

# ADCSS 2014

SOFTWARE REFERENCE ARCHITECTURE – OSRA SPECIFICATION

## THE COMPONENT MODEL EDITOR PROTOTYPE (CORDET-3)

## A COMPLETE TOOL CHAIN FROM COMPONENTS TO BINARY (CORDET-2)

2014, October 27<sup>th</sup>



# CONTENTS

- ❑ Introduction.
- ❑ COrDeT-2 tool chain.
- ❑ COrDeT-3 graphical model editor.
- ❑ Conclusions.

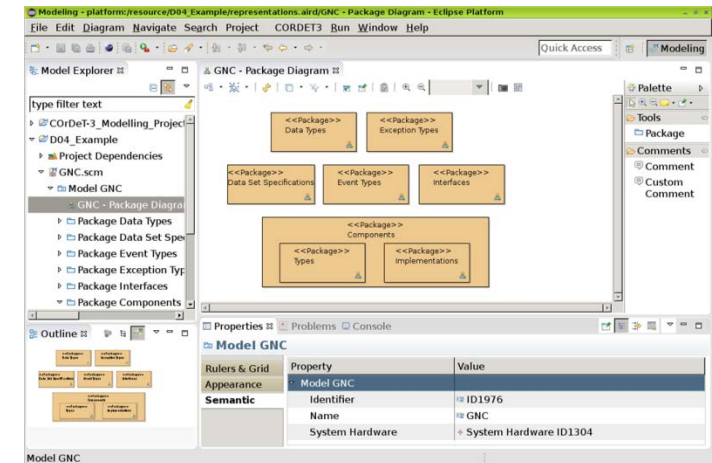
# INTRODUCTION

# CORDET-2 AND CORDET-3 STUDIES

- ❑ COrDeT studies have been supporting SAVOIR-FAIRE on the definition of the OSRA.
- ❑ In the scope of these studies two tool chains have been built up:
  - **COrDeT-2 tooling framework.**
    - Intended to demonstrate the whole OSRA process: from the design of the user model to the generation of the final executable.
    - It demonstrated that the approach is feasible.
  - **COrDeT-3 tooling framework.**
    - COrDeT-3 study is intended to provide the OSRA specification.
      - The tool supports the OSRA specification process → proof of concept.
    - COrDeT-3 tool focuses on a specific part of COrDeT-2 toolset: the design of the user model.
      - It refines the COrDeT-2 graphical editor.
      - It is based on open source tools.

# CORDET-2 & CORDET-3 TOOLS

- ❑ COrDeT-3 takes as input the COrDeT-2 Graphical Editor.
- ❑ **Capabilities:**



	COrDeT-2	COrDeT-3
<b>Capabilities</b>	<ul style="list-style-type: none"> <li>○ Graphical model editor.</li> <li>○ Verification and analyses tools.</li> <li>○ Code generation.</li> </ul>	<ul style="list-style-type: none"> <li>○ Graphical model editor.</li> </ul>

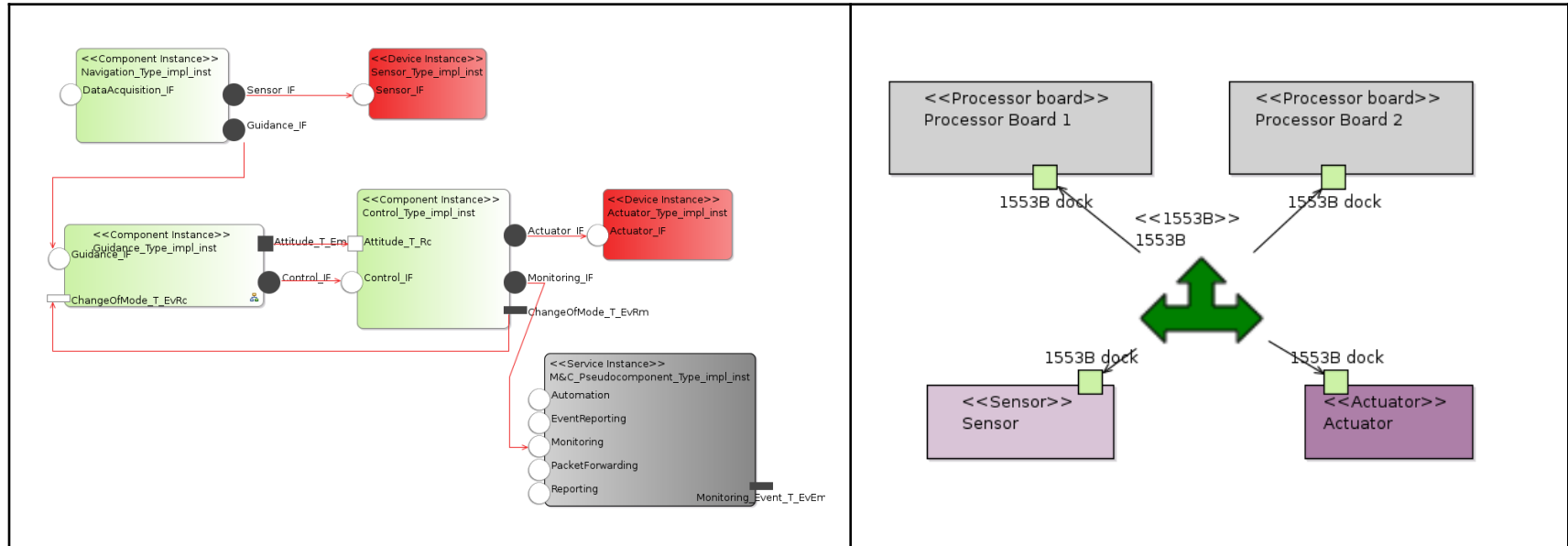
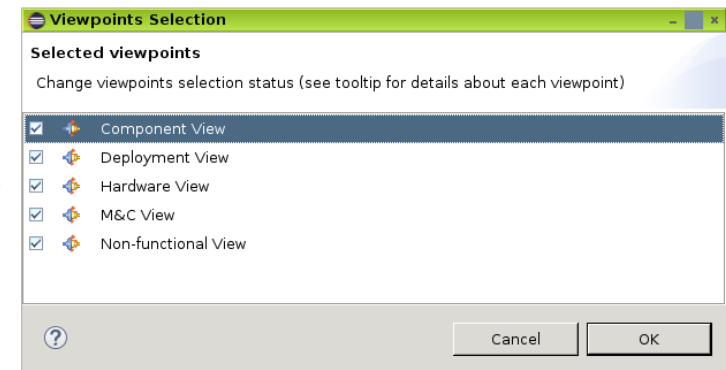
The COrDeT-3 toolset can be extended to integrate the COrDeT-2 capabilities.

- ❑ **Design Environments:**

	COrDeT-2	COrDeT-3
<b>Development Environment</b>	<ul style="list-style-type: none"> <li>○ Eclipse 3.</li> <li>○ Obeo Designer.</li> <li>○ Integrated with TASTE.</li> </ul>	<ul style="list-style-type: none"> <li>○ Eclipse 4.</li> <li>○ Sirius.</li> </ul>

# FUNDAMENTAL CONCEPTS

- ❑ Separation of concerns.
  - Design views. →
- ❑ Component-Based Development Engineering (CBDE).
- ❑ Model-Driven Engineering (MDE).

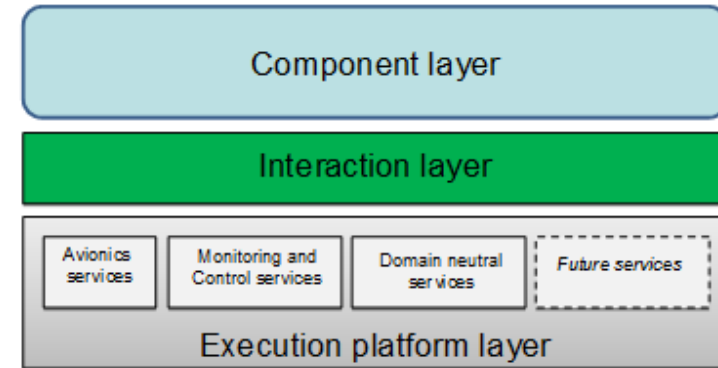


# CORDET-2 TOOL CHAIN

# OBJECTIVES

❑ COrDeT-2 produced the **COrDeT-2 Tooling Framework**.

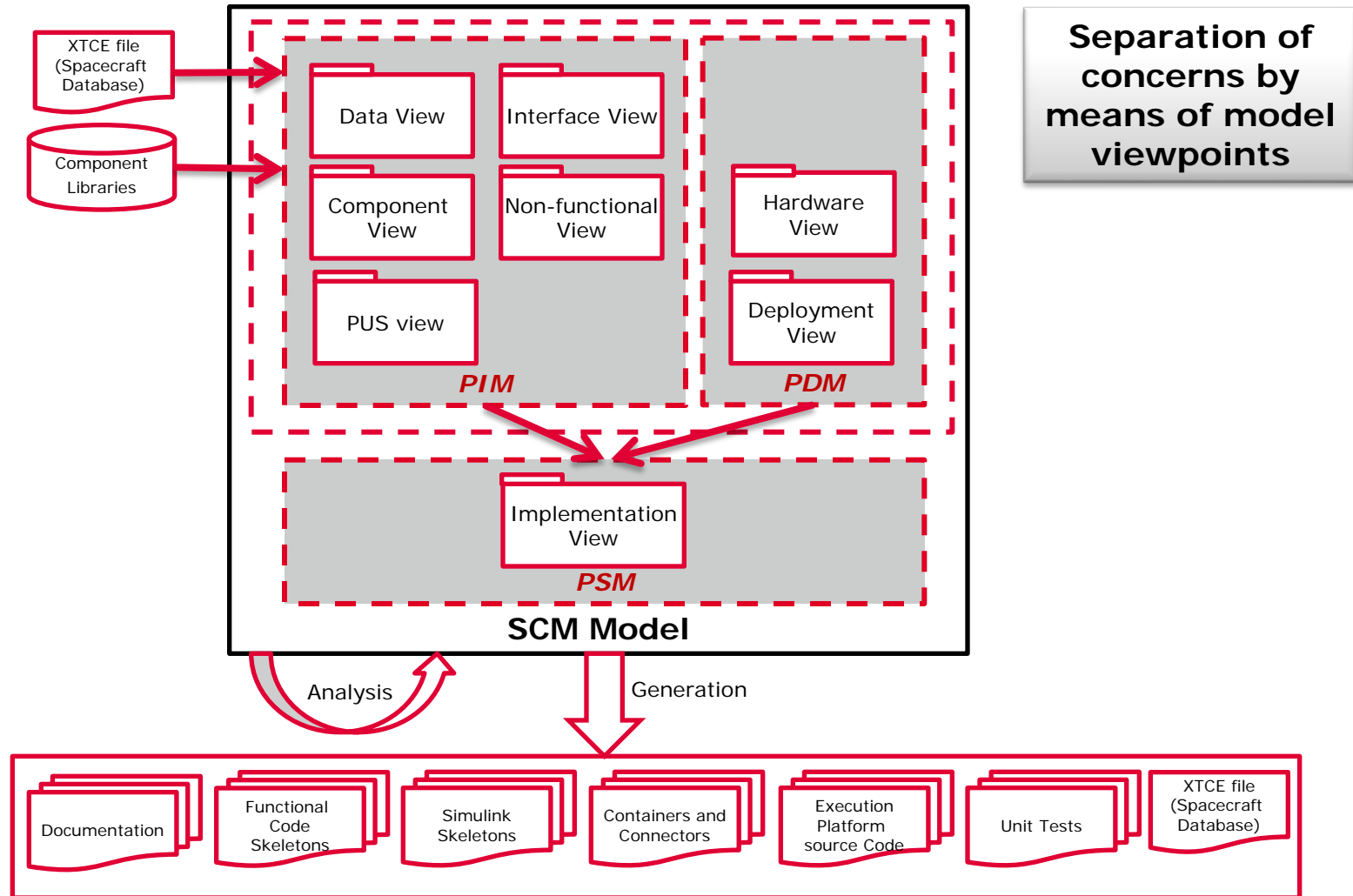
❑ Goal:



- Support the definition of the On-Board Software Reference Architecture definition.
  - Methodological approach for developing Space Applications.
  - The prototype toolset elaborates the Component Layer, the Interaction Layer and some representative Platform Services.
- Prototype the most important interfaces:
  - Monitoring and Control (M&C) pseudo-components (e.g., Monitoring Service, Housekeeping Service).
  - Avionics SOIS services (e.g., CDAS, MTS, TAS).
  - Automatic generation of the Execution Platform code.
  - Automatic generation of Ground.



# METHODOLOGICAL APPROACH



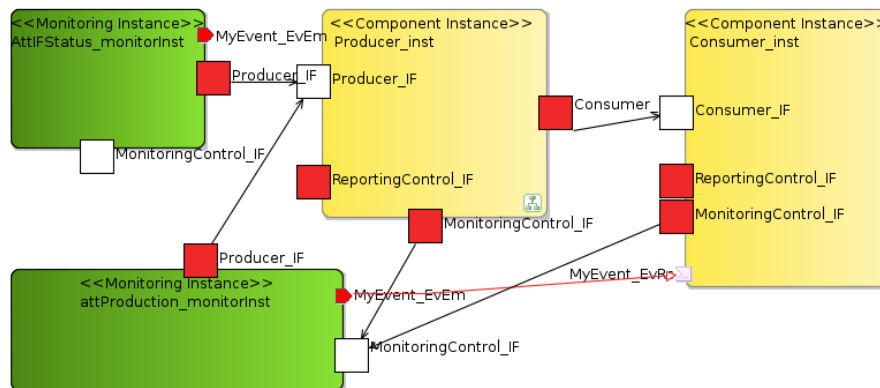
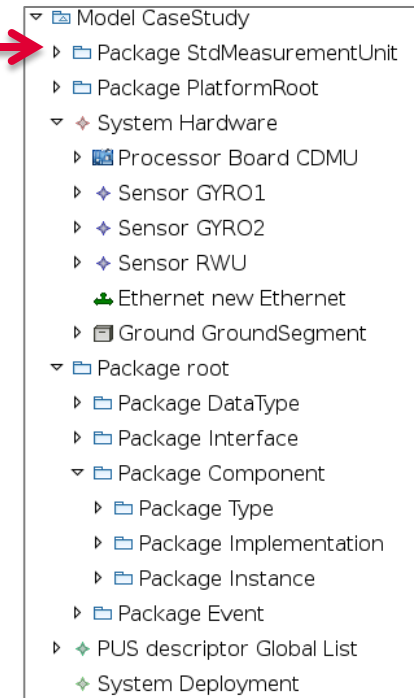
# FEATURES (1/2)

## ❑ SCM meta-model and SCM model Editor:

- Ecore meta-model and source code.
- EMF-based editor for manipulating SCM models.

## ❑ Graphical Model Editor:

- Library management.
- Design views.
- Representations: diagrams and tables.



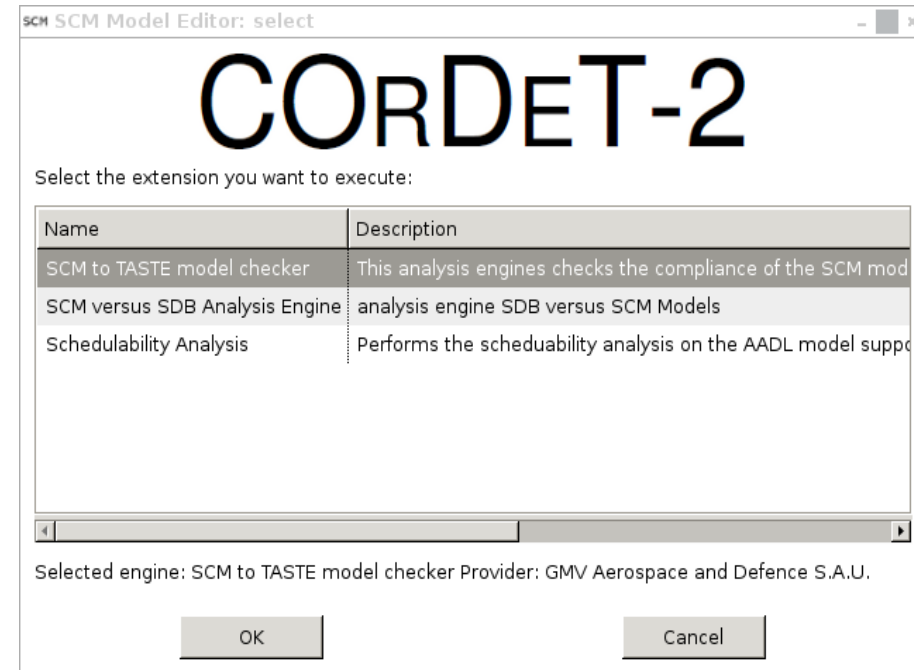
## FEATURES (2/2)

### ❑ Verification and Analysis Tools.

- Schedulability analysis.
- Model verification analyses.

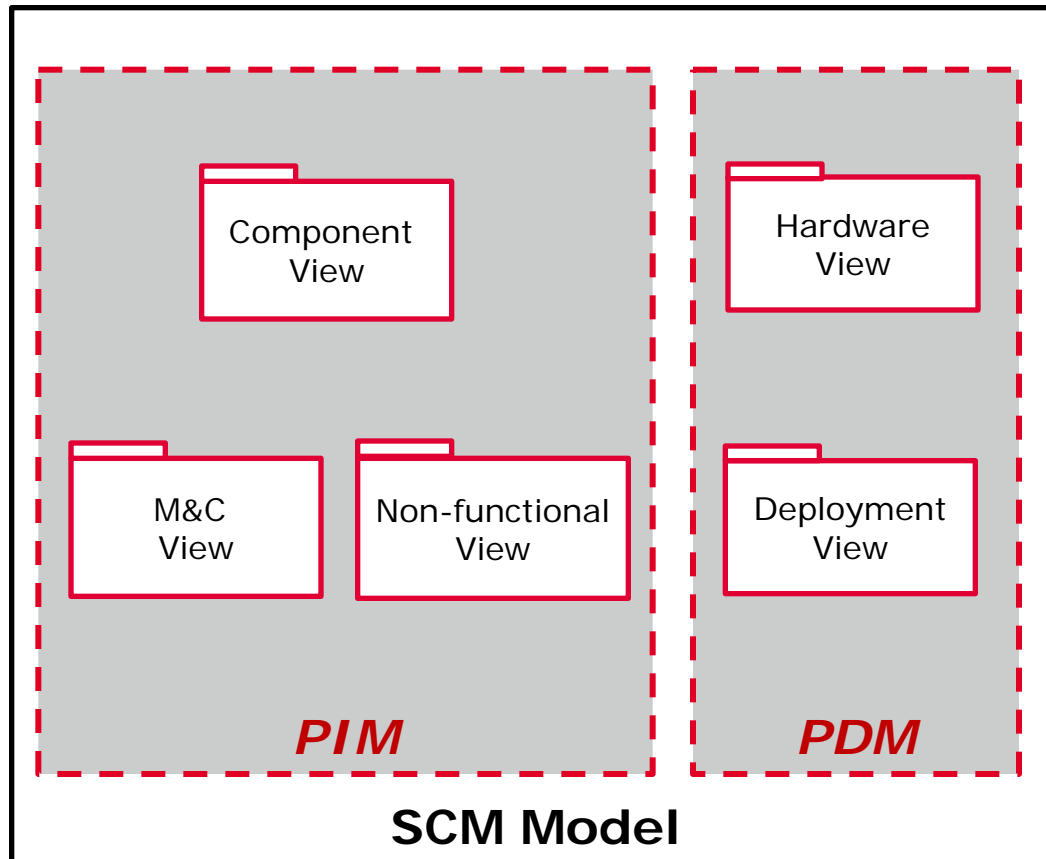
### ❑ Code Generators.

- Generation of source code skeletons.
- Generation of the execution platform services.
- Generation of the executable.



# CORDET-3 GRAPHICAL MODEL EDITOR

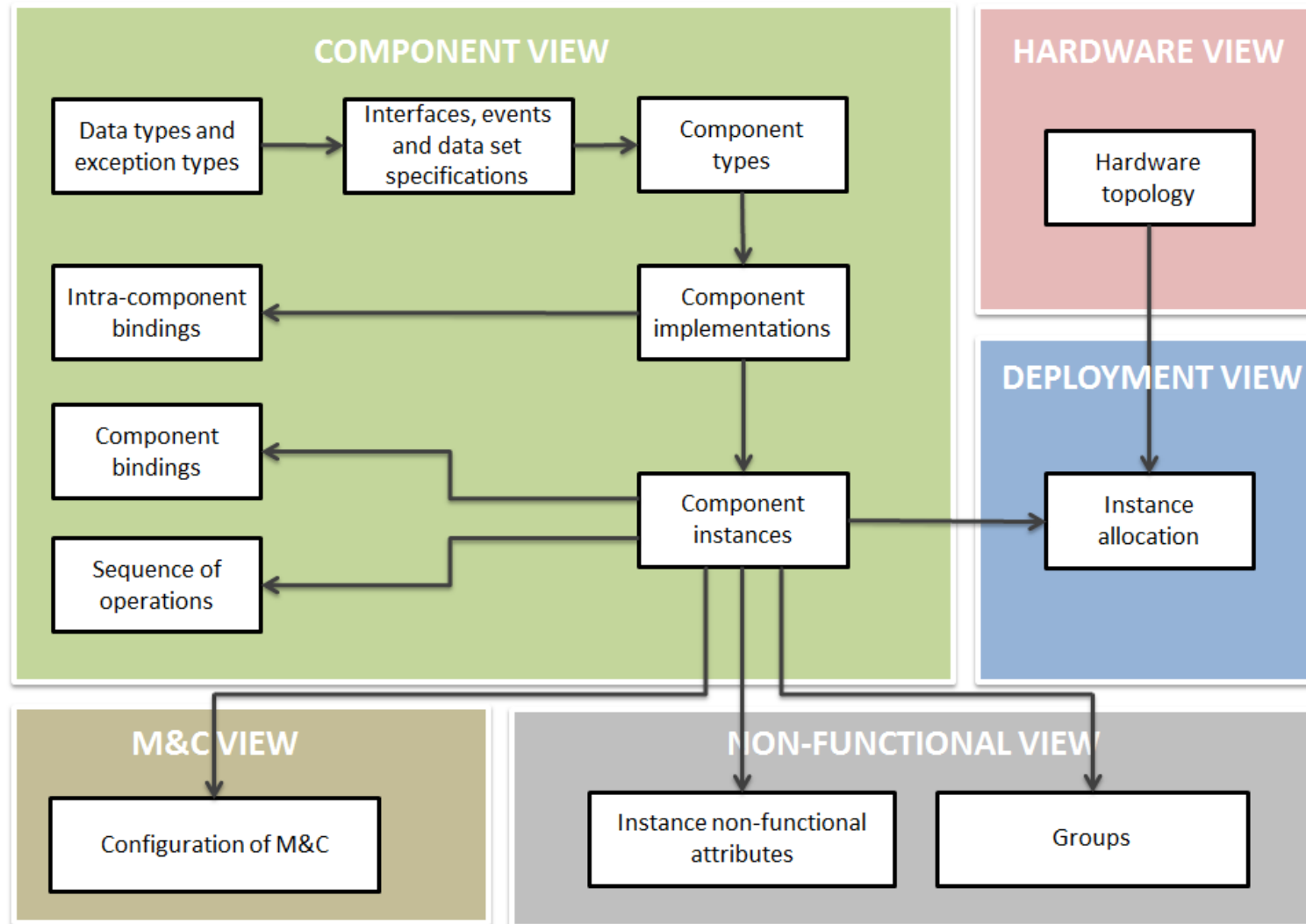
# METHODOLOGICAL APPROACH



Separation of concerns by means of model viewpoints

- The number of views has been reduced to 5:
  - **Component View:** Definition of the SW system.
  - **Non-functional View:** Specification of non-functional properties (e.g., concurrency kind, groups) of the components.
  - **M&C View:** Specification of M&C features (e.g., observability).
  - **Hardware View:** Definition of the HW topology.
  - **Deployment View:** Allocation of components onto nodes.

# DESIGN DEPENDENCIES

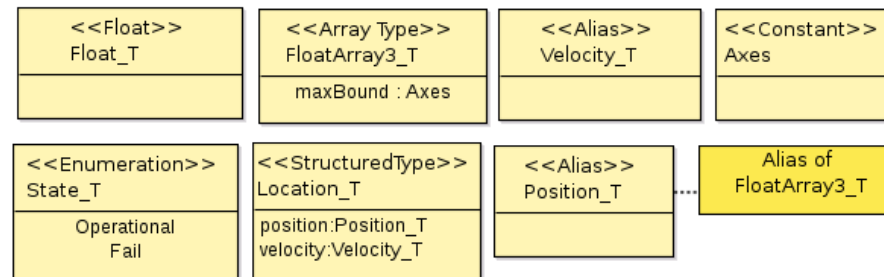


# COMPONENT VIEW (1/3)

## □ Definition of **Data** and **Exceptions** types.

### ■ Data Types:

- Elementary and structure data types.
- Alias, constants and constrained data types.



### ■ Exception Types:

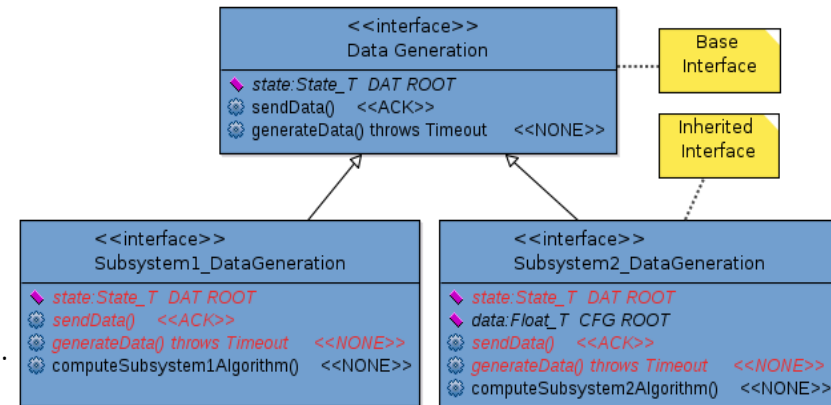


# COMPONENT VIEW (2/3)

## ❑ Definition of **Interfaces**, **Events** and **Data Set Specifications**.

### ■ Interfaces:

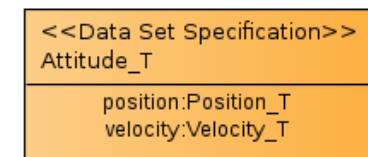
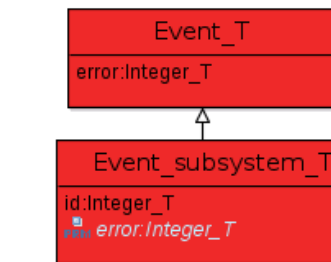
- Inheritance.
- Interface attributes:
  - Kind: DAT (read-only), CFG (read-write).
  - Accessibility : ROOT, LEAF, ROOT AND LEAF.
- Interface operations:
  - Interaction patterns: SEND, REQUEST, SUBMIT, INVOKE and PROGRESS.
  - Exceptions.



### ■ Events:

- Inheritance.
- Event parameters.

### ■ Data Set Specifications.



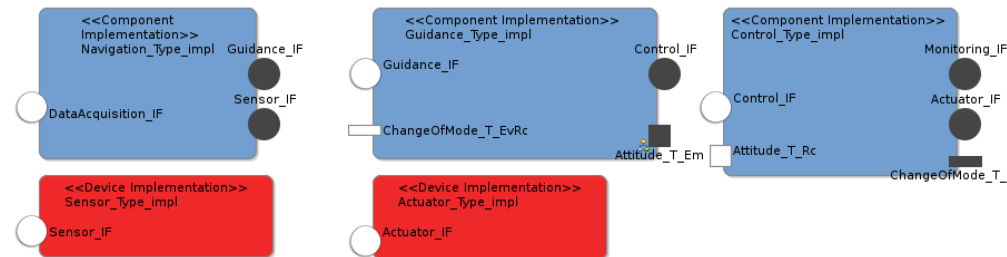


# COMPONENT VIEW (3/3)

## ❑ Definition of **Component Types** and **Implementations**.

### ■ Interaction:

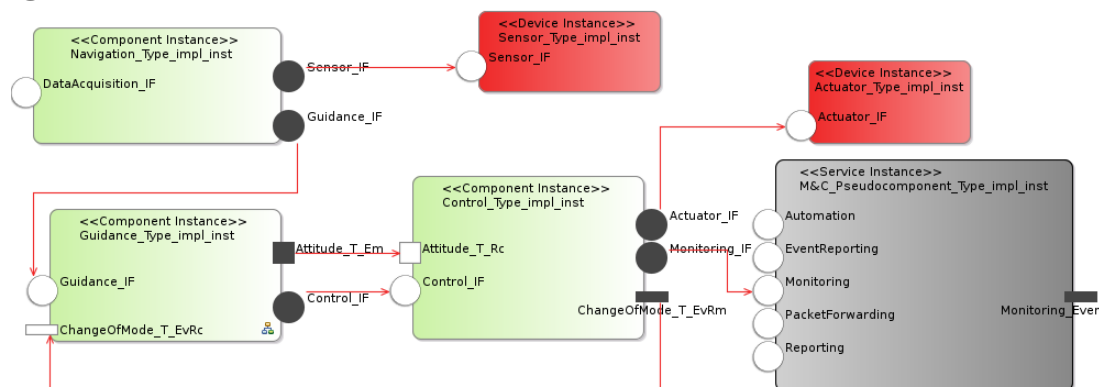
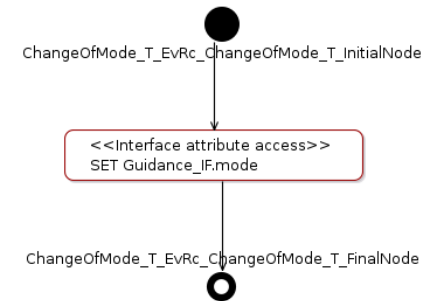
- Provided/Required Ports.
- Data Emitter/Receiver Ports.
- Event Emitter/Receiver Ports.



### ■ Definition of component type/implementation attributes.

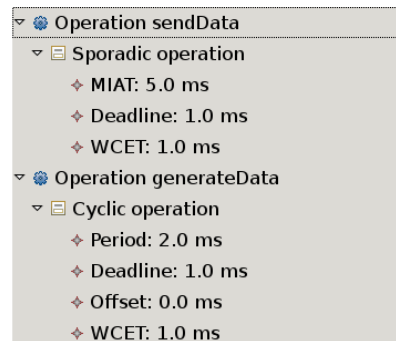
### ■ Definition of the intra-component bindings of the operations/events of the component implementations.

## ❑ Definition of **Component Instances** and the **sequence of operations**.

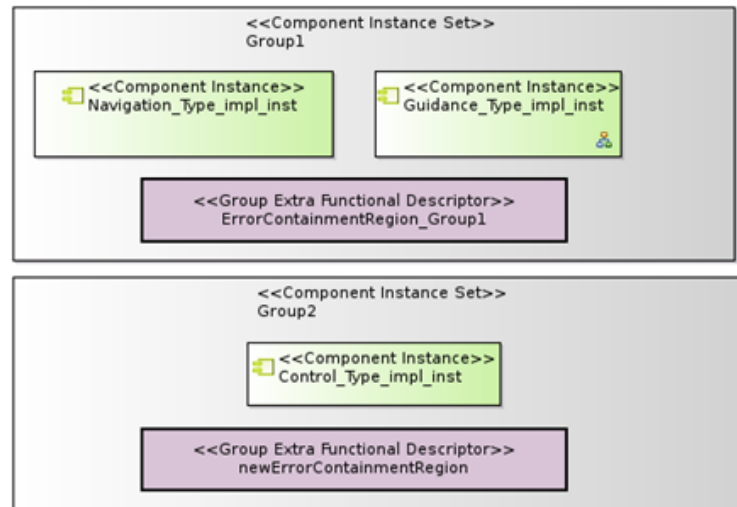


# NON-FUNCTIONAL VIEW

## ❑ Operations/Event/Data handlers **Non-Functional Descriptors:**



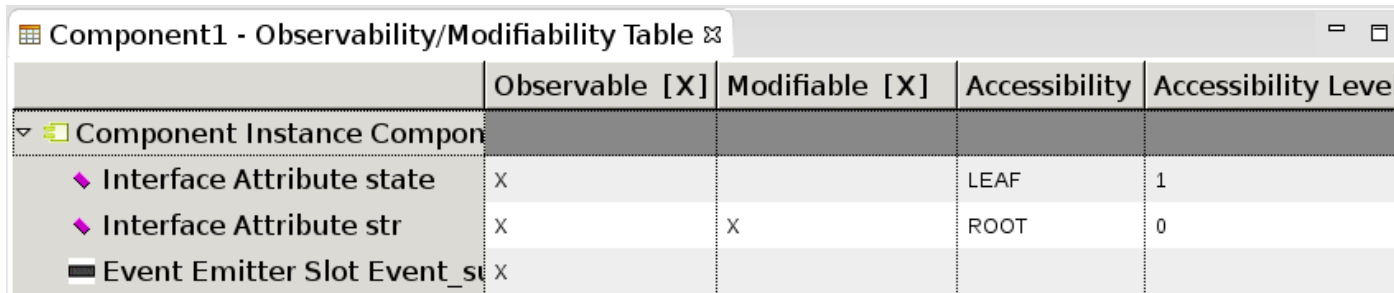
## ❑ Definition of **Groups**:



# MONITORING AND CONTROL VIEW

## □ Observability and Modifiability.

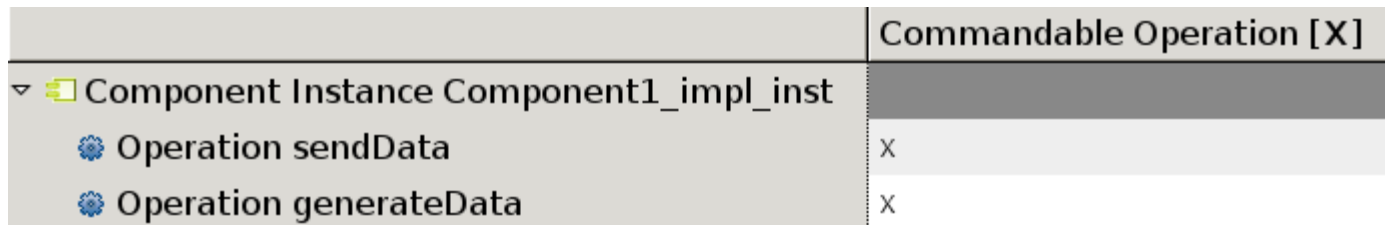
- The observability/modifiability descriptor specifies if the interface attributes or events can be observed/modified from ground.



	Observable [X]	Modifiable [X]	Accessibility	Accessibility Level
▼ Component Instance Component1				
◆ Interface Attribute state	X		LEAF	1
◆ Interface Attribute str	X	X	ROOT	0
■ Event Emitter Slot Event_slot	X			

## □ Commandability.

- A commandability descriptor is added to those interface operations that can be commanded from ground.

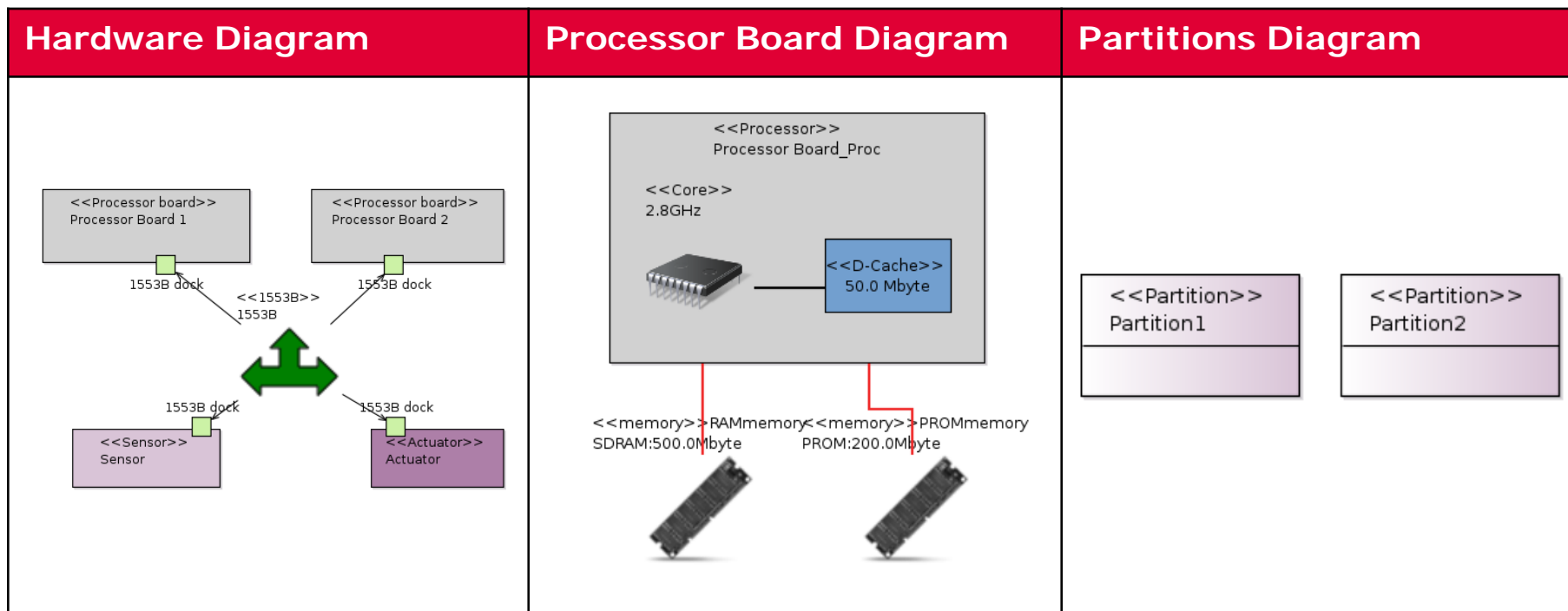


	Commandable Operation [X]
▼ Component Instance Component1_impl_inst	
⚙ Operation sendData	X
⚙ Operation generateData	X

# HARDWARE VIEW




❑ Definition of the Hardware topology. It includes:

- Mono-core and multi-core systems.
- Non-TSP and TSP platforms.



# DEPLOYMENT VIEW



## ❑ Software deployment: Non-TSP and TSP:

	Processor Core
▼  Component Instance:Control_Type_impl_inst	
◆ Deployment Descriptor	Processor Core PB1_core1
 Component Instance:Guidance_Type_impl_inst	
 Component Instance:Navigation_Type_impl_inst	

	Partition
▼ ◆ Component Instance Set Group1	
◆ Deployment Descriptor	Partition Partition1
▼ ◆ Component Instance Set Group2	
◆ Deployment Descriptor	Partition Partition2

	Processor Core
▼ ◆ Partition Partition1	
◆ Deployment Descriptor	Processor Core PB1_core1
▼ ◆ Partition Partition2	
◆ Deployment Descriptor	Processor Core PB2_core1

## ❑ Device deployment:

	HW Device
▼  Device Instance:Actuator_Type_impl_inst	
◆ Deployment Descriptor	Actuator Thruster
▼  Device Instance:Sensor_Type_impl_inst	
◆ Deployment Descriptor	Sensor Star Tracker

## ❑ Bindings deployment:

	Slot binding	Deployed on
▼ ◆ System Deployment System De		
◆ Slot Binding Deployment Des	Data Slot Binding Guidance_Type_in	MIL STD 1553B 1553B
◆ Slot Binding Deployment Des	Event Slot Binding Control_Type_imp	MIL STD 1553B 1553B
◆ Slot Binding Deployment Des	Interface Slot Binding Control_Type_	MIL STD 1553B 1553B
◆ Slot Binding Deployment Des	Interface Slot Binding Control_Type_	MIL STD 1553B 1553B

# CONCLUSIONS

# CONCLUSIONS

## ❑ COrDeT-2 toolset has demonstrated:

- Feasibility to develop the whole on-board software development: from the design of the user model to the generation of the executable.

## ❑ COrDeT-3 toolset:

- It is in line with the OSRA specification.
- It can be extended to include the functionalities of COrDeT-2.
- It is supported by open-source tools.



# Thank you!

Elena Alaña: [ealana@gmv.com](mailto:ealana@gmv.com)  
Santiago Urueña: [suruena@gmv.com](mailto:suruena@gmv.com)

Space Systems Business Unit  
Avionics & On-Board SW Division

