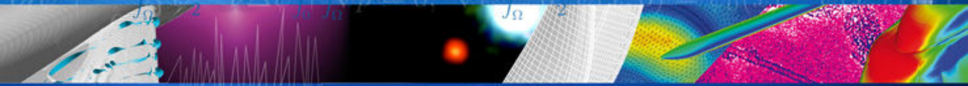


ONERA

THE FRENCH AEROSPACE LAB

retour sur innovation

www.onera.fr

Validation of real-time properties of a robotic software architecture

Charles Lesire (Onera - DCSD), David Doose (Onera - DTIM),
Hugues Cassé (IRIT)

CAR 2011 — Grenoble, France — May 2011

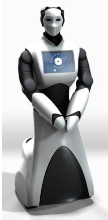
ONERA

THE FRENCH AEROSPACE LAB

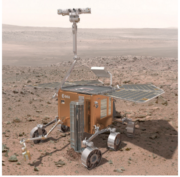
retour sur innovation

Motivations

- Robots are critical systems that must be **safe**, otherwise:
  - they may hurt people,




  - they may fail and be unusable.

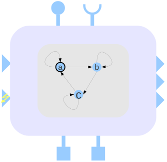


Motivations

- The temporal constraints are crucial in the safety analysis:
  - embedded software are designed to be executed at specific rates,
  - any overshooting of software deadlines could disturb the system behavior;
- The **schedulability** analysis allows to check offline that all the tasks will be executed on time;
- Schedulability analysis in robotics usually consists in **measuring** the response time of embedded software;
- Formal schedulability analysis in embedded systems based on WCRT computation;

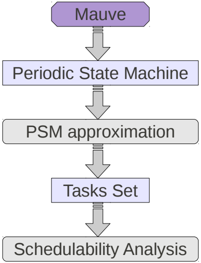
Plan

The Mauve DSL



- Component model approach
  - Service: provides *operations* and requires *methods*
  - Ports: oriented data communication
  - Properties: set of component parameters
  - Real-time properties: period, deadline, priority
  - Behavior: defined by a finite state machine
  - call elementary processing functions, **codels**
- Components allocated to tasks
- Prototype created using Eclipse MDT

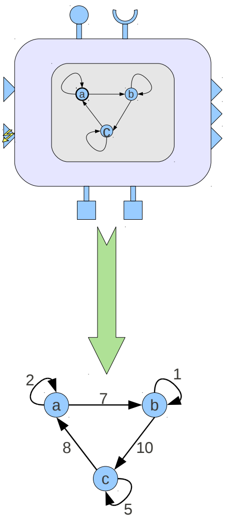
Validation process



Transform an instance of the Mauve DSL into a classical tasks model:

- Extract the component behavior into a simple formalism: PSM (Periodic State Machine)
- Convert each PSM representing a component into a classical task representation
- Analyze the system schedulability

Mauve to PSM



PSM:

- Transition : computation time
- Activation : one transition

PSM Transformation:

- Component behavior (finite state machine)
- Component communication (methods & operations)
- Component mode
- Component dynamic properties

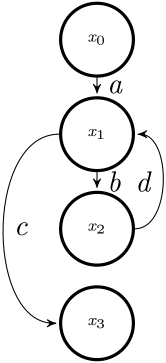
WCET computation

- Estimation of the Worst Case Execution Time of component codels
- Take into account architecture specificities (caches, pipelines)
- Analyze the assembly code:
  - build the codel Control Flow Graph (CFG)
  - solve a Integer Linear Programming system

WCET computation

Example

```
int sum(int t[100]) {
  int i, s;
  s = 0;
  for(i = 0; i < 100; i++)
    s += t[i]
  return s;
}
```

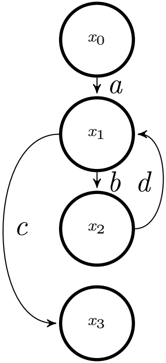


WCET computation

Example

```
int sum(int t[100]) {
  int i, s;
  s = 0;
  for(i = 0; i < 100; i++)
    s += t[i]
  return s;
}
```

$$
\begin{aligned}
WCET &= MAX(t_0 x_0 + t_1 x_1 + t_2^h x_2^h + t_2^m x_2^m + t_3 x_3) \\
x_0 &= 1 \\
x_1 &= a + d = b + c \\
x_2 &= b = d \\
x_3 &= c \\
d &\leq 100 \\
x_2 &= x_2^h + x_2^m \\
x_2^m &\leq 1
\end{aligned}
$$



PSM to Tasks

Classical tasks model:

- Monoprocessor
- Scheduler: Fixed Priority (FP, RM, DM)
- Task
  - Worst Case Execution Time (C_i)
  - Priority (P_i)
  - Period (T_i)
  - Deadline (D_i)

Transformation:

- Compute "all" the PSM timelines
- Compute an approximation of the timelines
- PSM approximation: set of instances of the same task

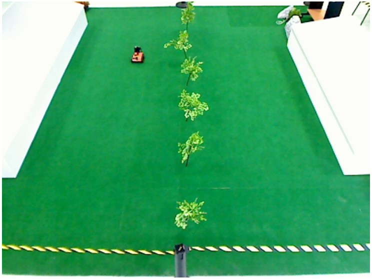
Schedulability analysis

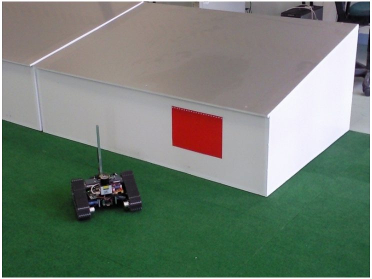
- Compute task (i.e. component) worst case response time ($\mathcal{R}_i$)
- Modification of the classical worst case response time computation in order to take into account tasks instances
- A task is schedulable iff $\mathcal{R}_i \leq D_i$

The fix point computation is defined by the following process:

1. $\mathcal{R}_i^0 = C_{i,1}$
2. $\mathcal{R}_i^{n+1} = C_{i,1} + \sum_{(j,l),j \in hp(i),l=1..k_j / r_{j,l} \leq \mathcal{R}_i^n} C_{j,l}$
3. If $\mathcal{R}_i^{n+1} \geq D_i$, the deadline is exceeded;
4. If $\mathcal{R}_i^{n+1} = \mathcal{R}_i^n$ then $\mathcal{R}_i = \mathcal{R}_i^n$
5. Otherwise, $\mathcal{R}_i^n := \mathcal{R}_i^{n+1}$ and go back to step 2.

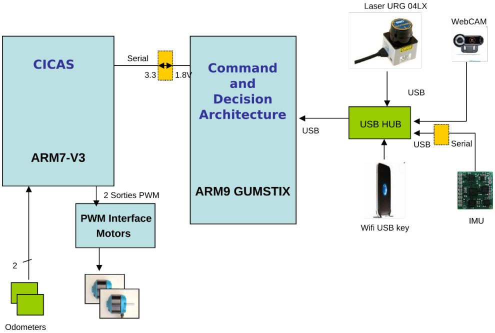
Illustrative example





Illustrative example

Hardware architecture

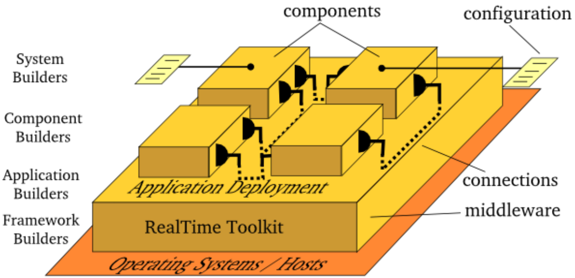


- The Command and Decision Architecture runs Linux with the Xenomai RT patch;
- The software architecture is built over **Orocos**.

Orocos

Real-Time Toolkit

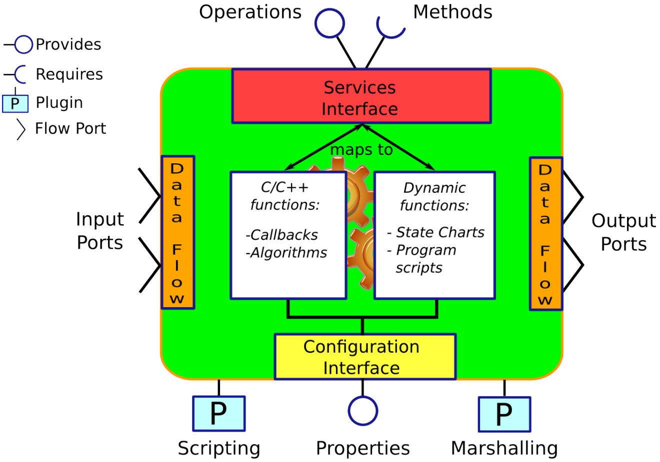
- Orocos/RTT (Real-Time Toolkit): an open-source library for developing and deploying real-time components



http://www.orocos.org

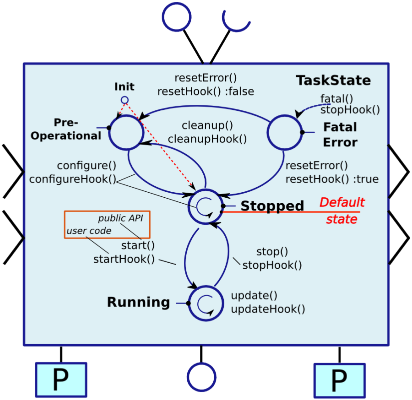
Orocos

Component models



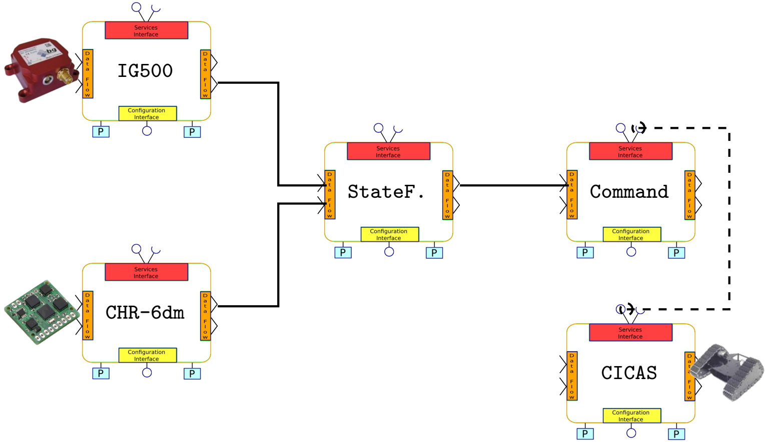
Orocos

Component models



Illustrative example

Component architecture



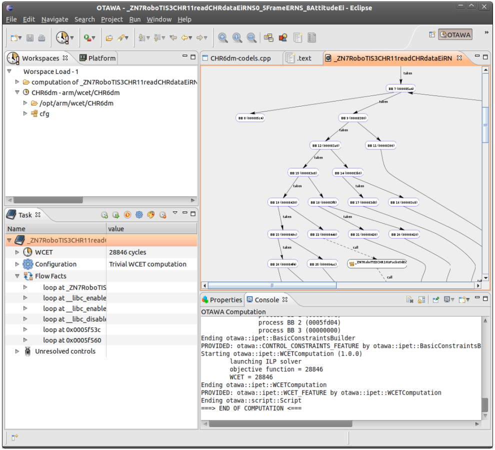
Illustrative example

Mauve models

Component	Period (ms)	Priority	Codel
CICAS	-	-	send
CHR-6dm	1	1	update
IG500	10	2	update
StateFusion	10	3	update
Command	10	4	update
			Rotating
			Reaching

Schedulability results

WCET computation with Otawa



http://www.otawa.fr

Schedulability results

Component	T (ms)	Pr.	Codel	WCET (μs)	WCRT (μs)
CICAS	-	-	send	$5'512$	-
CHR-6dm	1	1	update	145	145
IG500	10	2	update	1	146
StateFusion	10	3	update	2	413
Command	10	4	update	$5'324$	$6'607$
			Rotating	69	
			Reaching	173	

- The system is **schedulable**;
- The processor load is about 67%.

Conclusion

- Mauve: a DSL for component-based (robotic) systems
- Direct mapping into Orocos/RTT (a robotic framework)
- Codel WCET computation with Otawa
- Component WCRT computation and schedulability results

Future work

- Enhance the example architecture with more complex components
  - vision-based object recognition
  - laser-based SLAM
  - motion planning
  - task planning
- Integrate Orocos primitives into the WCET/WCRT analysis
  - Data exchange
  - Operation calls
  - Task management
  - etc.
- Generate (Orocos) code from Mauve specifications