# Scheduling Configuration of Real-Time Component-Based Applications

Patricia López Martínez, Laura Barros and José M. Drake

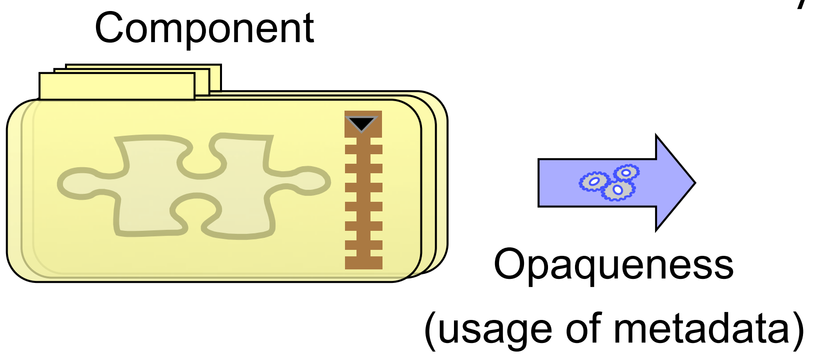
Grupo de Computadores y Tiempo Real
Universidad de Cantabria, Spain

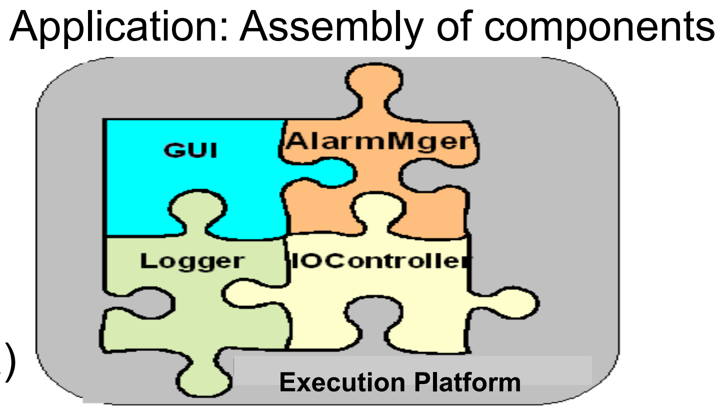
15th International Conference on Reliable Software
AdaEurope 2010
Valencia, June 2010

Funded by the European Union's FP7 under contracts FP7/NoE/214373 and IST-004527, and by the Spanish Government under grant TIN2008-06766-C03-03

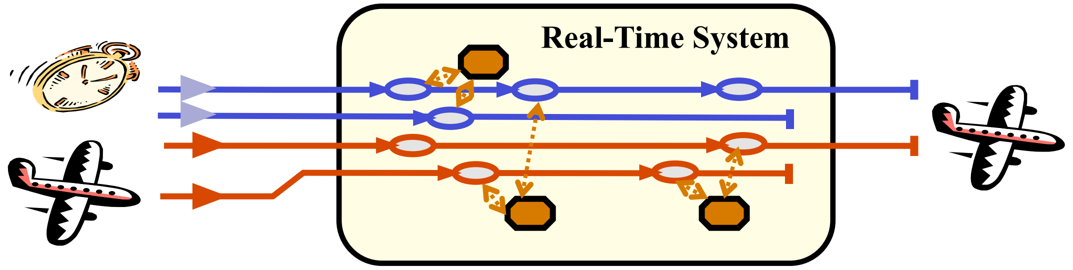
## Objective: Component-based real-time applications

- **Component-based approaches**





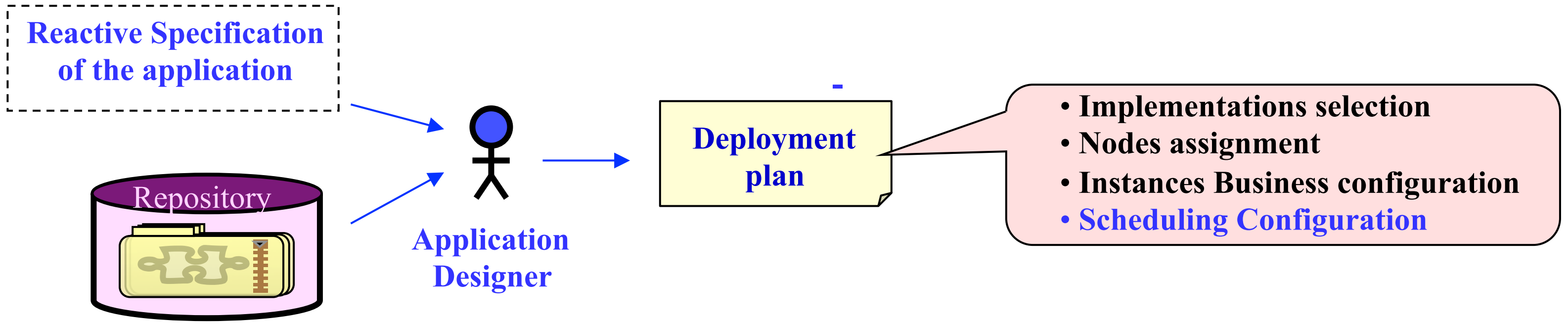
- **Real-time systems**
  - Reactive model of real-time systems:
    - Applications conceived as a set of concurrent end-to-end flow transactions
    - Timing requirements defined as temporal constraints in the transaction



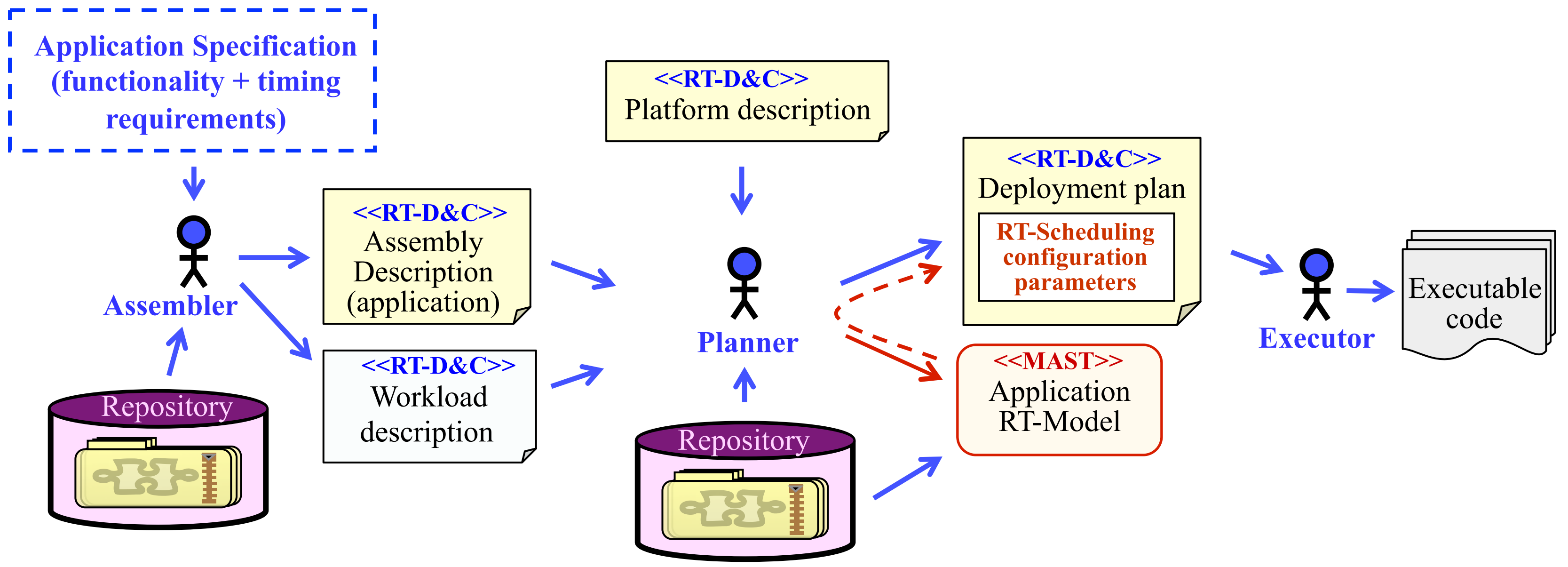
## Real-time design

- In traditional systems:
  - The designer can define and control:
    - The number of threads
    - The assignment of activities to the threads
    - The synchronization mechanisms
    - The scheduling parameters and policies
  - A real-time model is usually used:
    - To obtain the correct scheduling parameters assignment or to certify the fulfilment of the timing requirements
    - It is formulated at the same time as the code is elaborated

- In a component-based system:
  - The code of the components is opaque
  - The deployment plan is the only way to configure the application
  - The real-time model must be obtained from metadata provided by the components

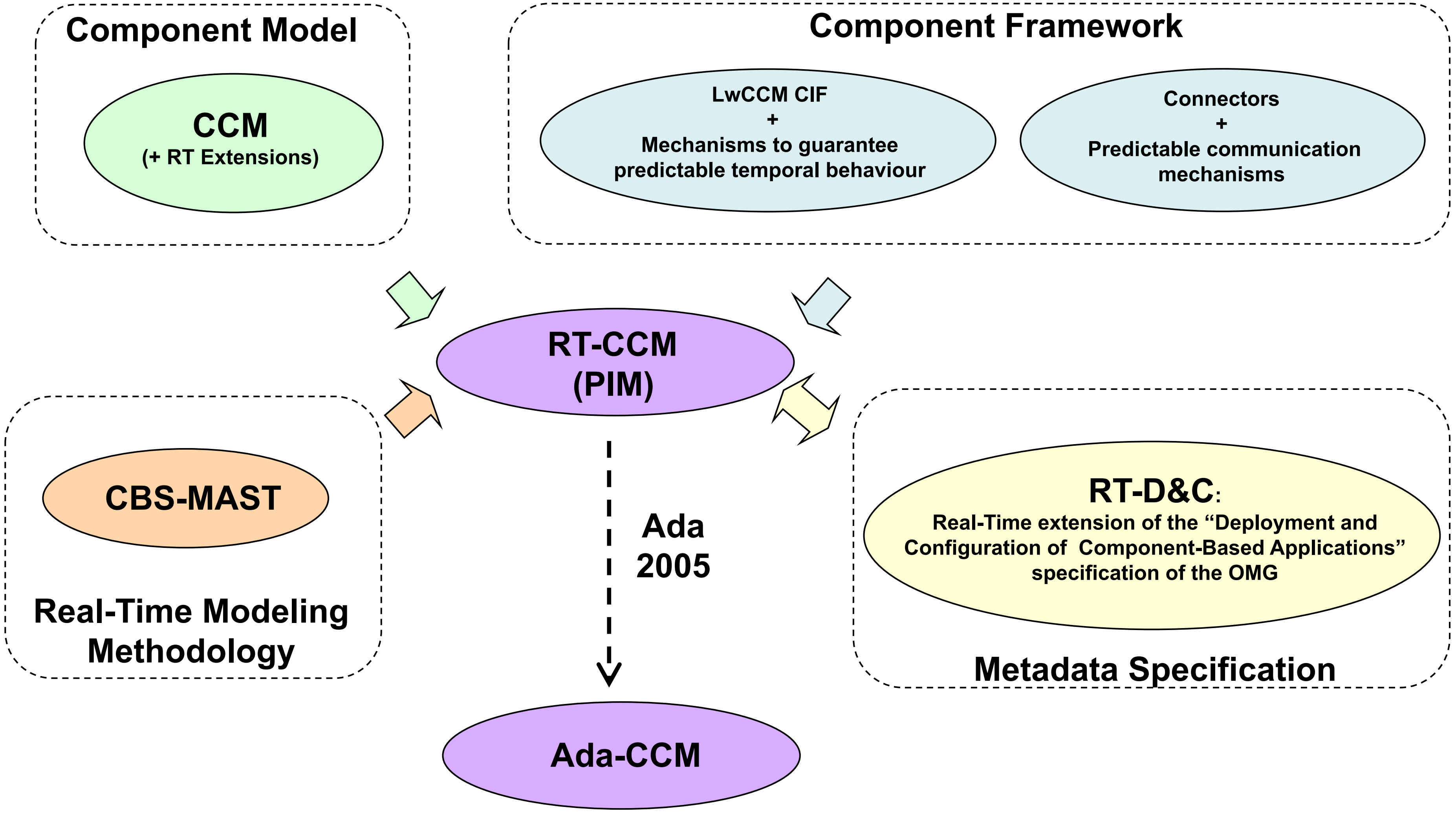

Reactive Specification of the application
Repository
Application Designer
Deployment plan
• Implementations selection
• Nodes assignment
• Instances Business configuration
• Scheduling Configuration

# Real-time Component-Based Design Process

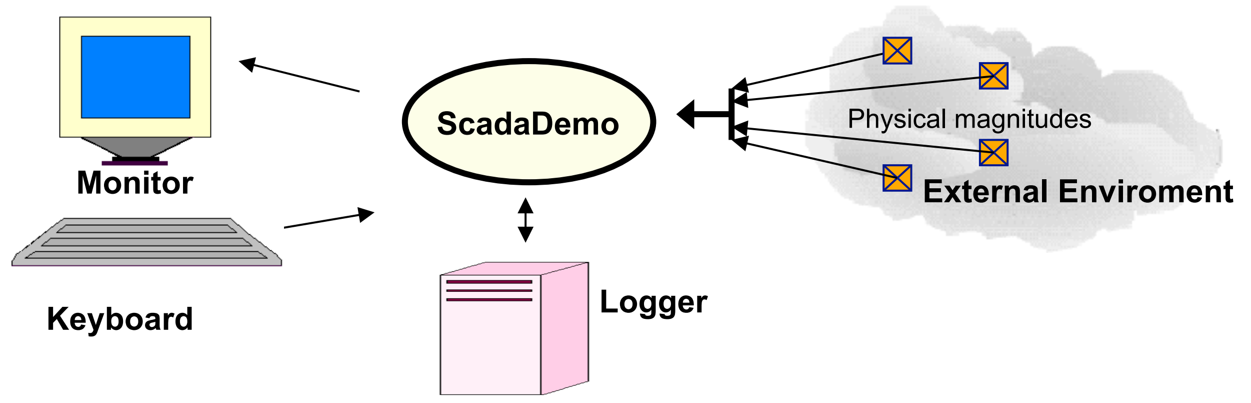

Application Specification (functionality + timing requirements)
Assembler
Repository
<<RT-D&C>> Assembly Description (application)
<<RT-D&C>> Workload description
<<RT-D&C>> Platform description
Planner
Repository
<<RT-D&C>> Deployment plan
RT-Scheduling configuration parameters
<<MAST>> Application RT-Model
Executor
Executable code

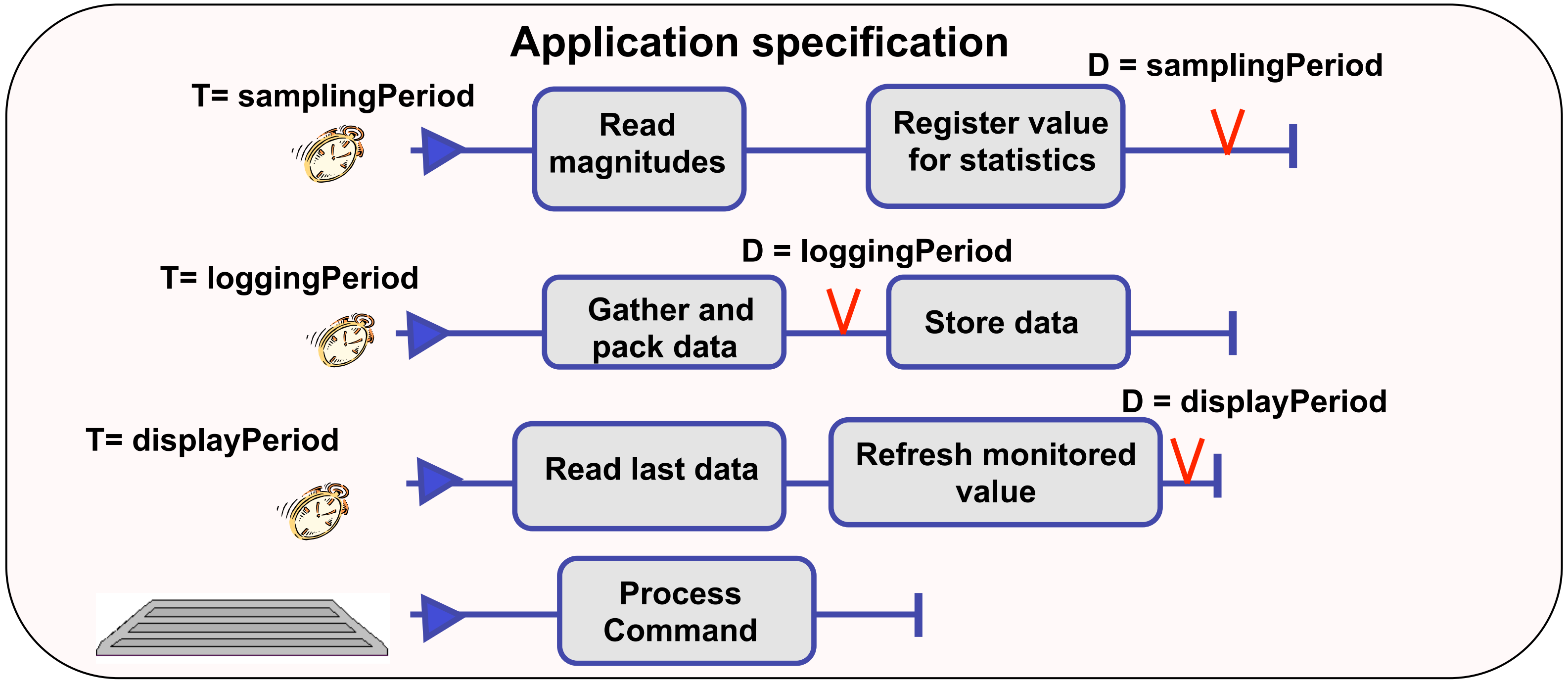
- Timing metadata associated to the components and the deployment plan to configure the application schedulability in an opaque way => **RT-D&C**
- The components must provide temporal behaviour models => **CBS-MAST**
- The component technology must provide mechanisms to control the application scheduling (also in an opaque way) => **RT-CCM**

## RT-CCM: Real-Time Container Component Model

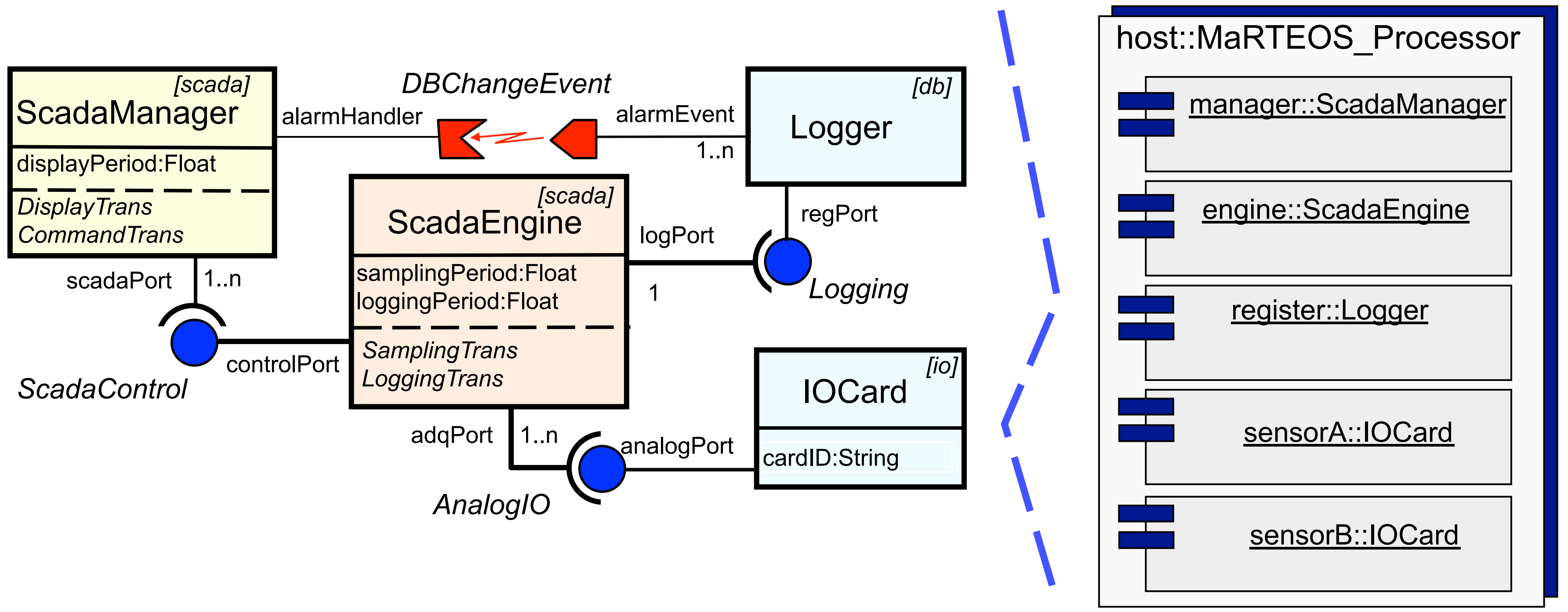

Component Model
CCM
(+ RT Extensions)
Component Framework
LwCCM CIF
+
Mechanisms to guarantee predictable temporal behaviour
Connectors
+
Predictable communication mechanisms
RT-CCM
(PIM)
CBS-MAST
Real-Time Modeling Methodology
Ada 2005
RT-D&C:
Real-Time extension of the "Deployment and Configuration of Component-Based Applications" specification of the OMG
Metadata Specification
Ada-CCM

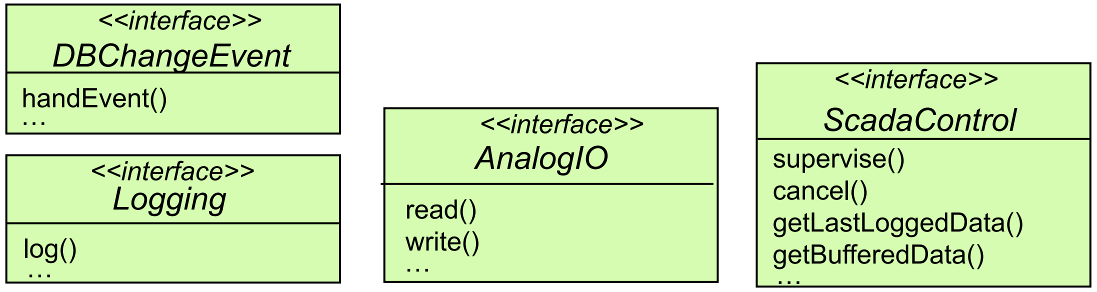
# Application example: ScadaDemo





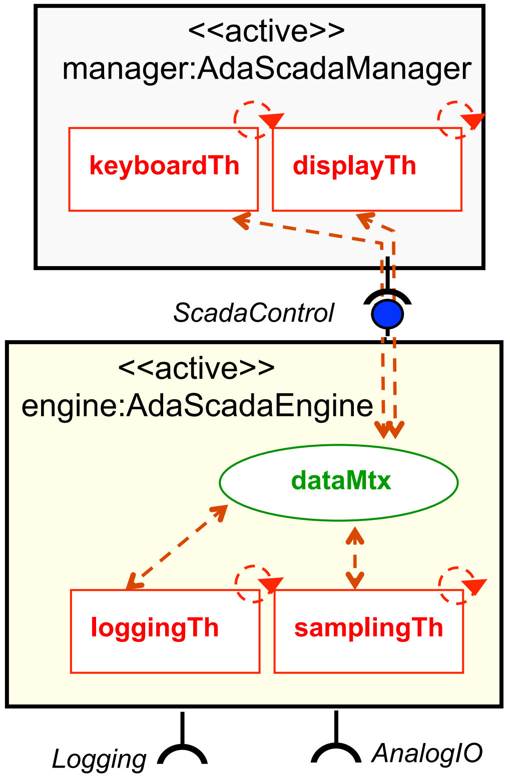
## ScadaDemo architecture and deployment





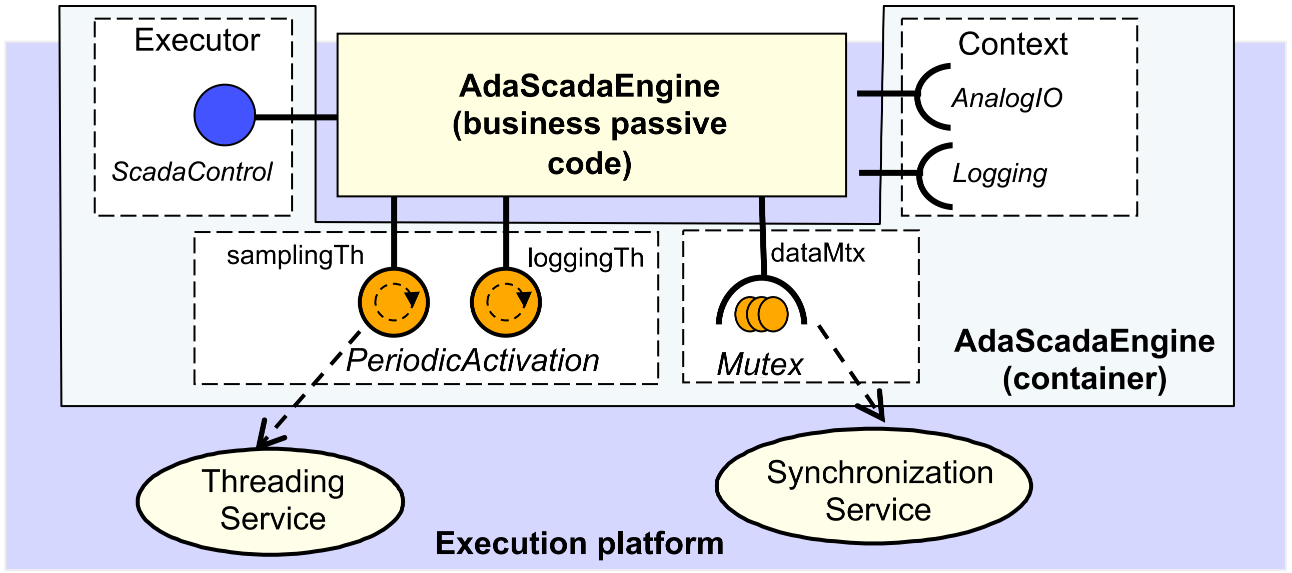
## Concurrency in an RT-CCM component

- The business code of an RT-CCM component may be concurrently executed by multiple threads:
  - Created by the component itself
    - To attend external events or execute internal activities
    - Ex: loggingTh and samplingTh
  - Coming from external components that invoke its services:
    - Ex: keyboardTh and displayTh
- Synchronization mechanisms are required to guarantee mutual exclusive access to shared resources:
  - Ex: dataMtx
- Suitable values of the scheduling parameters of threads and synchronization mechanisms obtained from the real-time design

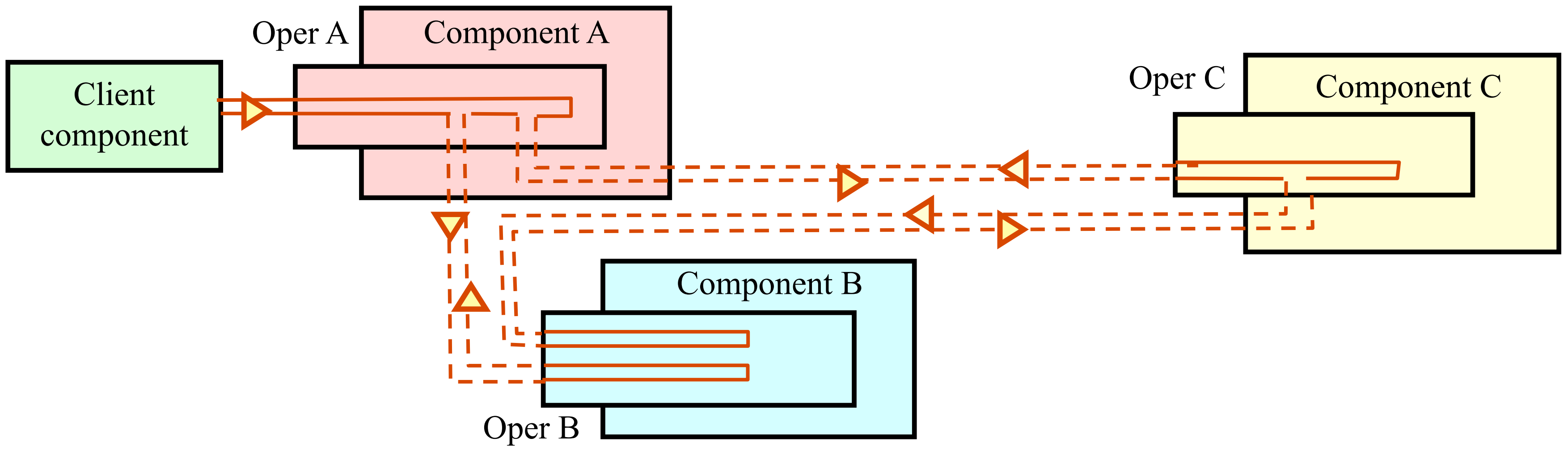


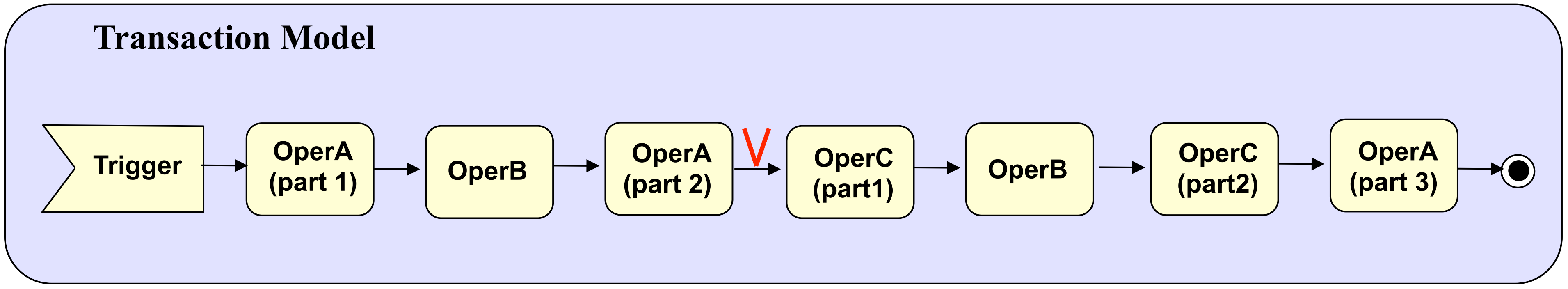
## Concurrency support in RT-CCM

- In RT-CCM, the container is responsible of creating and managing:
  - Threads => Activation Ports and ThreadingService
    - Two types of activation ports:
      - PeriodicActivation port
      - OneShotActivation port
  - Synchronization mechanisms > Synchronization ports + SynchronizationService
    - Two types of mechanisms:
      - Mutex
      - ConditionVariable



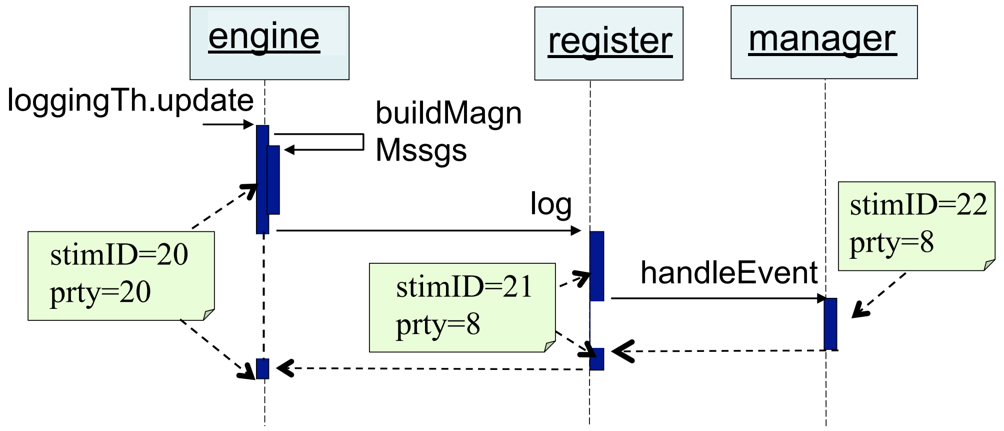
# Scheduling parameters assignment in RT-CCM

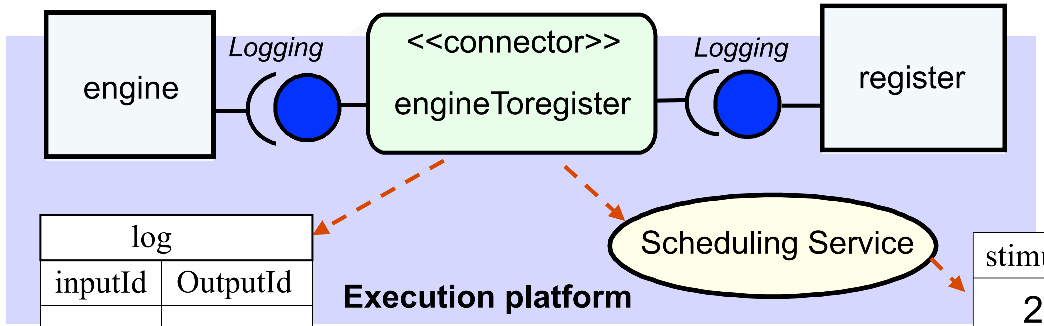




## Scheduling Support in RT-CCM

- A *Transaction_Defined* policy is managed in RT-CCM by connectors through
  - StimulusId
  - SchedulingService

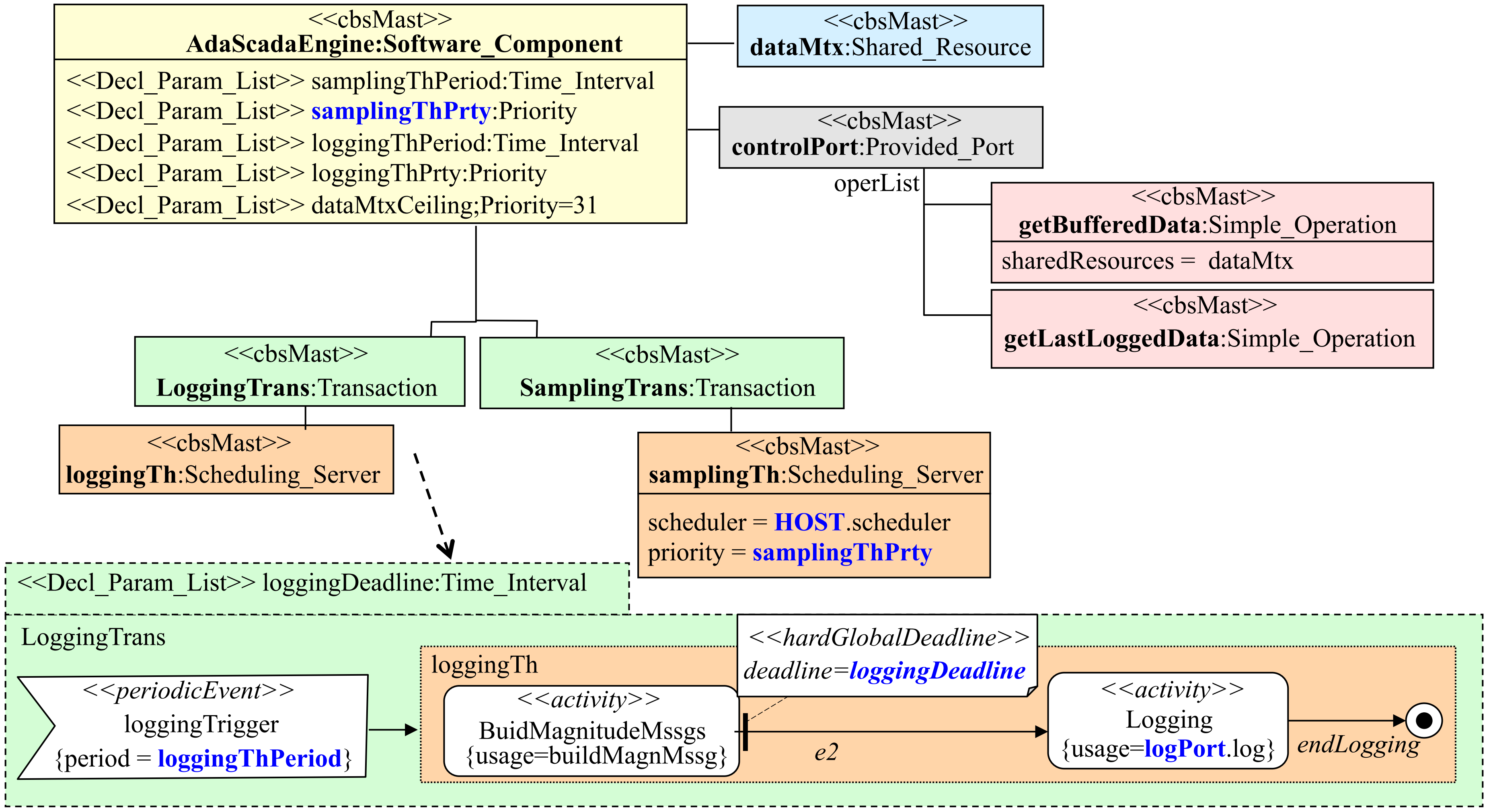




| log | |
|---|---|
| inputId | OutputId |
| 20 | 21 |
| 80 | 90 |

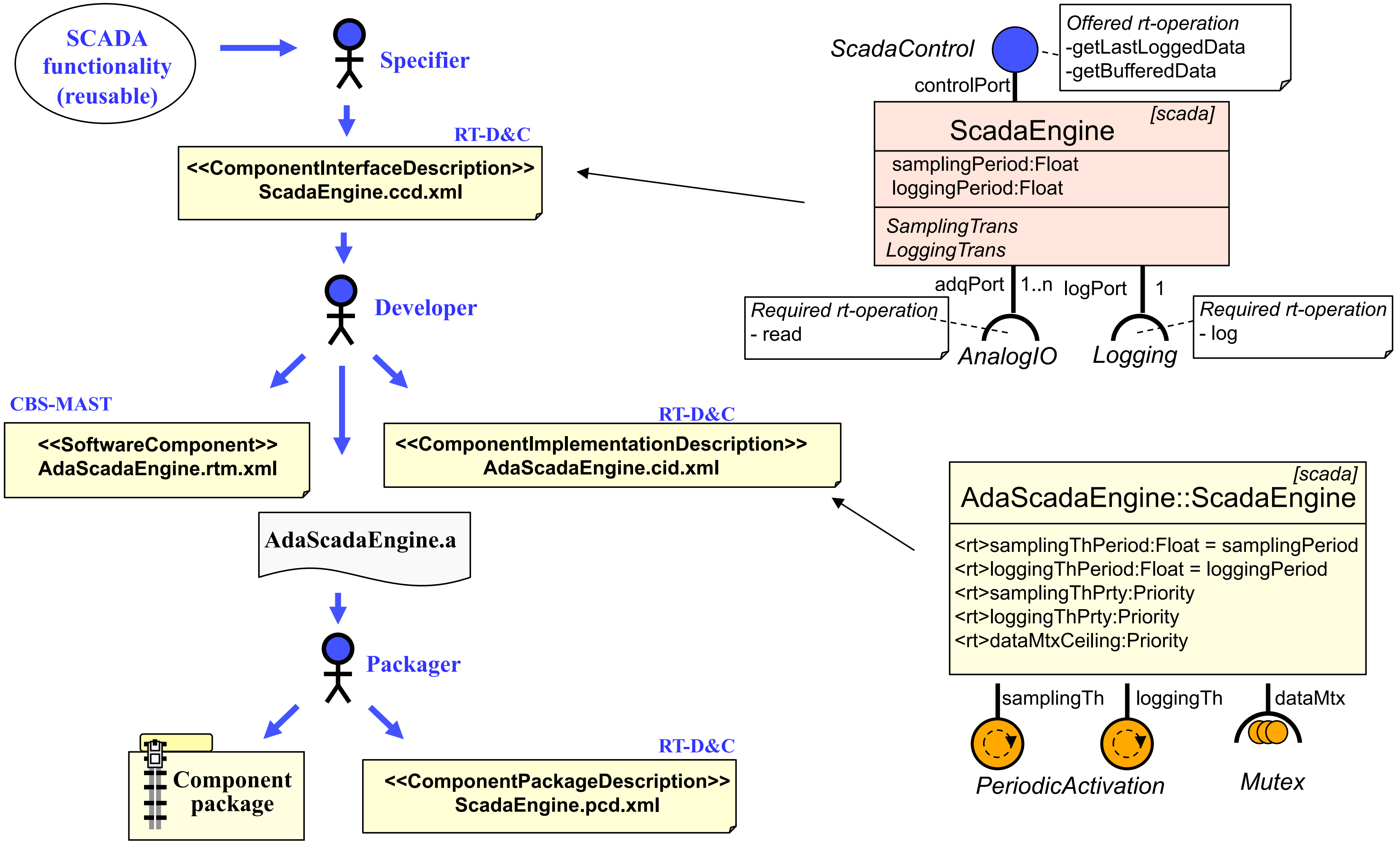
| stimulusId | Prioridad |
|---|---|
| 20 | 20 |
| 21 | 8 |
| 22 | 8 |

# Real-Time model of a RT-CCM Component



<<cbsMast>>
AdaScadaEngine:Software_Component
<<Decl_Param_List>> samplingThPeriod:Time_Interval
<<Decl_Param_List>> samplingThPrty:Priority
<<Decl_Param_List>> loggingThPeriod:Time_Interval
<<Decl_Param_List>> loggingThPrty:Priority
<<Decl_Param_List>> dataMtxCeiling;Priority=31
<<cbsMast>>
dataMtx:Shared_Resource
<<cbsMast>>
controlPort:Provided_Port
operList
<<cbsMast>>
getBufferedData:Simple_Operation
sharedResources = dataMtx
<<cbsMast>>
getLastLoggedData:Simple_Operation
<<cbsMast>>
LoggingTrans:Transaction
<<cbsMast>>
SamplingTrans:Transaction
<<cbsMast>>
loggingTh:Scheduling_Server
<<cbsMast>>
samplingTh:Scheduling_Server
scheduler = HOST.scheduler
priority = samplingThPrty
<<Decl_Param_List>> loggingDeadline:Time_Interval
LoggingTrans
<<periodicEvent>>
loggingTrigger
{period = loggingThPeriod}
loggingTh
<<activity>>
BuidMagnitudeMssgs
{usage=buildMagnMssg}
<<hardGlobalDeadline>>
deadline=loggingDeadline
e2
<<activity>>
Logging
{usage=logPort.log}
endLogging

# RT-CCM Component Development Process


SCADA functionality (reusable)
Specifier
RT-D&C
<<ComponentInterfaceDescription>> ScadaEngine.ccd.xml
Developer
CBS-MAST
<<SoftwareComponent>> AdaScadaEngine.rtm.xml
RT-D&C
<<ComponentImplementationDescription>> AdaScadaEngine.cid.xml
AdaScadaEngine.a
Packager
Component package
RT-D&C
<<ComponentPackageDescription>> ScadaEngine.pcd.xml
ScadaControl
controlPort
Offered rt-operation
-getLastLoggedData
-getBufferedData
ScadaEngine
[scada]
samplingPeriod:Float
loggingPeriod:Float
SamplingTrans
LoggingTrans
adqPort
1..n
logPort
1
Required rt-operation
- read
AnalogIO
Logging
Required rt-operation
- log
[scada]
AdaScadaEngine::ScadaEngine
<rt>samplingThPeriod:Float = samplingPeriod
<rt>loggingThPeriod:Float = loggingPeriod
<rt>samplingThPrty:Priority
<rt>loggingThPrty:Priority
<rt>dataMtxCeiling:Priority
samplingTh
loggingTh
dataMtx
PeriodicActivation
Mutex

# RT-CCM Application Development Process

SCADADemo Specification
(reactive description)

*Workload: Context Analysis*

*Business transactions declaration*

*f external events*

*nts*

*Applicatio*

*- Compone*

*- Compone*

*- Business*

Domain
ministrator

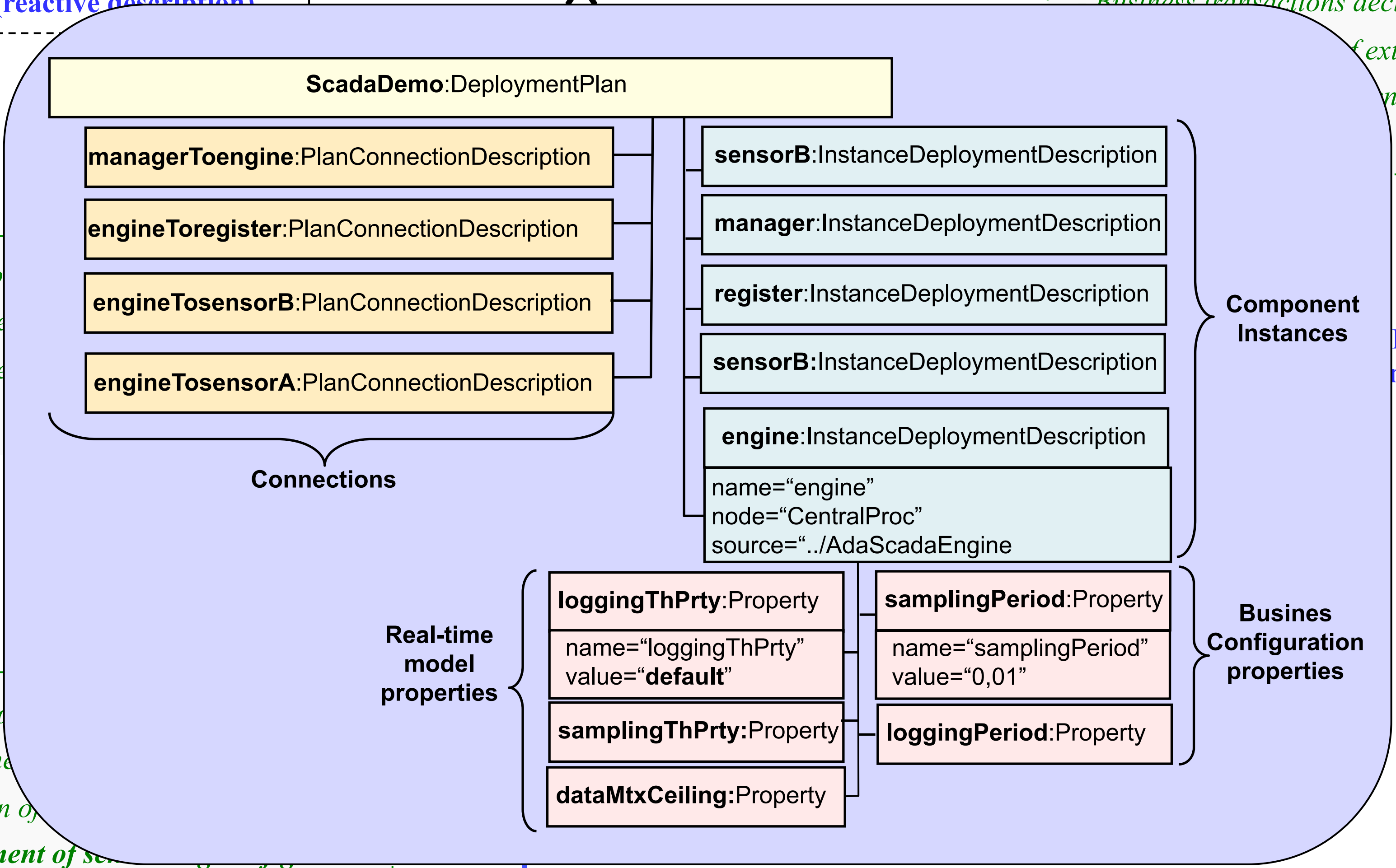
*Applicati*

*- Compone*

*- Selection o*

***- Assignment of s***

***properties***

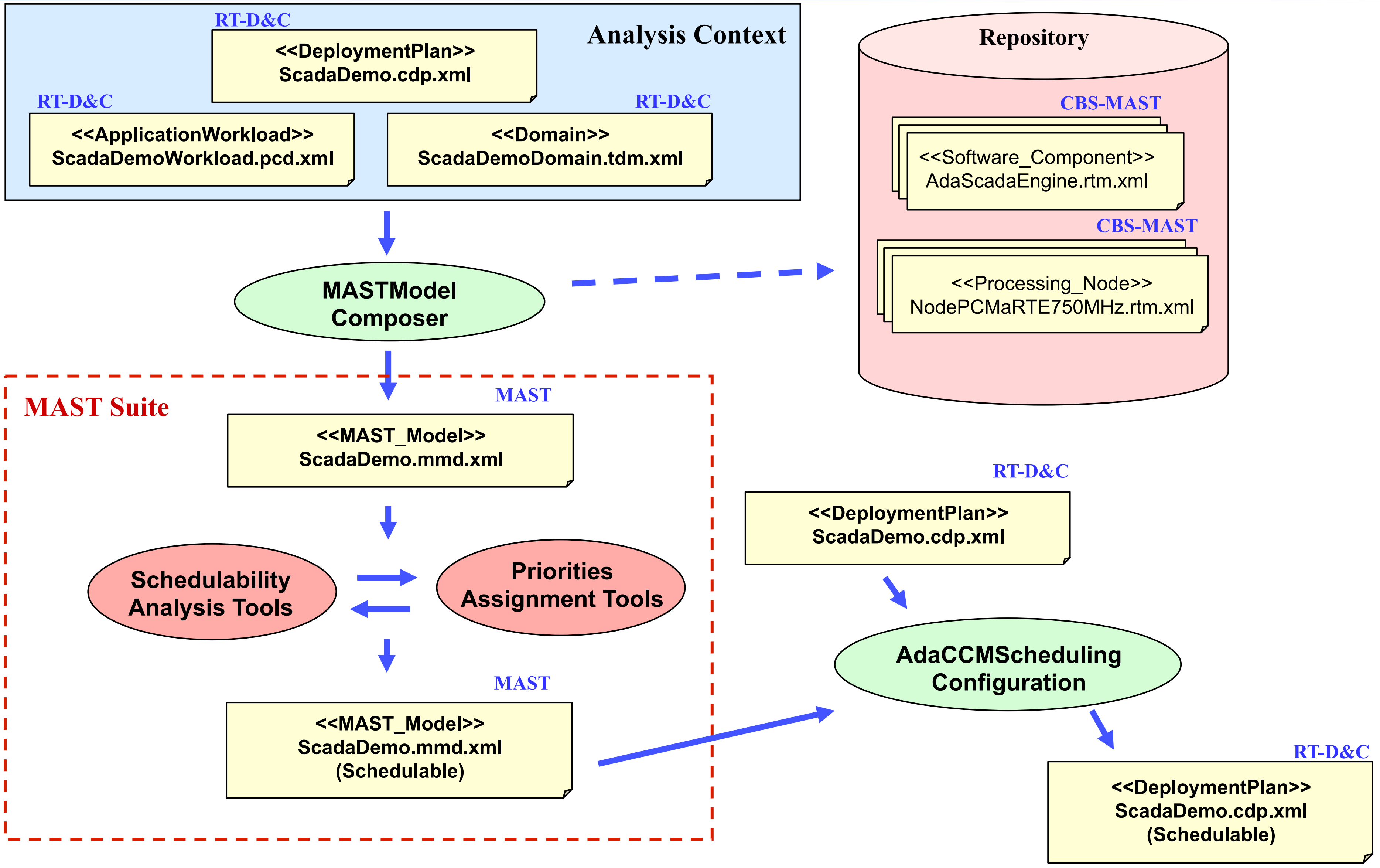


**ScadaDemo**:DeploymentPlan
**managerToengine**:PlanConnectionDescription
**engineToregister**:PlanConnectionDescription
**engineTosensorB**:PlanConnectionDescription
**engineTosensorA**:PlanConnectionDescription
Connections
**sensorB**:InstanceDeploymentDescription
**manager**:InstanceDeploymentDescription
**register**:InstanceDeploymentDescription
**sensorB:**InstanceDeploymentDescription
**engine**:InstanceDeploymentDescription
name="engine"
node="CentralProc"
source="../AdaScadaEngine
Component Instances
Real-time model properties
**loggingThPrty**:Property
name="loggingThPrty"
value="**default**"
**samplingThPrty:**Property
**dataMtxCeiling**:Property
**samplingPeriod**:Property
name="samplingPeriod"
value="0,01"
**loggingPeriod**:Property
Busines Configuration properties

ScadaDemo.exe

Application Execution

# Scheduling Configuration phase



## Results from the RTCCM Scheduling Configuration

- Assignment of stimulusId and priorities

| Transaction | Invocation (Instance.Operation) | Input StimulusId | Output StimulusId | Priority |
|---|---|---|---|---|
| **samplingTransInst** | | | 1 | 30 |
| | sensorA.read | 1 | 11 | 30 |
| | sensorB.read | 1 | 12 | 30 |
| **loggingTransInst** | | | 2 | 20 |
| | register.log | 2 | 21 | 8 |
| | manager.handEvent | 21 | 22 | 8 |
| **displayTransInst** | | | 3 | 10 |
| | engine.getLasLoggedMssg | 3 | 31 | 10 |
| | engine.getBufferedData | 3 | 32 | 10 |
| **commandTransInst** | | | 4 | 5 |

- Priority Ceilings of the synchronization ports

| Instance | Port | Ceiling |
|---|---|---|
| engine | dataMtx | 30 |
| sensorA | aiMtx | 30 |
| sensorB | aiMtx | 30 |
| manager | displayMtx | 15 |

- Initial StimulusId for the activation ports

| Instance | Port | StimulusId |
|---|---|---|
| engine | samplingTh | 1 |
| | loggingTh | 2 |
| manager | displayTh | 3 |
| | keyboardTh | 4 |

# Services and Connectors Configuration

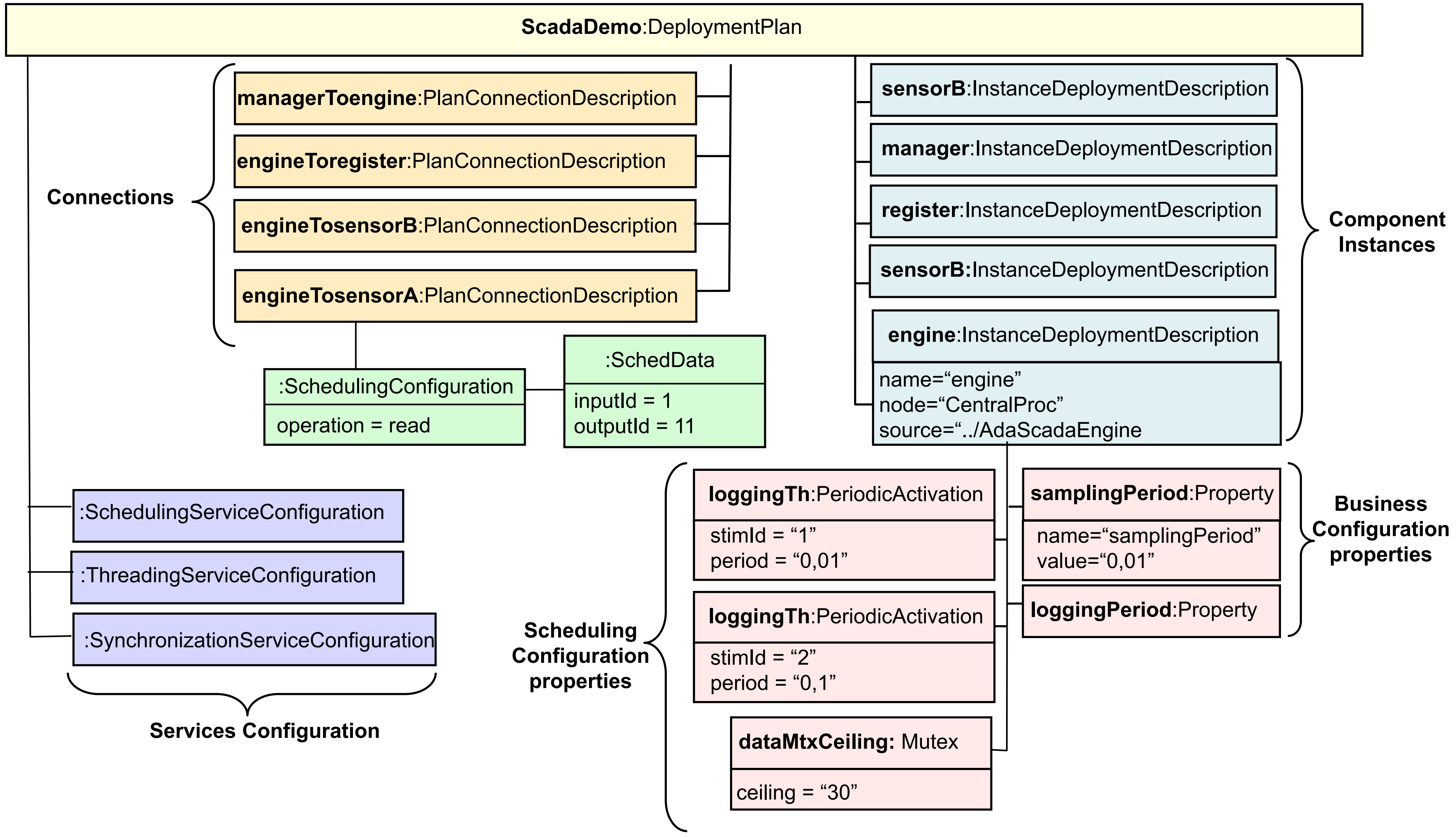
engineToSensorA Connector Configuration

| Transaction | Invocation (Instance.Operation) | Input StimulusId | Output StimulusId | Priority |
|---|---|---|---|---|
| samplingTransInst | | | 1 | 30 |
| | sensorA.read | 1 | 11 | 30 |
| | sensorB.read | 1 | 12 | 30 |
| loggingTransInst | | | 2 | 20 |
| | register.log | 2 | 21 | 8 |
| | manager.handEvent | 21 | 22 | 8 |
| displayTransInst | | | 3 | 10 |
| | engine.getLasLoggedMssg | 3 | 31 | 10 |
| | engine.getBufferedData | 3 | 32 | 10 |
| commandTransInst | | | 4 | 5 |

managerToengine Connector Configuration

SchedulingService Configuration

# ScadaDemo final deployment plan



**ScadaDemo**:DeploymentPlan
Connections
**managerToengine**:PlanConnectionDescription
**engineToregister**:PlanConnectionDescription
**engineTosensorB**:PlanConnectionDescription
**engineTosensorA**:PlanConnectionDescription
:SchedulingConfiguration
operation = read
:SchedData
inputId = 1
outputId = 11
**sensorB**:InstanceDeploymentDescription
**manager**:InstanceDeploymentDescription
**register**:InstanceDeploymentDescription
**sensorB:**InstanceDeploymentDescription
**engine**:InstanceDeploymentDescription
name="engine"
node="CentralProc"
source="../AdaScadaEngine
Component Instances
:SchedulingServiceConfiguration
:ThreadingServiceConfiguration
:SynchronizationServiceConfiguration
Services Configuration
Scheduling Configuration properties
**loggingTh**:PeriodicActivation
stimId = "1"
period = "0,01"
**loggingTh**:PeriodicActivation
stimId = "2"
period = "0,1"
**dataMtxCeiling:** Mutex
ceiling = "30"
**samplingPeriod**:Property
name="samplingPeriod"
value="0,01"
**loggingPeriod**:Property
Business Configuration properties

## Conclusions

- Strategy for configuring the schedulability of component-based real-time applications
  - Keeping the opacity of components => Using only the metadata included in the RT-D&C descriptors
  - On top of the RT-CCM component technology:
    - The container and the environment services control the scheduling of the applications
  - The configuration values are obtained from the analysis of the real-time model of the application
    - Built by composition of the models of the components that form the application and the model of the execution platform
  - It has been implemented on Ada-CCM, an Ada 2005 implementation of RT-CCM