Orocos

Realtime Hybrid Task-Based Control for Robots and Machine Tools

Open Robot Control Software

Peter Soetens, Herman Bruyninckx

K.U.Leuven – Division PMA

www.orocos.org

What's in this Presentation

- RT-Orocos Overview
  - Open Real-Time Control Services
  - Open (RT-)Robot Control Software
- RT-Orocos Focus
  - Real-Time Task Specification
  - Deterministic Inter - Task Communication
  - Architecture for Control
- RT-Orocos Technical Features
  - Inter-Task Communication : Data, Commands, Events
  - Task Specification : Programs, Hierachical State Machines
  - Powerful Interactive 'TaskBrowser'
  - Task Persistency : Properties & XML
  - Controllers : The Control Kernel Architecture
- Conclusions

RT-Orocos

RealTime Orocos provides two application frameworks:

- Open Real-Time Control Services
  - Application Independent Machine Control *Infrastructure*
- Open Robot Control Software
  - Feedback Control *Architecture*

Open Realtime Control Services

Provides an ***application independent, portable*** control framework for one or more of :

- Realtime (feedback) control
  - Setpoint control and Command execution
- Deterministic realtime ↔ non-realtime communication
  - Data exchange and synchronisation
- Deterministic reaction to events
  - Reacting *synchronously* and *asynchronously*
- Hierachical and parallel state machines
- Runtime configuration (property system)

Open (RT-)Robot Control Software

Realtime feedback control for robotic applications :

- Multi axis *motion* control
- Kinematics and motion interpolation
- Velocity control, hybrid force/position control, impedance control ...
- Remote user interface (CORBA)
- Complex Task Specification
- Bayesian estimation (Kalman/particle filters)
  (via *Bayesian Filtering Library, K. Gadeyne*)

Orocos is ...

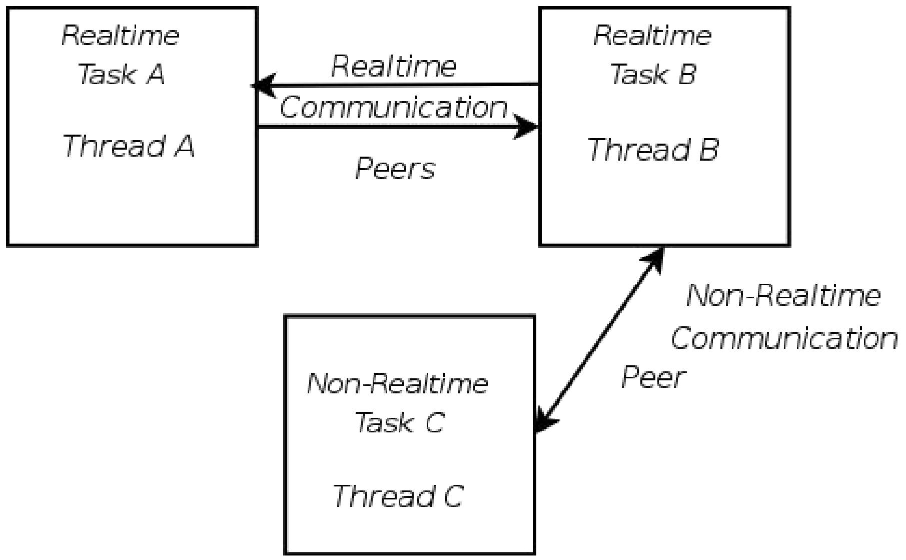
- *not* Formal verification of software
  - *yet* founded on formally verified concepts and runtime verified by testing
- *not* 'Black Box' or expensive (lock-in) solution
  - *yet* an Open and Free *Framework* for *Integration* of Control Solutions
- *not* a 'one-size-fits-all' solution !
  - *yet* a multitude of building blocks and possibilities

RT-Orocos Focus

- Real-Time Task Specification
  - Define your *interactive* tasks which make up your application
- Deterministic Inter-Task Communication
  - The Framework manages your communication needs
- Architecture for Control
  - Apply the Control Services to numerical feedback controllers

Focus:*Realtime Task Specification*

- Tasks form a 'network'
- Task Interface :
  - **Data**
  - **Commands Methods**
  - **Events**
  - **Attributes Properties**



- Tasks run in parallel threads which :
  - Process the data flow
  - Execute commands
  - Execute hierarchical state machines

Focus:*Deterministic Inter-Task Communication*

- Integral *Lock-Free* Communication
  - No process is forced to block
  - No *dead-locks* or *priority-inversions*

...due to communication

- Applied in Orocos for :
  - Data exchange (of any C++ type)
  - Passing of Commands (Objects)
  - Event handling (via Callback functions)
  - Real-Time program script execution

Focus:*Architecture for Control*

- Define reusable Components for Control
  - Controllers, Sensors, setpoint Generators,...
  - Store/Load configuration from XML files
  - Real-Time Component switching
  - Data capturing : Export calculated data to files
  - Execute State Machines and Program Scripts

- Uses the Real-Time Control Services for :
  - Data exchange (of any C++ type)
  - Accepting Commands (from user or other task)
  - Configuration (Properties/XML)
  - Real-Time program script execution

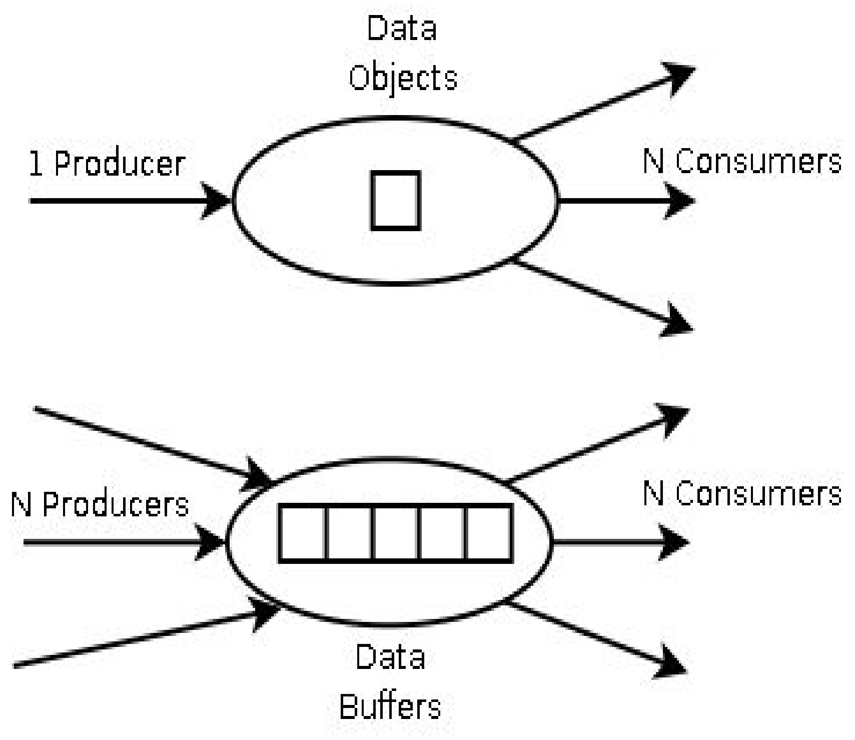
Technical Features

- Inter-Task Communication :
  - Data, Buffers, Commands, Methods, Events
- Script Execution :
  - Programs
  - Hierachical State Machines
- Powerful Interactive 'TaskBrowser'
  - Inspect and modify real-time tasks online.
- Task Persistency
  - Configuration with Property sets
  - Save/Load from XML
- Controllers : The Control Kernel Architecture
  - Multi-Axis Controllers, Data Capturing, GUI Frontend

Inter-Task Communication : Data

Aim : Lock-free data exchange via

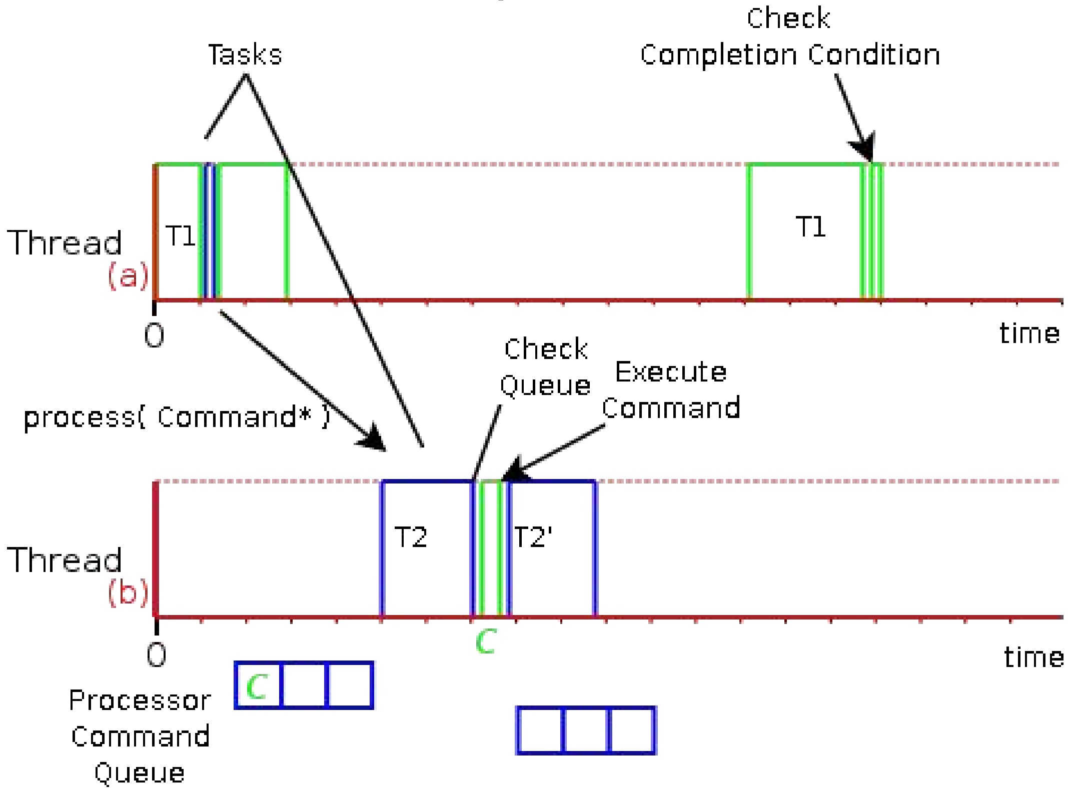
- DataObjects (Last)
  - positions
  - parameters
- Buffers (FIFO)
  - measurements
  - setpoints



Inter-Task Communication : Commands

Aim :

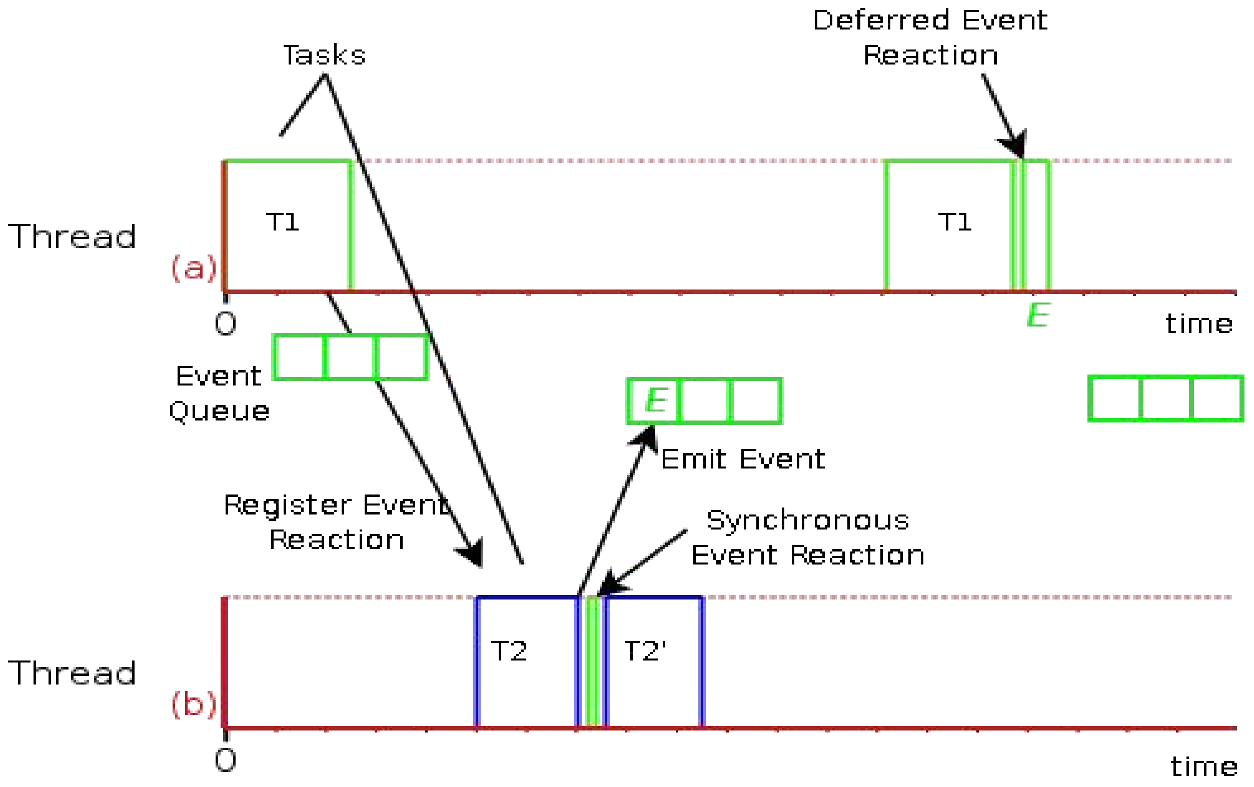
- Asynchronously call a task-function
- Poll / wait on its completion condition



Inter-Task Communication : Events

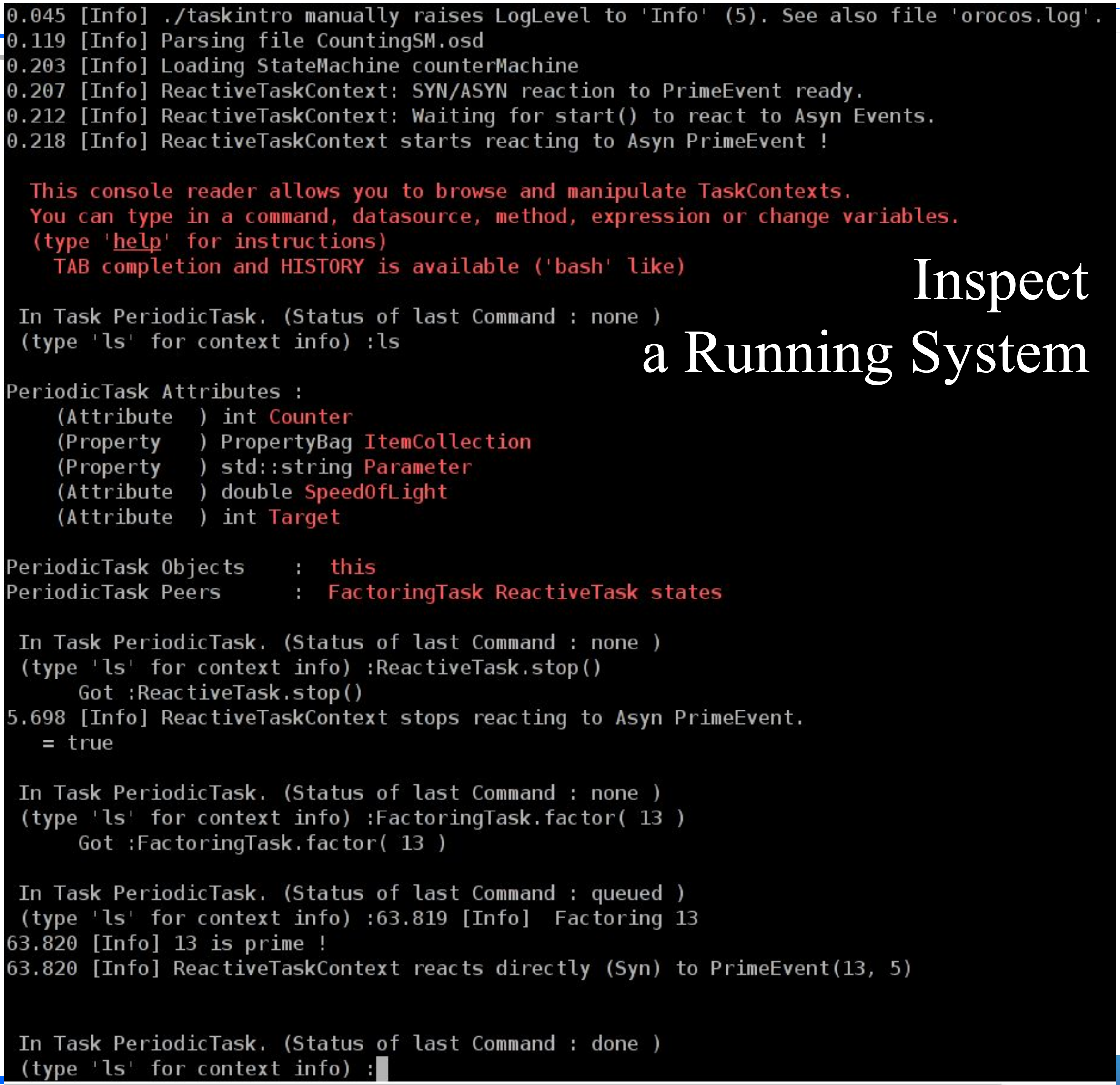
Aim :

- Synchronously and/or Asynchronously react to events
- Detection in 1 location, reaction on N locations



Powerful Interactive 'TaskBrowser'

- Tool for application designers
- Inspect and modify real-time tasks online.
- Send Commands to peer tasks
- Minimal setup effort



```
0.045 [Info] ./taskintro manually raises LogLevel to 'Info' (5). See also file 'orocos.log'.
0.119 [Info] Parsing file CountingSM.osd
0.203 [Info] Loading StateMachine counterMachine
0.207 [Info] ReactiveTaskContext: SYN/ASYN reaction to PrimeEvent ready.
0.212 [Info] ReactiveTaskContext: Waiting for start() to react to Asyn Events.
0.218 [Info] ReactiveTaskContext starts reacting to Asyn PrimeEvent !

  This console reader allows you to browse and manipulate TaskContexts.
  You can type in a command, datasource, method, expression or change variables.
  (type 'help' for instructions)
    TAB completion and HISTORY is available ('bash' like)

 In Task PeriodicTask. (Status of last Command : none )
 (type 'ls' for context info) :ls

PeriodicTask Attributes :
    (Attribute  ) int Counter
    (Property   ) PropertyBag ItemCollection
    (Property   ) std::string Parameter
    (Attribute  ) double SpeedOfLight
    (Attribute  ) int Target

PeriodicTask Objects    :  this
PeriodicTask Peers      :  FactoringTask ReactiveTask states

 In Task PeriodicTask. (Status of last Command : none )
 (type 'ls' for context info) :ReactiveTask.stop()
      Got :ReactiveTask.stop()
5.698 [Info] ReactiveTaskContext stops reacting to Asyn PrimeEvent.
   = true

 In Task PeriodicTask. (Status of last Command : none )
 (type 'ls' for context info) :FactoringTask.factor( 13 )
      Got :FactoringTask.factor( 13 )

 In Task PeriodicTask. (Status of last Command : queued )
 (type 'ls' for context info) :63.819 [Info]  Factoring 13
63.820 [Info] 13 is prime !
63.820 [Info] ReactiveTaskContext reacts directly (Syn) to PrimeEvent(13, 5)


 In Task PeriodicTask. (Status of last Command : done )
 (type 'ls' for context info) :
```

Inspect
a Running System

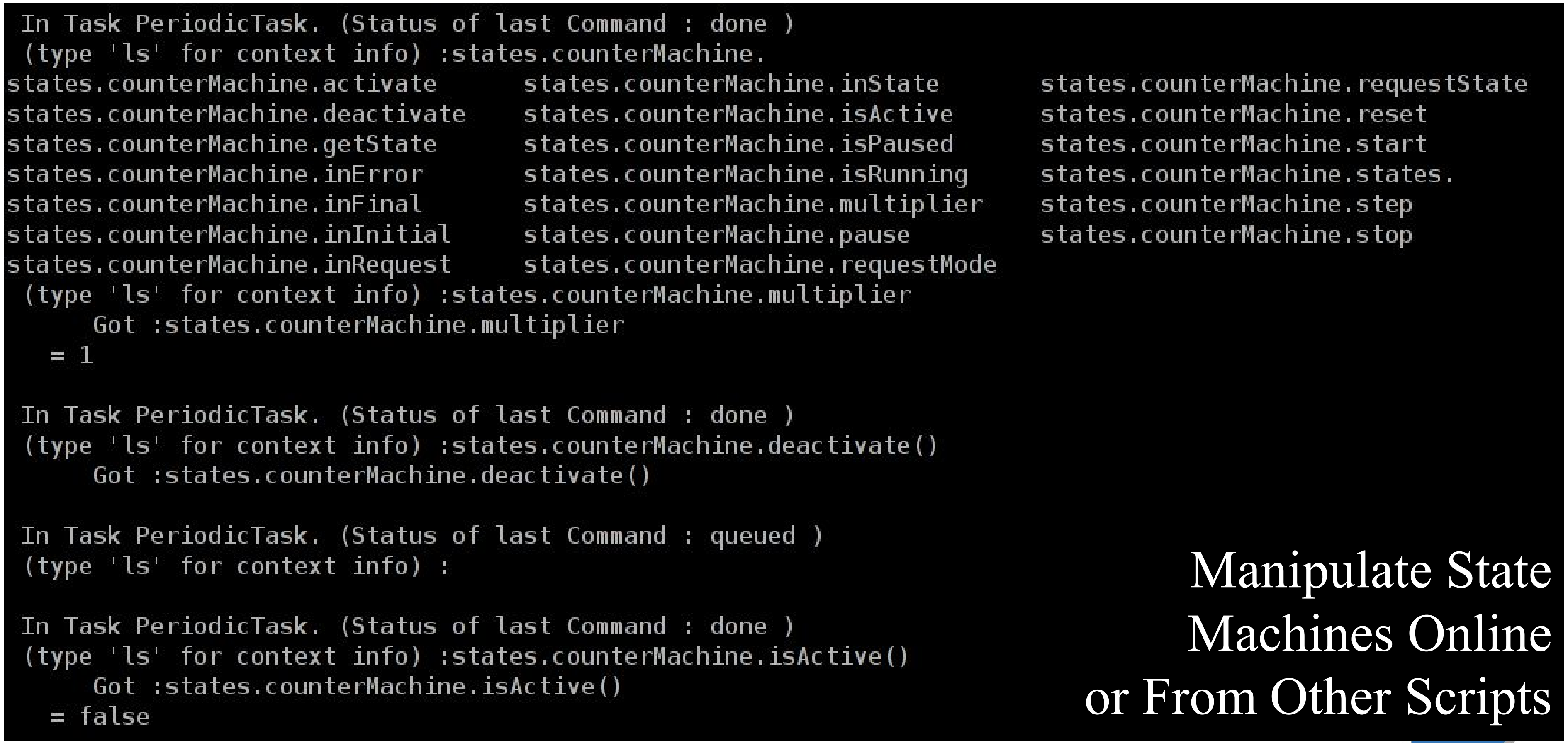
KATHOLIEKE UNIVERSITEIT LEUVEN

www.orocos.org

PMA

Script Execution

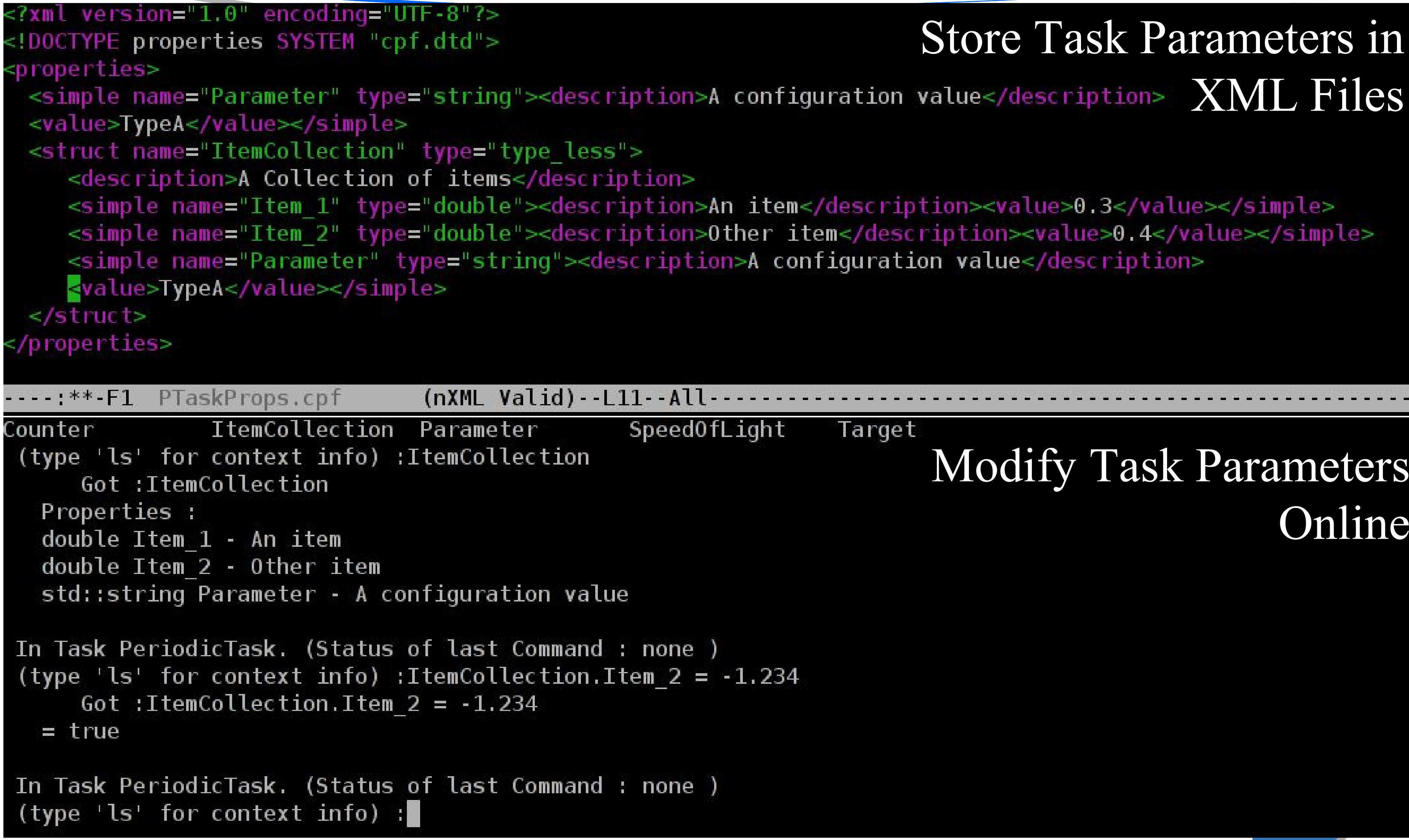
Load Program and Hierachical State Machine scripts online,control and inspect their execution status and even local variables.



```
 In Task PeriodicTask. (Status of last Command : done )
 (type 'ls' for context info) :states.counterMachine.
states.counterMachine.activate      states.counterMachine.inState       states.counterMachine.requestState
states.counterMachine.deactivate    states.counterMachine.isActive      states.counterMachine.reset
states.counterMachine.getState      states.counterMachine.isPaused      states.counterMachine.start
states.counterMachine.inError       states.counterMachine.isRunning     states.counterMachine.states.
states.counterMachine.inFinal       states.counterMachine.multiplier    states.counterMachine.step
states.counterMachine.inInitial     states.counterMachine.pause         states.counterMachine.stop
states.counterMachine.inRequest     states.counterMachine.requestMode
 (type 'ls' for context info) :states.counterMachine.multiplier
      Got :states.counterMachine.multiplier
   = 1

 In Task PeriodicTask. (Status of last Command : done )
 (type 'ls' for context info) :states.counterMachine.deactivate()
      Got :states.counterMachine.deactivate()

 In Task PeriodicTask. (Status of last Command : queued )
 (type 'ls' for context info) :

 In Task PeriodicTask. (Status of last Command : done )
 (type 'ls' for context info) :states.counterMachine.isActive()
      Got :states.counterMachine.isActive()
   = false
```

Manipulate State Machines Online or From Other Scripts

Task Persistency - XML



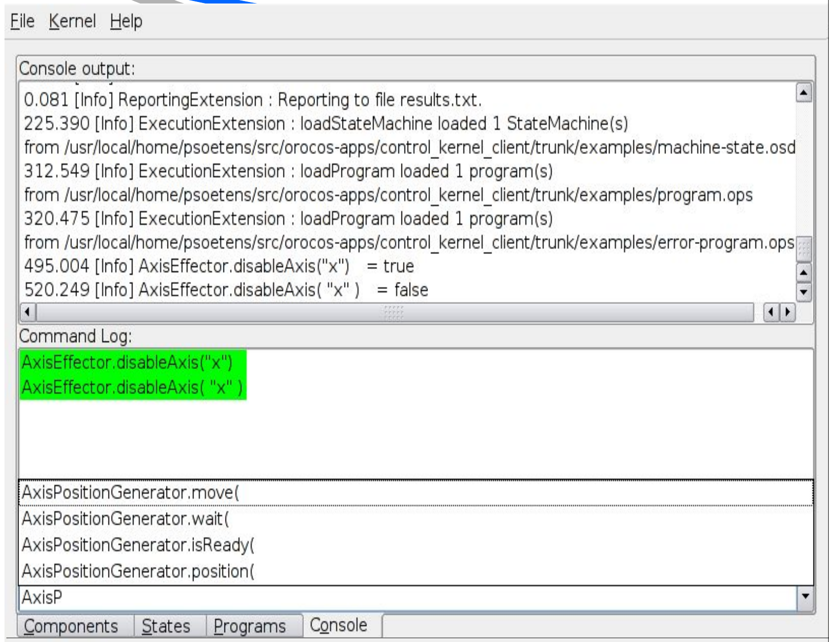
Store Task Parameters in XML Files

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE properties SYSTEM "cpf.dtd">
<properties>
  <simple name="Parameter" type="string"><description>A configuration value</description>
  <value>TypeA</value></simple>
  <struct name="ItemCollection" type="type_less">
     <description>A Collection of items</description>
     <simple name="Item_1" type="double"><description>An item</description><value>0.3</value></simple>
     <simple name="Item_2" type="double"><description>Other item</description><value>0.4</value></simple>
     <simple name="Parameter" type="string"><description>A configuration value</description>
     <value>TypeA</value></simple>
  </struct>
</properties>

----:**-F1  PTaskProps.cpf       (nXML Valid)--L11--All-------------------------------------------
```

Modify Task Parameters Online

```
Counter          ItemCollection  Parameter       SpeedOfLight    Target
 (type 'ls' for context info) :ItemCollection
      Got :ItemCollection
   Properties :
   double Item_1 - An item
   double Item_2 - Other item
   std::string Parameter - A configuration value

 In Task PeriodicTask. (Status of last Command : none )
 (type 'ls' for context info) :ItemCollection.Item_2 = -1.234
      Got :ItemCollection.Item_2 = -1.234
   = true

 In Task PeriodicTask. (Status of last Command : none )
 (type 'ls' for context info) :
```

The Control Kernel GUI



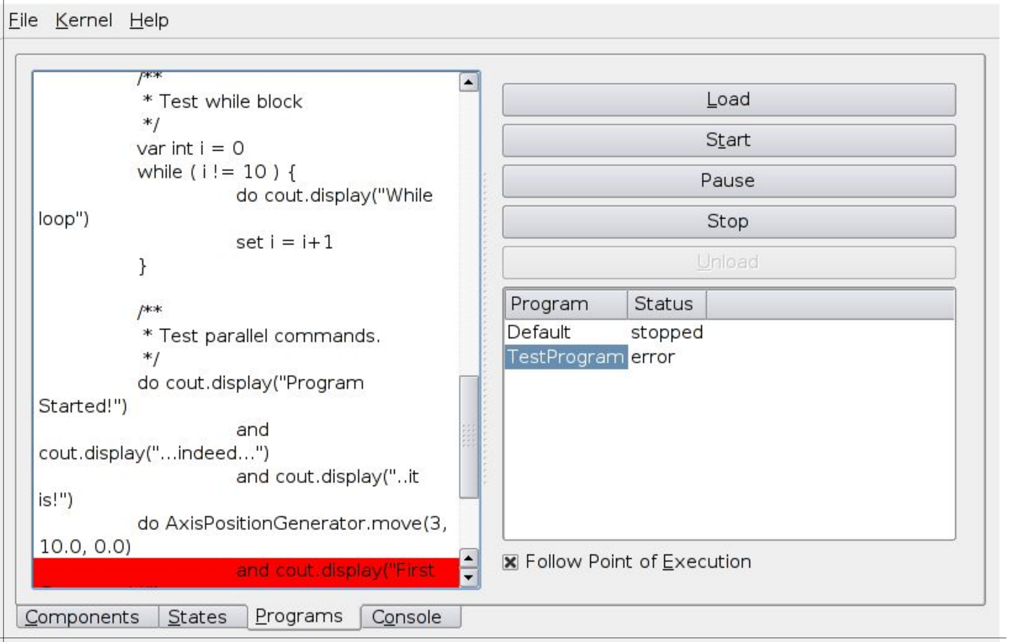
Console output:

```
0.081 [Info] ReportingExtension : Reporting to file results.txt.
225.390 [Info] ExecutionExtension : loadStateMachine loaded 1 StateMachine(s)
from /usr/local/home/psoetens/src/orocos-apps/control_kernel_client/trunk/examples/machine-state.osd
312.549 [Info] ExecutionExtension : loadProgram loaded 1 program(s)
from /usr/local/home/psoetens/src/orocos-apps/control_kernel_client/trunk/examples/program.ops
320.475 [Info] ExecutionExtension : loadProgram loaded 1 program(s)
from /usr/local/home/psoetens/src/orocos-apps/control_kernel_client/trunk/examples/error-program.ops
495.004 [Info] AxisEffector.disableAxis("x")   = true
520.249 [Info] AxisEffector.disableAxis( "x" )   = false
```

Command Log:

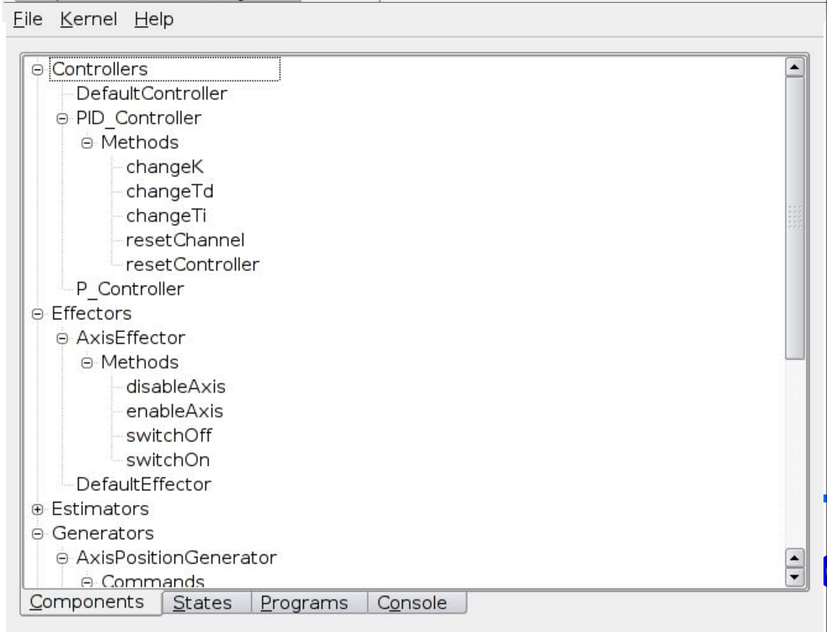
```
AxisEffector.disableAxis("x")
AxisEffector.disableAxis( "x" )
```

```
AxisPositionGenerator.move(
AxisPositionGenerator.wait(
AxisPositionGenerator.isReady(
AxisPositionGenerator.position(
AxisP
```

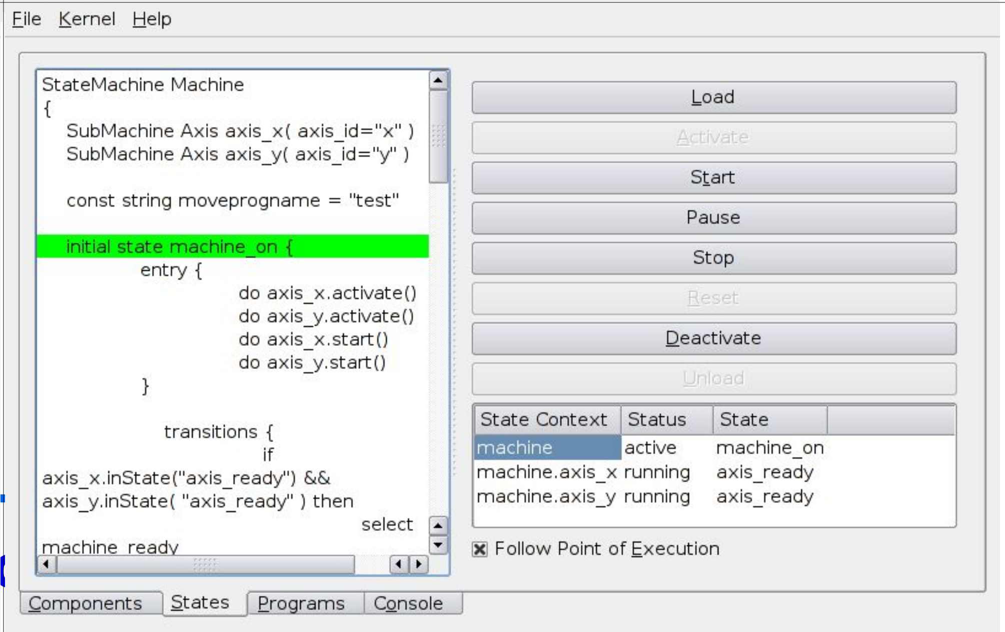


```
          /**
           * Test while block
           */
          var int i = 0
          while ( i != 10 ) {
                    do cout.display("While
loop")
                    set i = i+1
          }

          /**
           * Test parallel commands.
           */
          do cout.display("Program
Started!")
                    and
cout.display("...indeed...")
                    and cout.display("..it
is!")
          do AxisPositionGenerator.move(3,
10.0, 0.0)
                    and cout.display("First
```

Program	Status
Default	stopped
TestProgram	error



```
⊖ Controllers
    DefaultController
  ⊖ PID_Controller
    ⊖ Methods
        changeK
        changeTd
        changeTi
        resetChannel
        resetController
    P_Controller
⊖ Effectors
  ⊖ AxisEffector
    ⊖ Methods
        disableAxis
        enableAxis
        switchOff
        switchOn
    DefaultEffector
⊕ Estimators
⊖ Generators
  ⊖ AxisPositionGenerator
    ⊖ Commands
```



```
StateMachine Machine
{
   SubMachine Axis axis_x( axis_id="x" )
   SubMachine Axis axis_y( axis_id="y" )

   const string moveprogname = "test"

   initial state machine_on {
          entry {
                    do axis_x.activate()
                    do axis_y.activate()
                    do axis_x.start()
                    do axis_y.start()
          }

              transitions {
                        if
axis_x.inState("axis_ready") &&
axis_y.inState( "axis_ready" ) then
                                  select
machine_ready
```

State Context	Status	State
machine	active	machine_on
machine.axis_x	running	axis_ready
machine.axis_y	running	axis_ready

Conclusions

- Open Source Hard Realtime Control is available
- Cooperation with several machinetool vendors
- Both realtime and non realtime Linux supported
- *RT-Orocos* currently integrates with
  - RTAI (realtime Linux)
  - Comedi (data acquisition)
  - RTNet (realtime ethernet)
- Future integration opportunities e.g.:
  - Player/Stage (as server)
  - CORBA middleware (Orca, Miro,...)