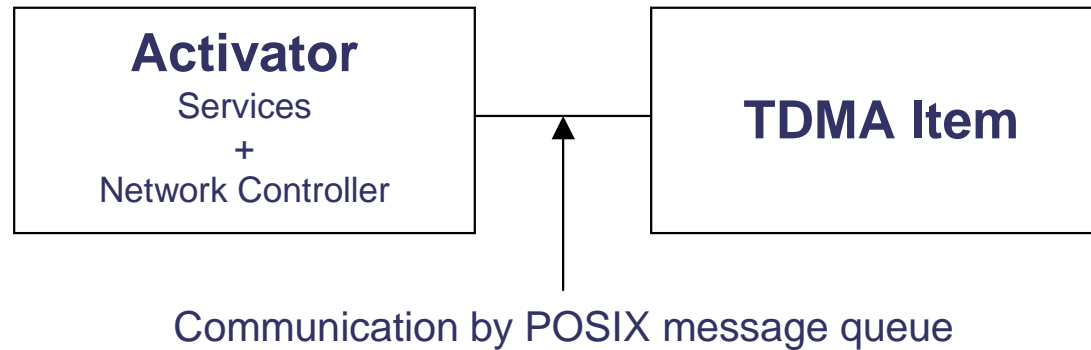
**Time-Division Multiple Access**

T: Transmission, R: Reception, I: Idle

**Internal representation**

## 2. Case-study application

### *Software tests implementation*



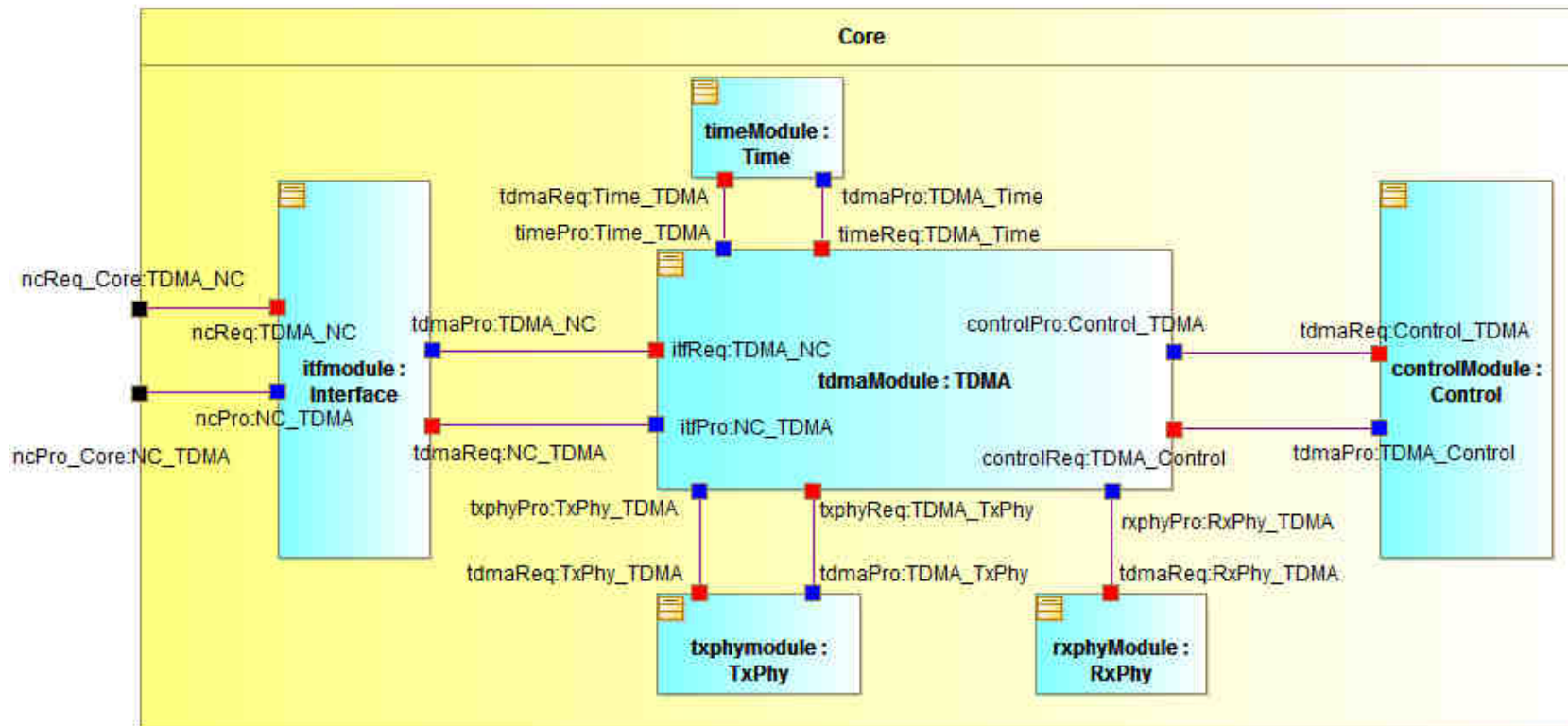
### *Execution environment*



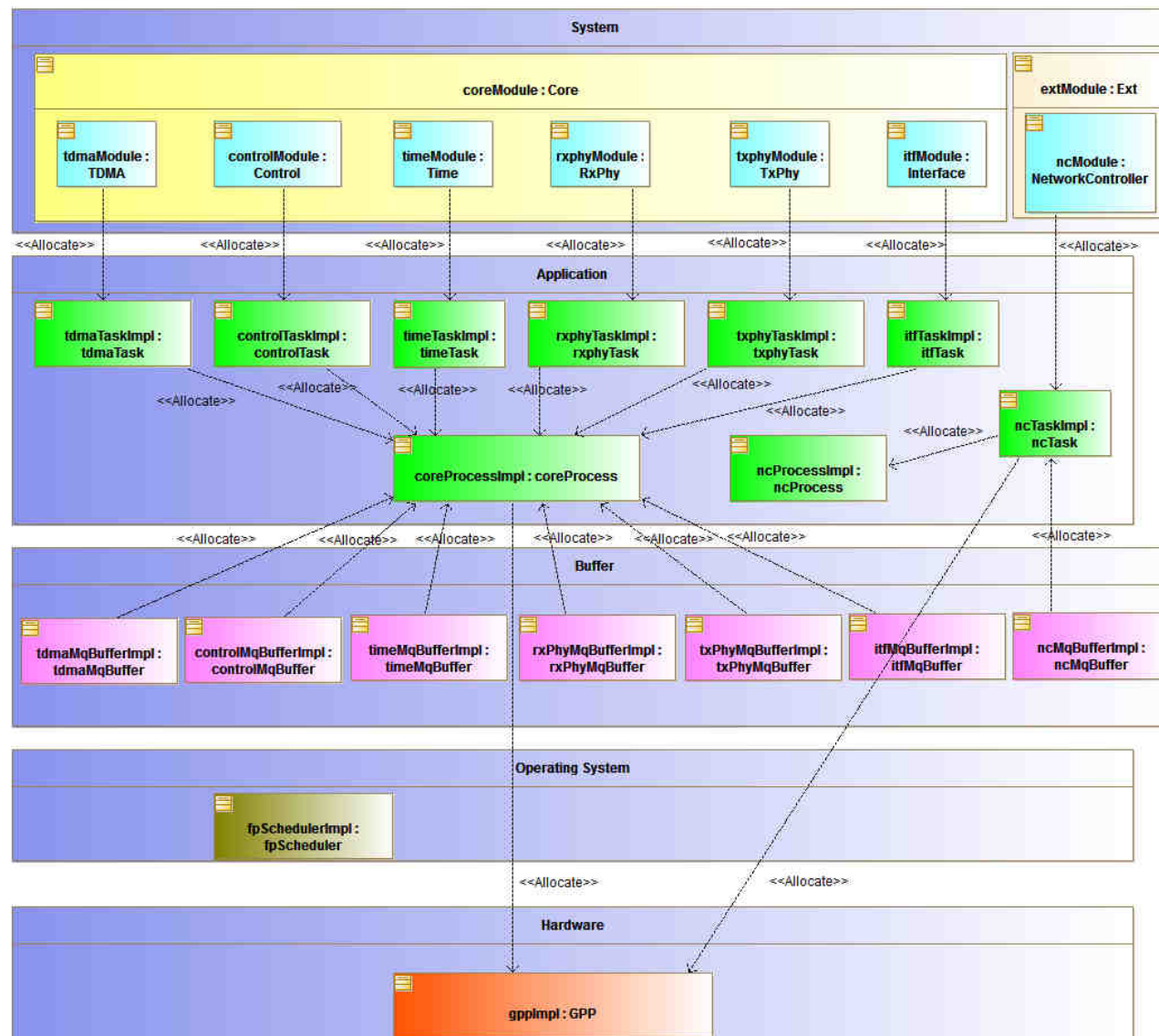
**PandaBoard (OMAP4430)**  
+  
**Linux 2.6.38.2**



### 3. Application MARTE model

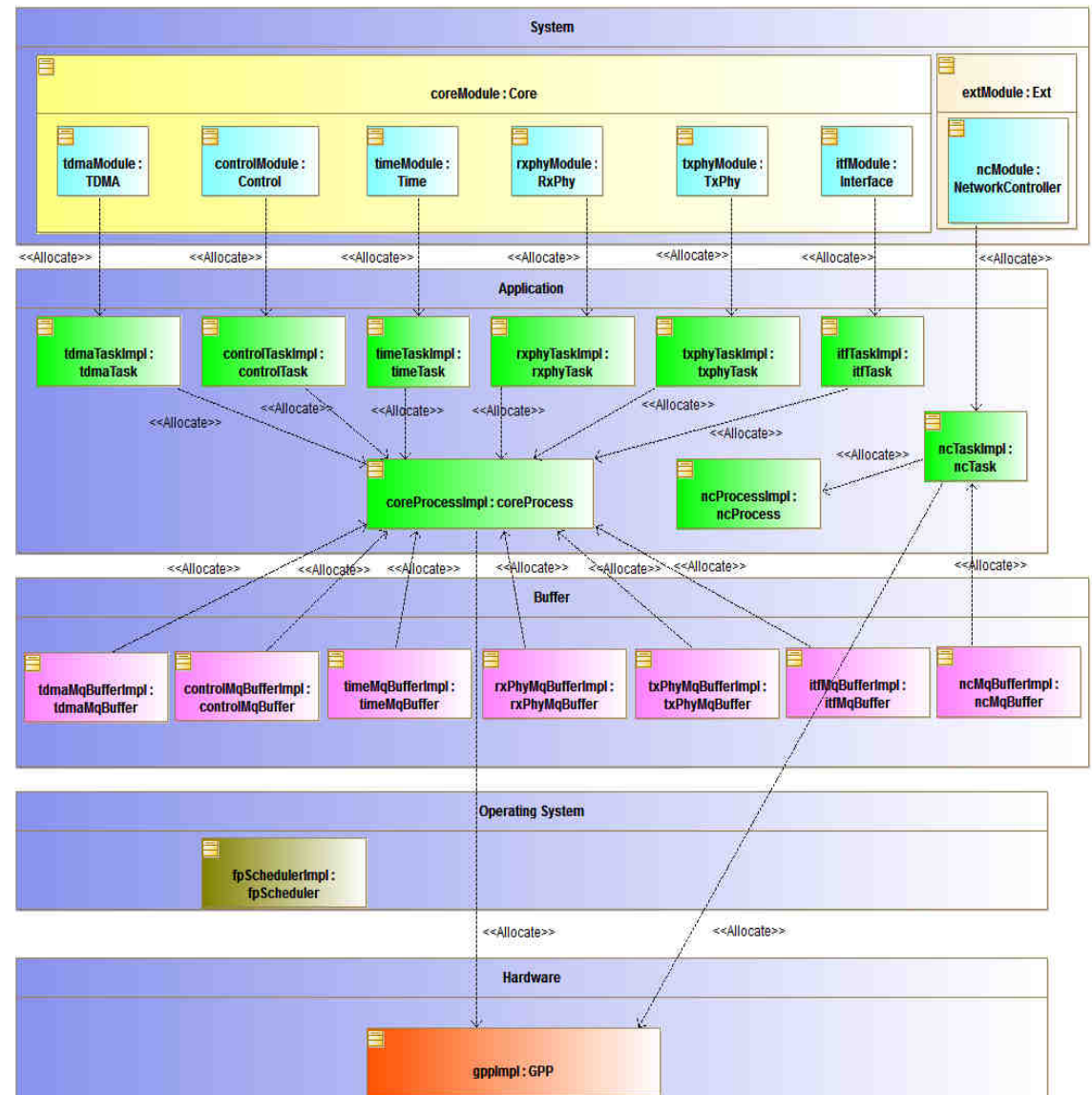


### 3. Application MARTE model

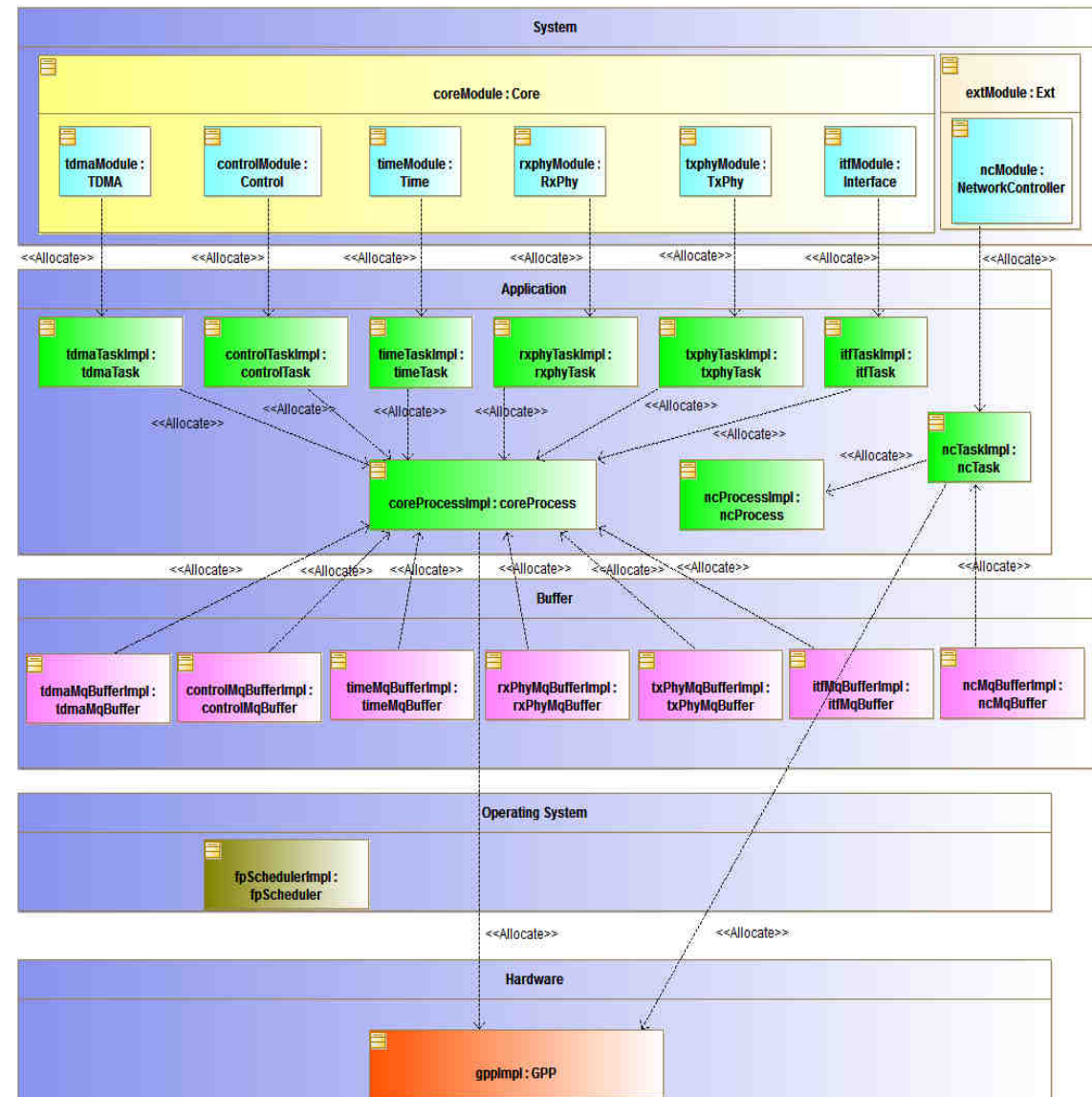
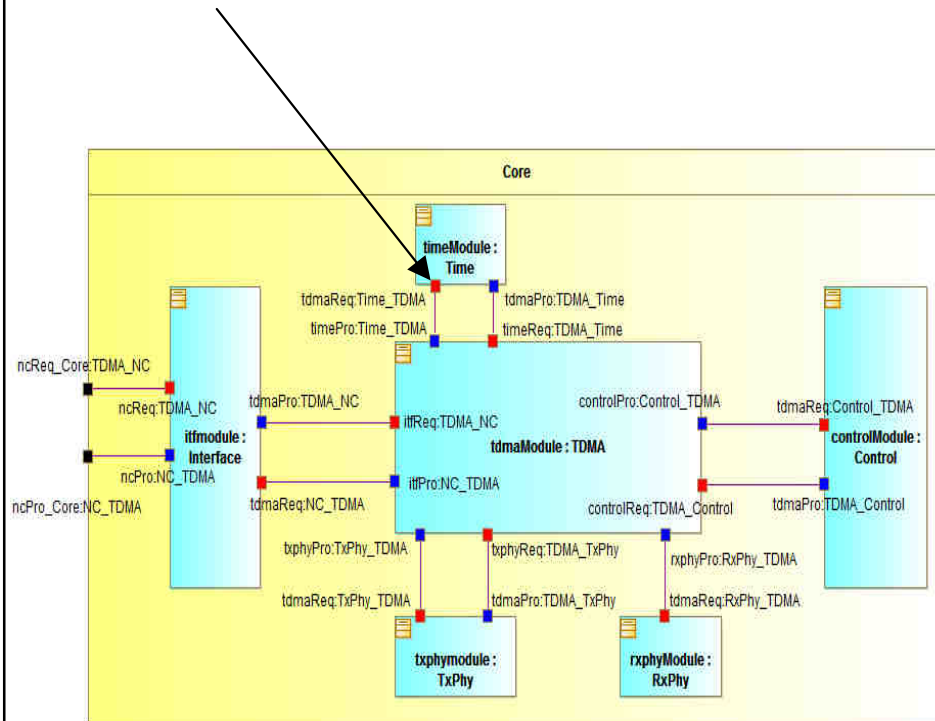




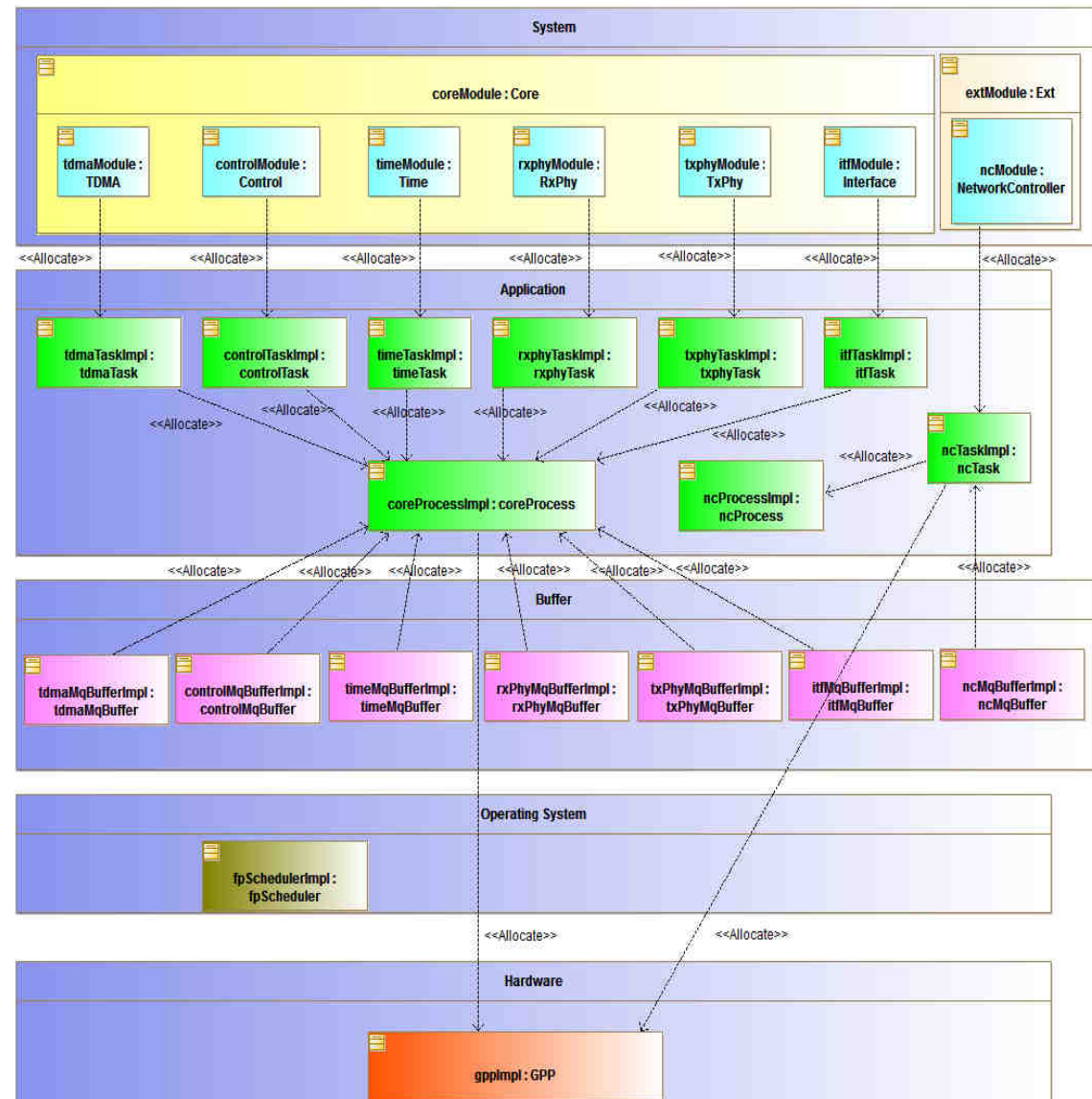
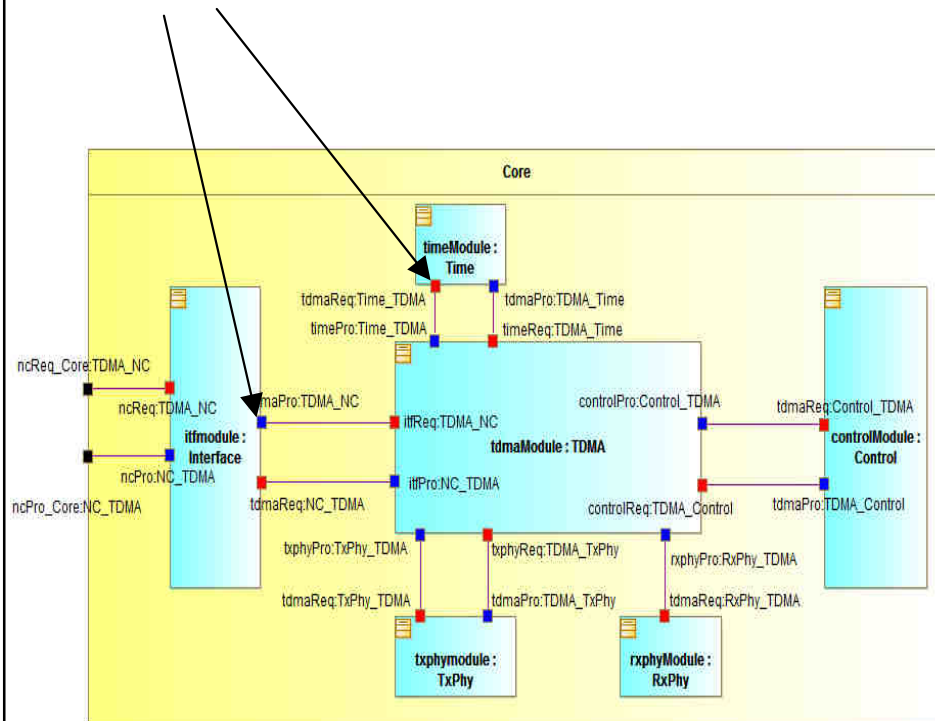
## imProvements of industrial Real time Embedded SysTems developement prOcess April 2012



### 3. Application MARTE model

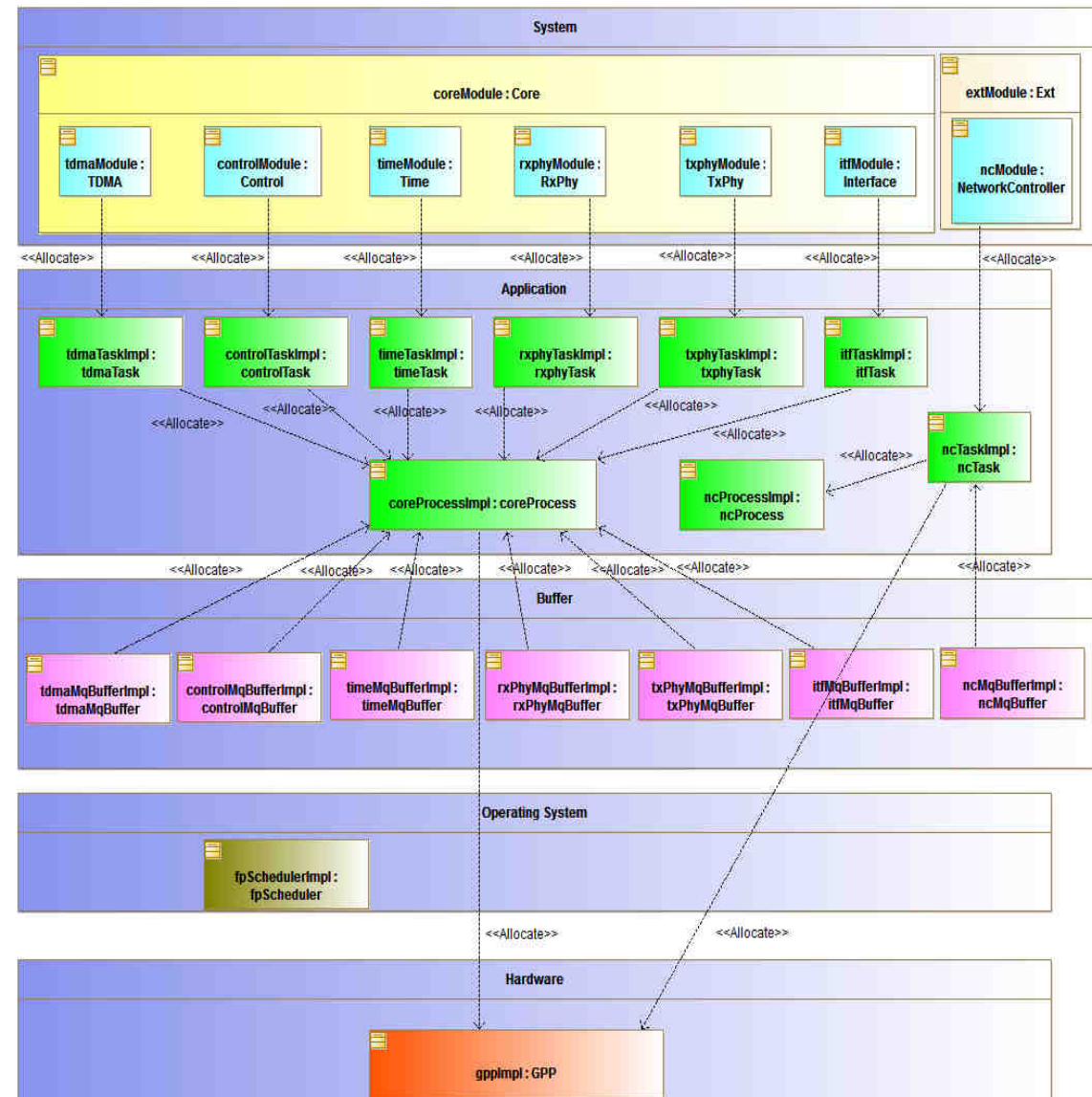
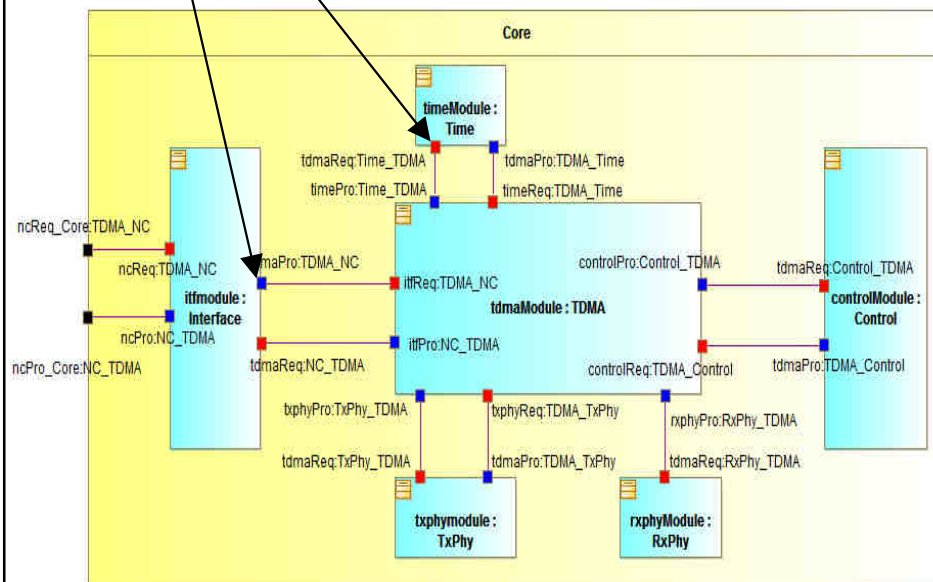


### 3. Application MARTE model



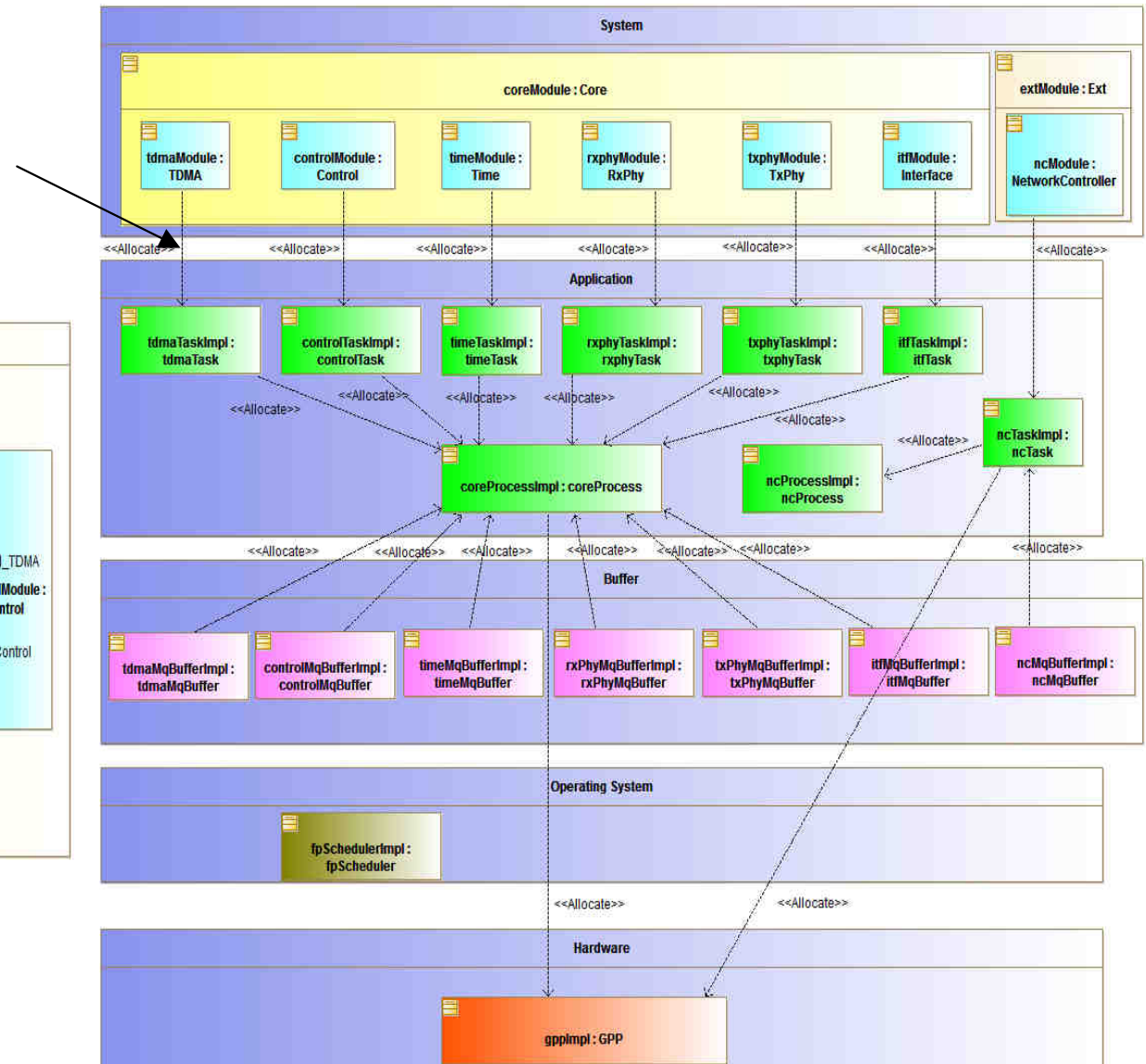
### 3. Application MARTE model

<<ClientServerPort>>

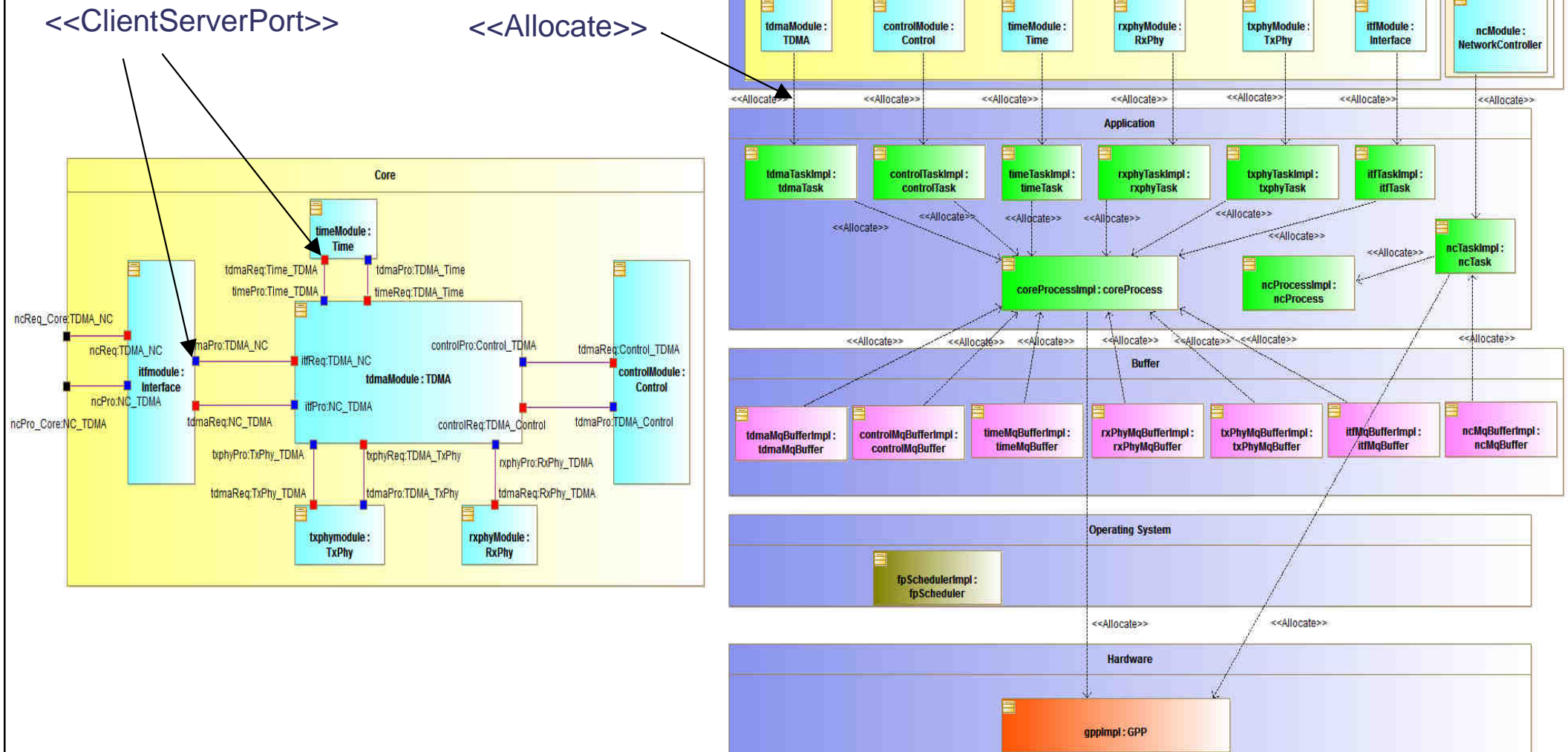




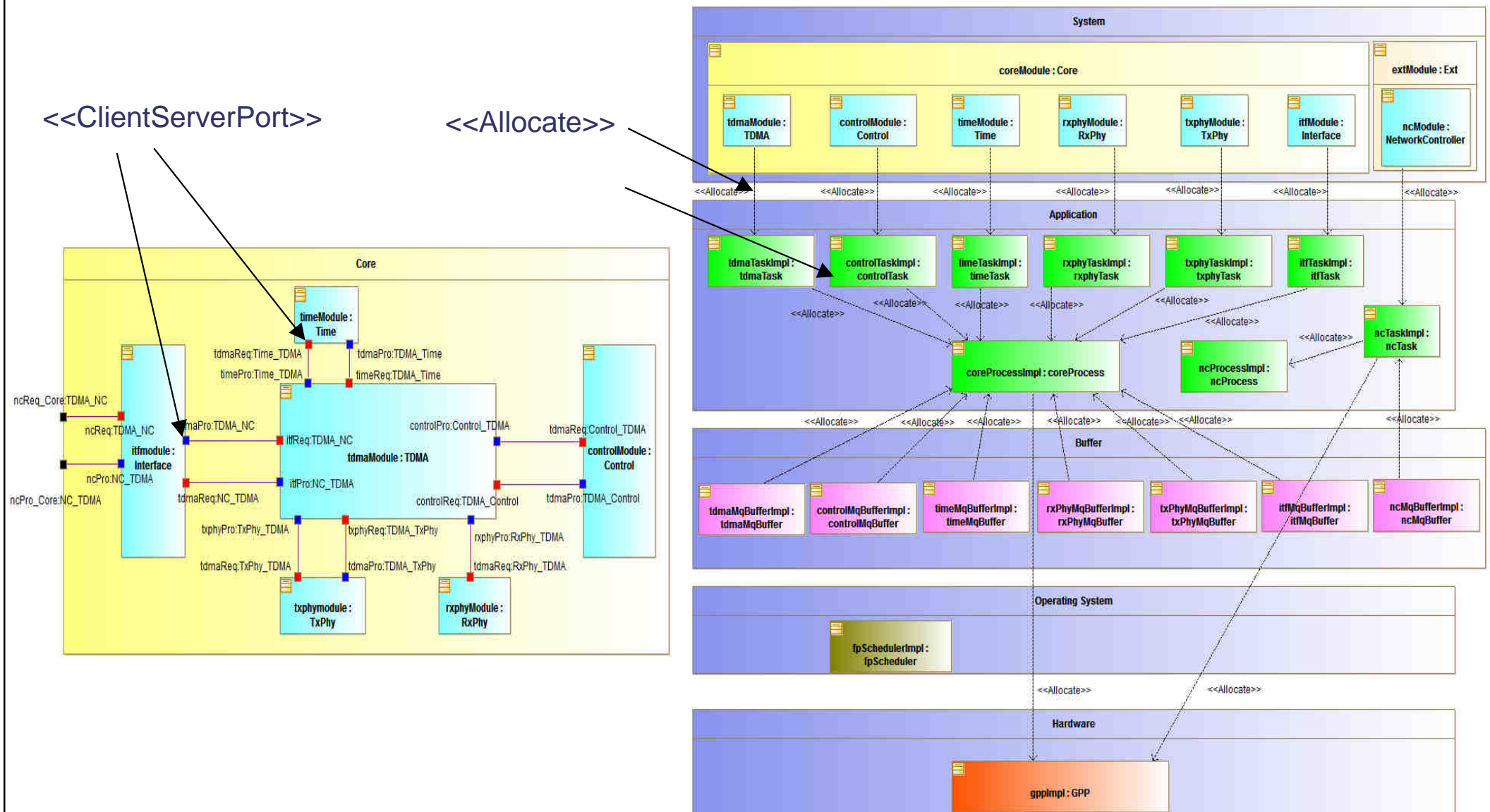
## <<ClientServerPort>>



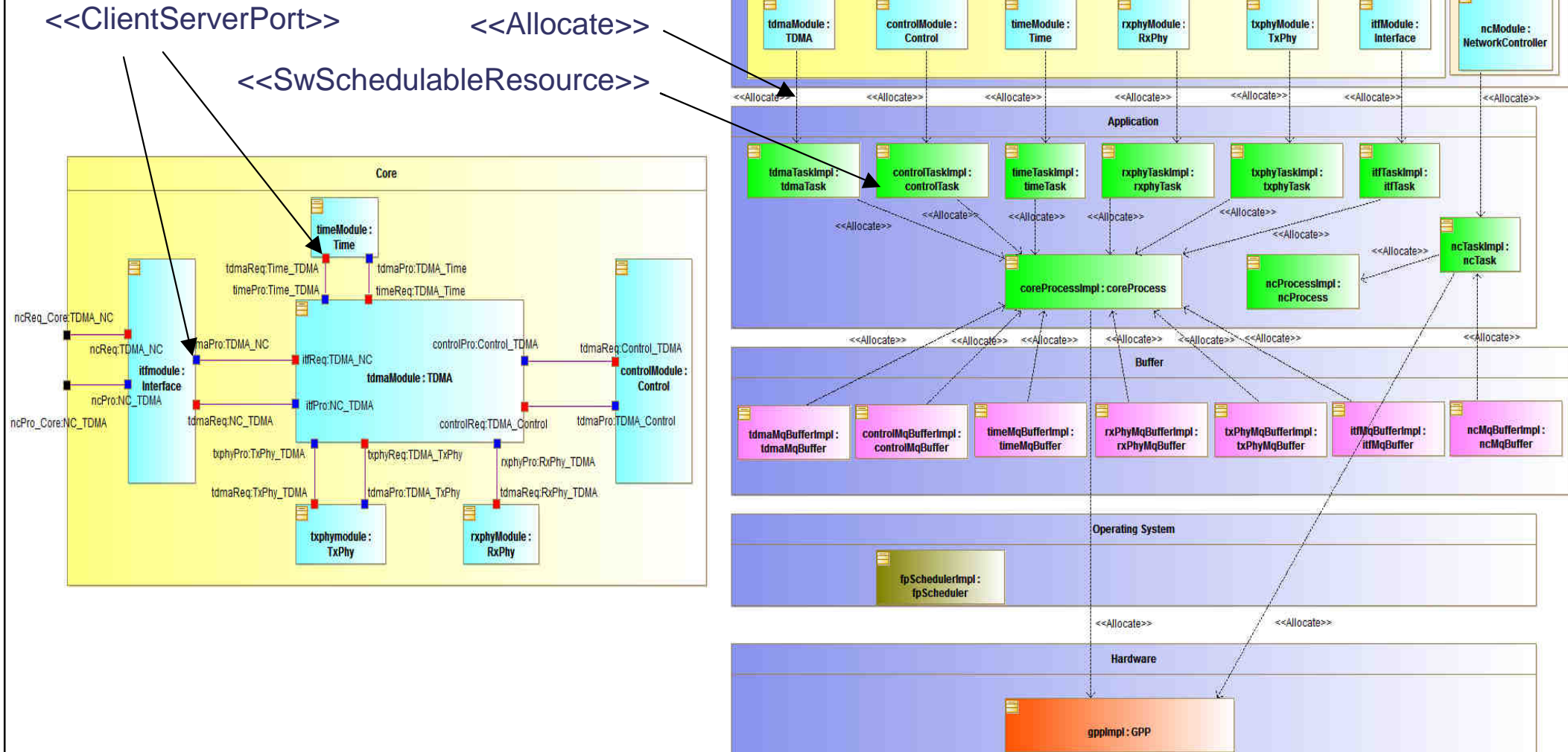
## 3. Application MARTE model



## 3. Application MARTE model

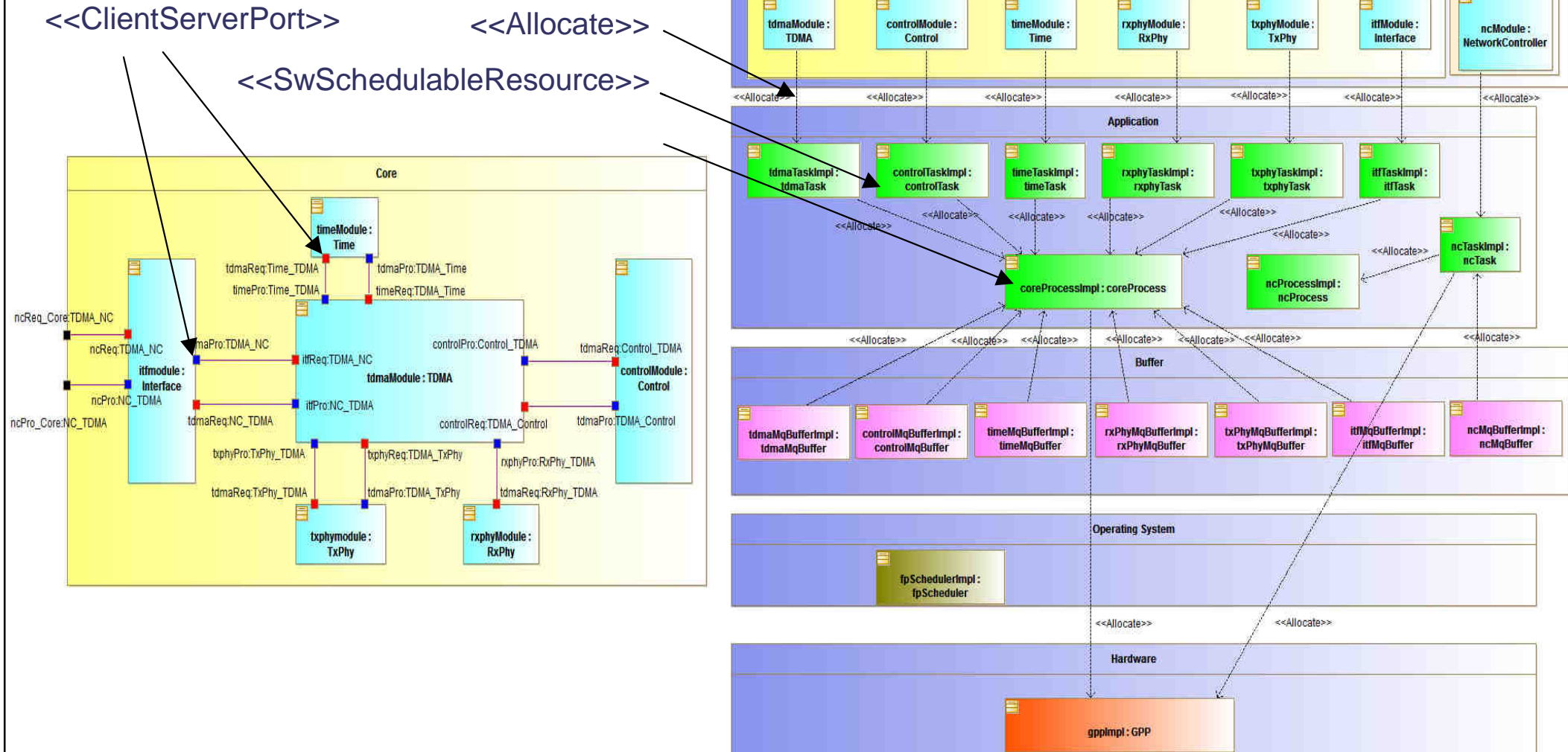


## 3. Application MARTE model

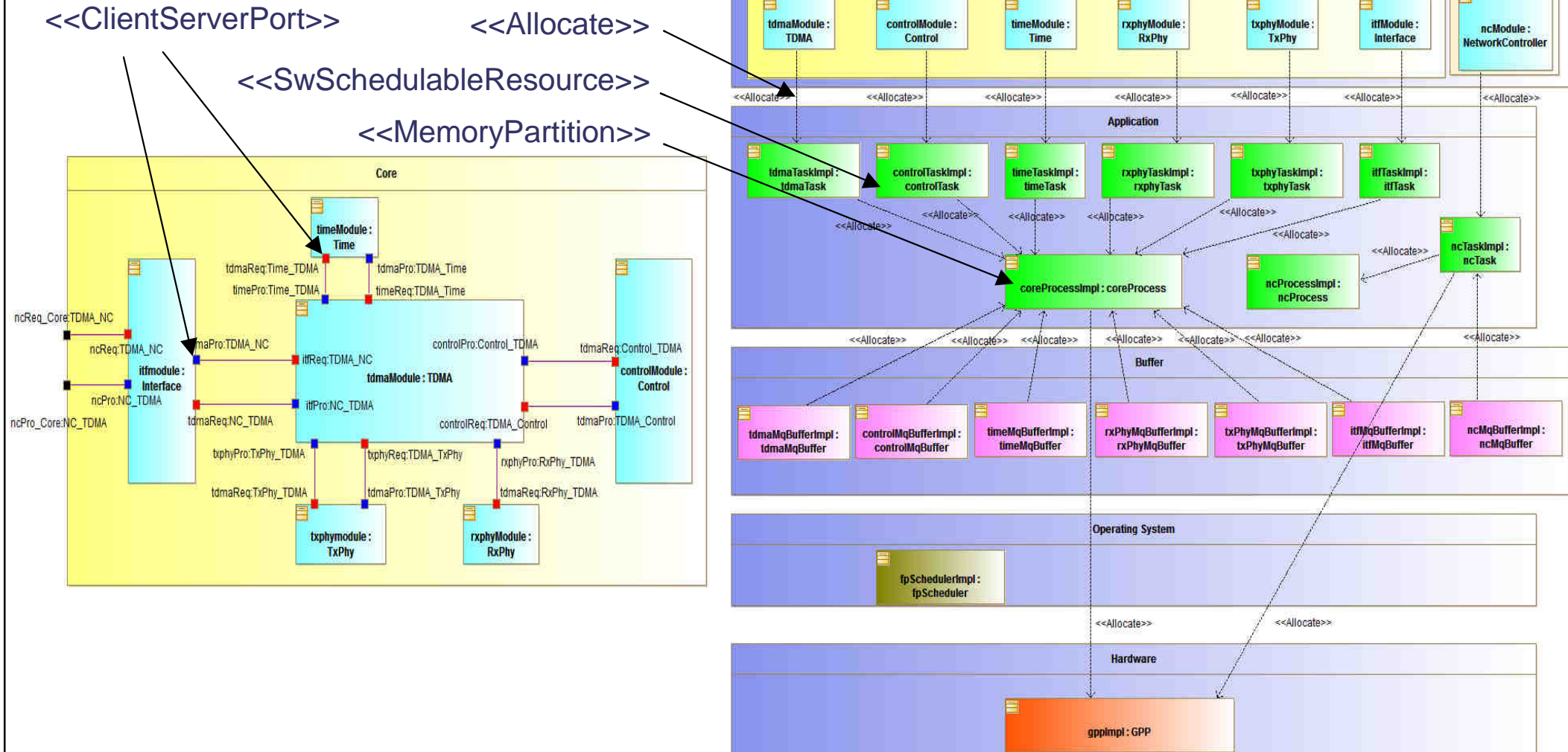




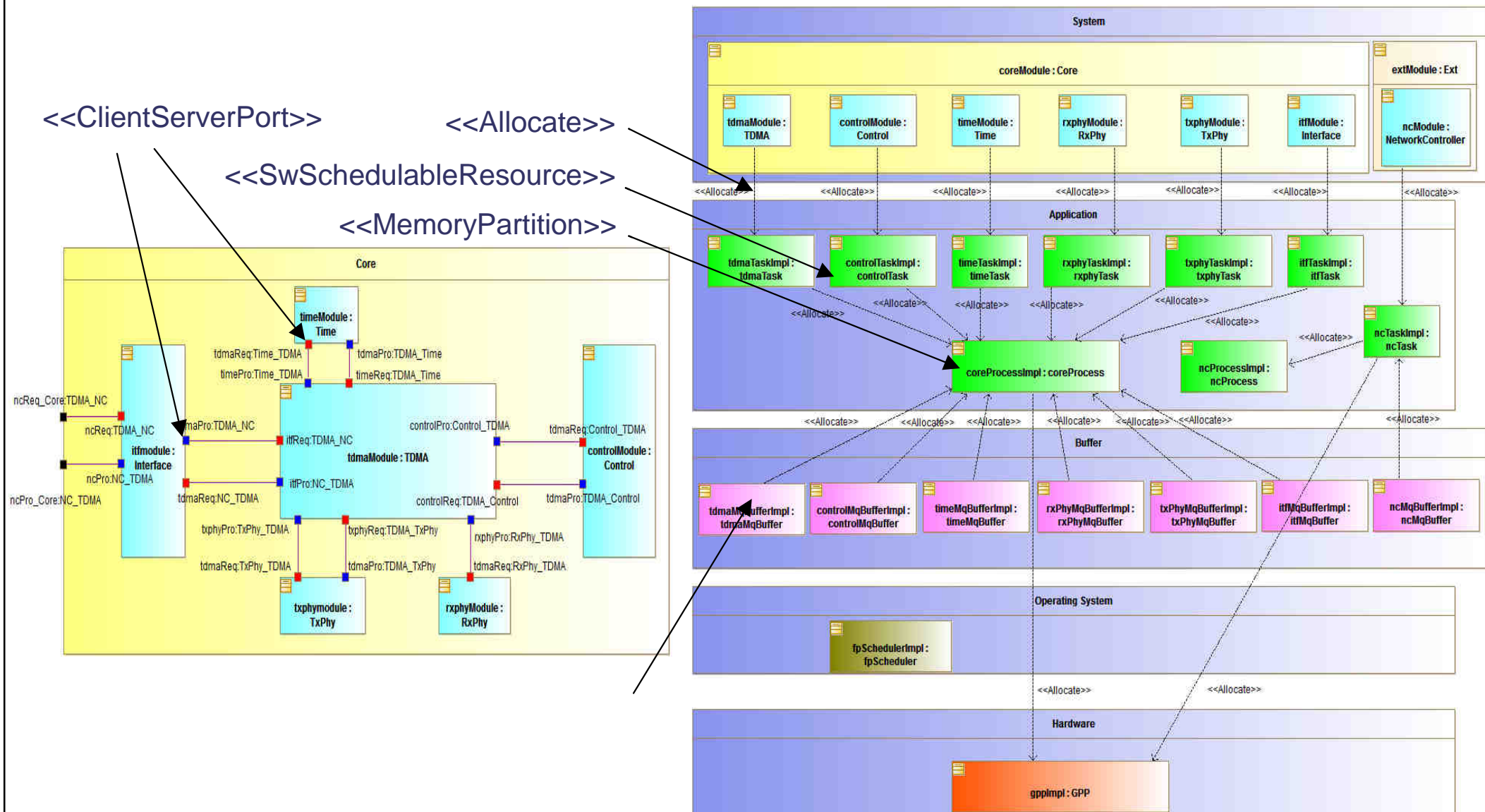
## 3. Application MARTE model



## 3. Application MARTE model

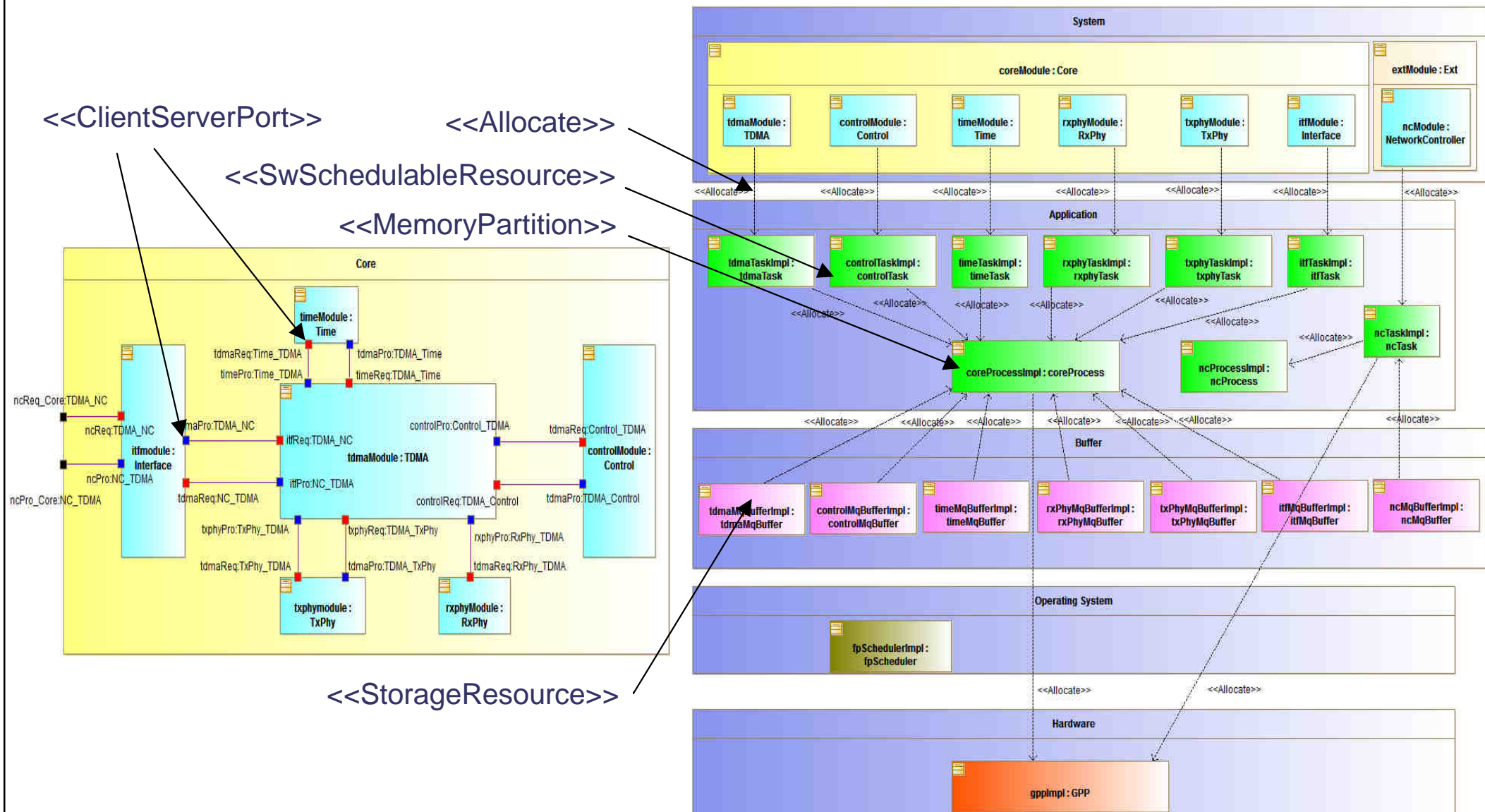


## 3. Application MARTE model

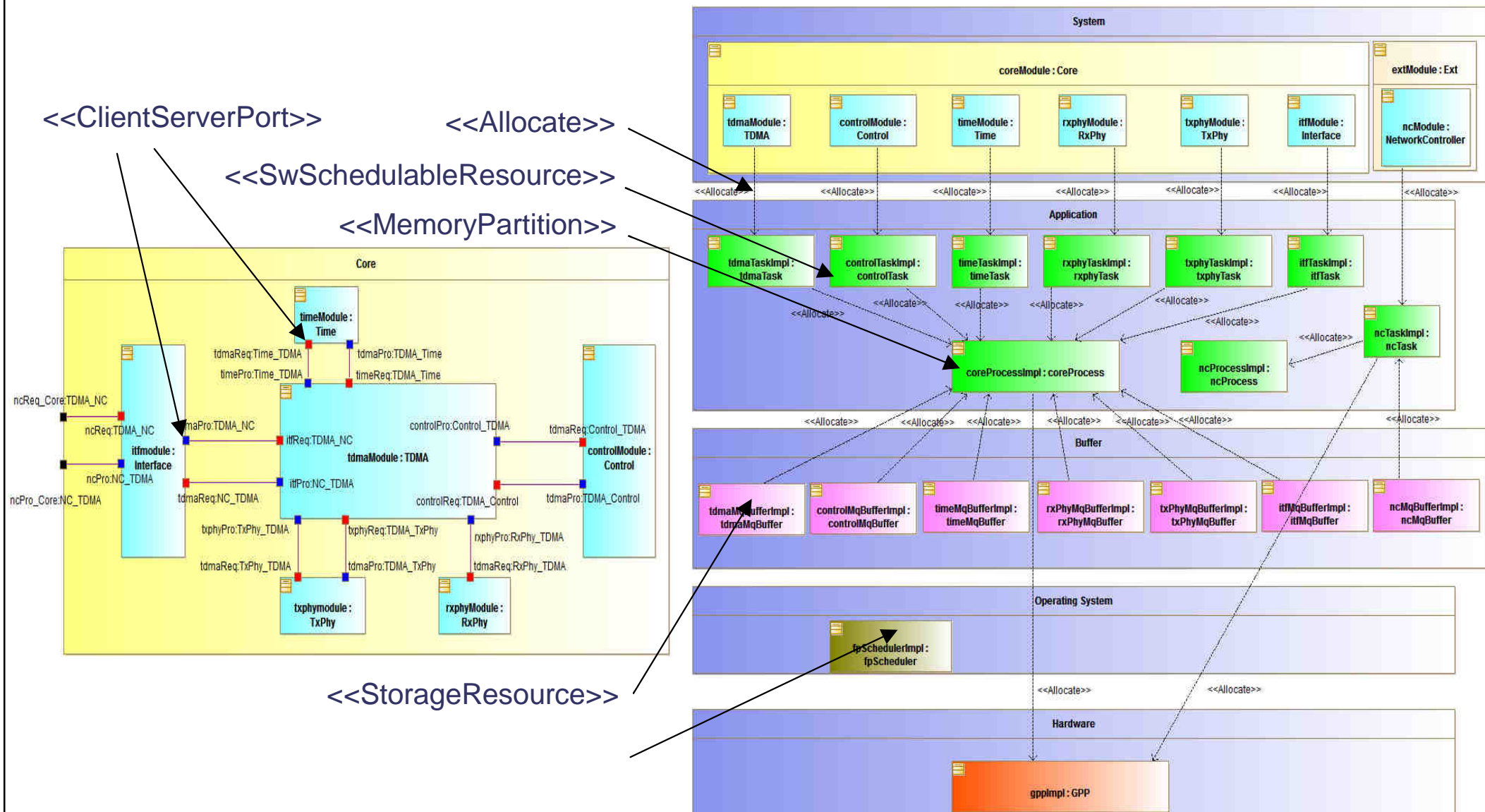




## 3. Application MARTE model



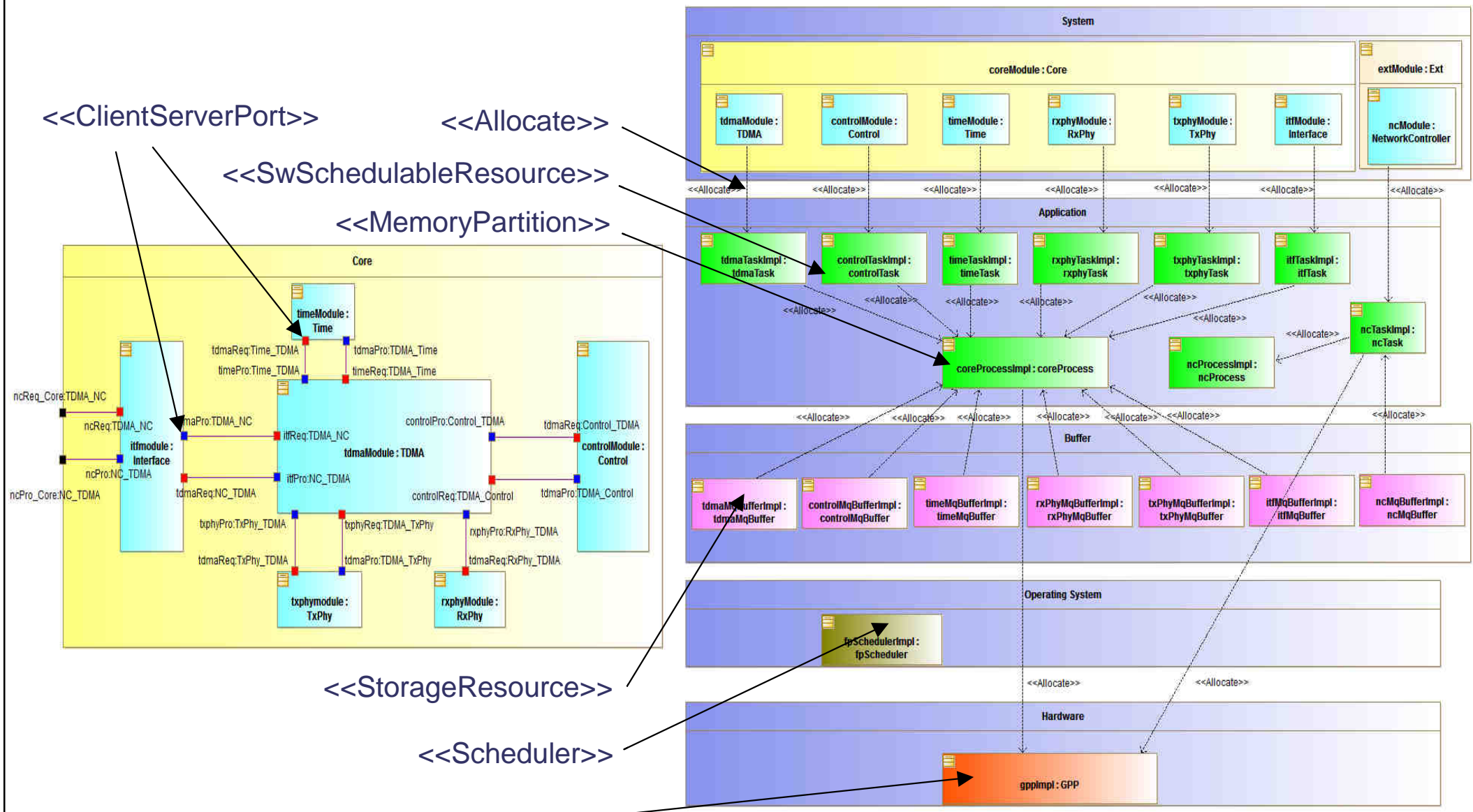
## 3. Application MARTE model



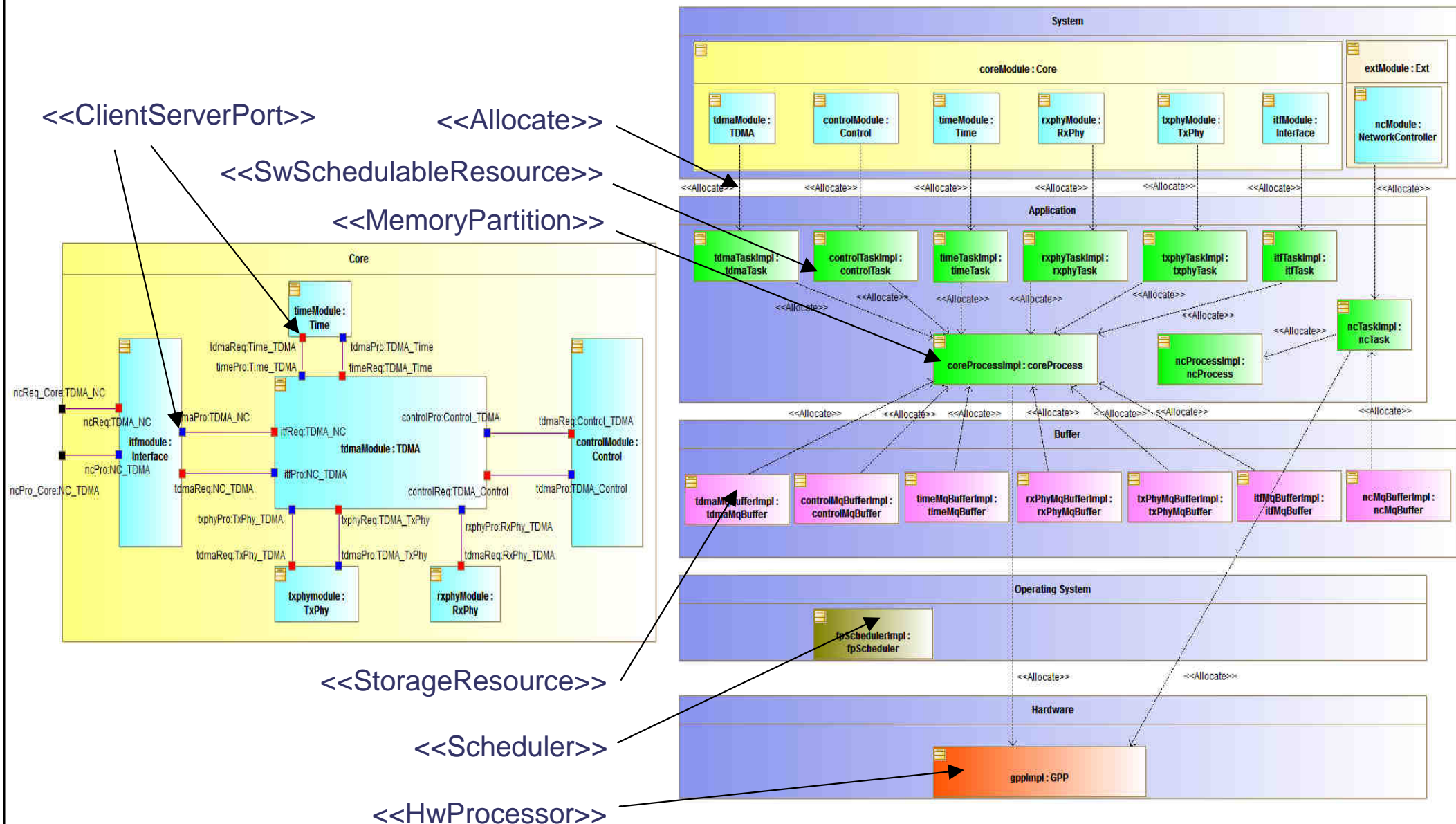
The diagram illustrates the GPP architecture across five layers: System, Application, Buffer, Operating System, and Hardware. The System layer contains the Core module, which is a <<MemoryPartition>> containing sub-modules like timeModule, txphyModule, rxphyModule, controlModule, and itfModule. The Application layer contains tasks like tdmaTask, controlTask, timeTask, rxphyTask, txphyTask, and itfTask, which are <<SwSchedulableResource>>. The Buffer layer contains buffers like tdmaMqBuffer, controlMqBuffer, timeMqBuffer, rxphyMqBuffer, txphyMqBuffer, itfMqBuffer, and ncMqBuffer, which are <<StorageResource>>. The Operating System layer contains the fpScheduler, which is a <<Scheduler>>. The Hardware layer contains the gpp. Arrows indicate data flow and allocation. Callouts on the left define stereotypes: <<ClientServerPort>>, <<SwSchedulableResource>>, <<MemoryPartition>>, <<StorageResource>>, and <<Scheduler>>.



## 3. Application MARTE model

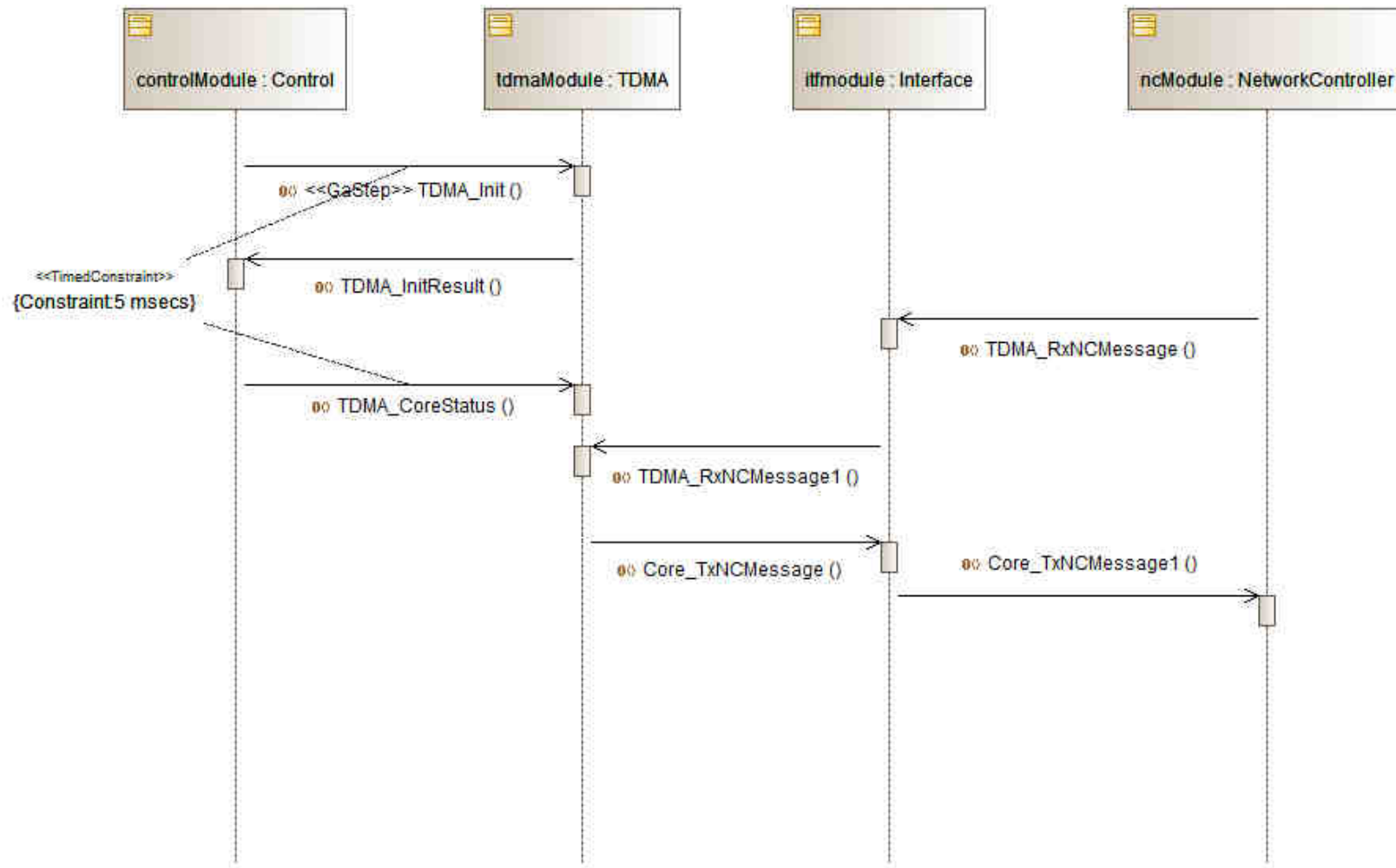


## 3. Application MARTE model





## 3. Application MARTE model



#### 4. Trace generation and analysis

##### **Specification trace**

###### ◆ Model to text transformation from sequence diagrams:

- Implemented as an Eclipse plugin
- Events, dependencies, duration constraints

##### **Execution trace**

###### ◆ Use of TAU to obtain execution traces:

- Code instrumentation:
  - PDT for code analysis
  - TAU for instrumentation and compiling
- Execution on board for raw traces
- Filter raw traces and present in a readable format
- Instant time, function start/end, thread and cpu info

Developed tool to compare execution and specification traces:

- Event order
- Duration between events



## 5. Future works

### ***Automatic instrumentation from models***

- ◆ Add new stereotypes to ports in order to add instrumentation points automatically in methods of the generated code.

### ***Data values in execution traces***

- ◆ Function parameters for PDU analysis
- ◆ Messages sent/received (e.g. structures exchanged by message queue)

### ***Traces to model***

- ◆ Use execution traces to automatically complete the model with quantitative and non-functional properties

### ***Host and target execution traces exploitation***

- ◆ Determine target timing information from host execution traces



***Thank you for you attention.***

Shuai.Li@fr.thalesgroup.com

