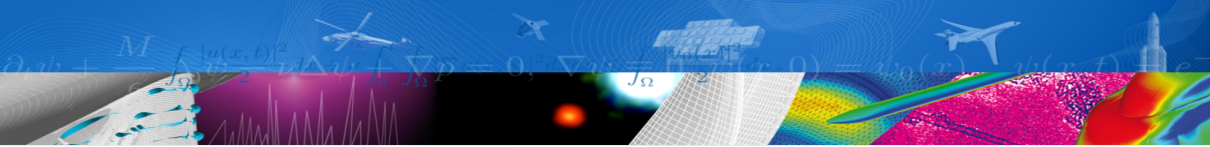


Formal models for verification and control of software architectures

Charles Lesire[1,2]

[1] ONERA/DTIS, University of Toulouse, France

[2] Artificial and Natural Intelligence Toulouse Institute (ANITI)

IEEE TC-VAS – 1 October 2020

CC BY NC ND

ONERA

THE FRENCH AEROSPACE LAB

ANITI

ARTIFICIAL & NATURAL INTELLIGENCE TOULOUSE INSTITUTE

charles.lesire@onera.fr

Context

Photo by **Matthew Henry**

BY-NC-ND

Context

mission programming

©Charles Lesire

Photo by **Matthew Henry**

BY-NC-ND

Context

mission programming

©Charles Lesire

mission-level guarantees?

software-level guarantees?

Photo by **Matthew Henry**

BY-NC-ND

Outline

BY-NC-ND

Outline

BY-NC-ND

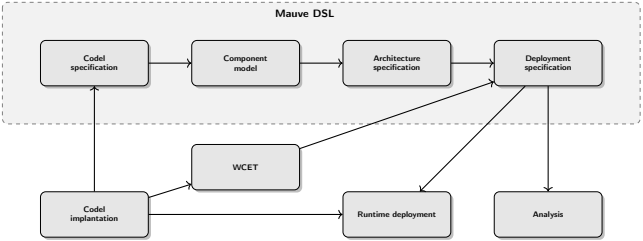
The MAUVE Toolchain

- MAUVE DSL to model component-based architectures
- MAUVE Runtime to execute real-time architectures
- MAUVE RT Analyses to evaluate WCET/WCRT of components

BY-NC-ND

MAUVE DSL and Toolchain

- Architecture modeling using a DSL[1]
- Code Generation (ROS/OROCOS)
- Model Analysis compliant with Execution Analysis



[1] Nicolas Gobillot et al. "A Design and Analysis Methodology for Component-Based Real-Time Architectures of Autonomous Systems". In: *J. Intell. Robotic Syst.* 96.1 (2019), pp. 123–138. doi: 10.1007/s10846-018-0967-5.

BY-NC-ND

MAUVE Runtime

MAUVE Runtime[2] rationales:

1. Provide a C++ API for programmers compliant with models
2. Provide reconfiguration mechanisms
3. Masterize the synchronization of real-time tasks
4. Provide an execution model both formal (to ease analyses) and expressive (to allow implementing complex behaviors)

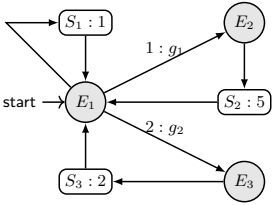
[2] David Doose et al. "MAUVE Runtime: A Component-Based Middleware to Reconfigure Software Architectures in Real-Time". In: *IEEE International Conference on Robotic Computing, (IRC)*. Taichung, Taiwan: IEEE Computer Society, 2017. doi: 10.1109/IRC.2017.47.

BY-NC-ND

MAUVE Runtime

A component-based architecture middleware to design architectures with:

- components, i.e. tasks that execute code
- resources, that own data
- connections between components and resources
- real-time activities assigned to components
- mechanisms to reconfigure parts of the architecture in real-time



BY-NC-ND

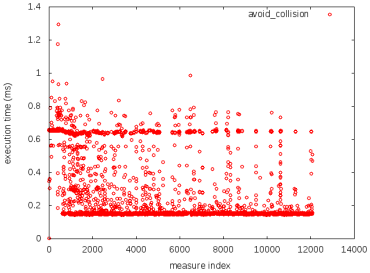
MAUVE RT analysis

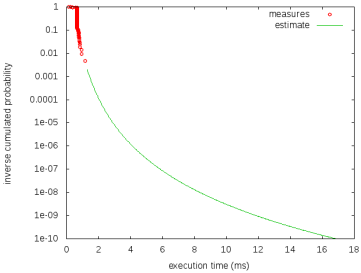
- Schedulability analysis:
  - Determine if the software components are executed on time.
- Worst Case Execution Time (WCET):
  - longest CPU time passed to compute a piece of code without any interaction (alone on the CPU)
  - depends on both the source code and the hardawre.
  - an input for the schedulability analysis and the computation of the WCRT
- Worst Case Response Time (WCRT):
  - longest time spent by a "task" from its beginning to its end
  - takes into account preemptions and delays from other "tasks"
  - A "task" is schedulable is its WCRT est lower or equals to its deadline

BY-NC-ND

MAUVE RT analysis

- WCET using probabilistic analysis[3]
  - gather real executions traces of the robot
  - Extreme Value Theory (EVT) to infer rare events
  - computes a probabilistic WCET
  - metrics on the applicability of the theory





[3] Nicolas Gobillot et al. "Measurement-based real-time analysis of robotic software architectures". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Daejeon, South Korea, 2016. doi: 10.1109/IROS.2016.7759509.

BY-NC-ND

Outline

BY-NC-ND

Robot Skills Models

- Organize the software architecture into functions, or skills

[4] Charles Lesire et al. "Formalization of Robot Skills with Descriptive and Operational Models". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Las Vegas, NV, USA, 2020.

BY-NC-ND

Robot Skills Models

- Organize the software architecture into functions, or skills

⇝ DSL for Skill Modeling... and tools[4]

[4] Charles Lesire et al. "Formalization of Robot Skills with Descriptive and Operational Models". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Las Vegas, NV, USA, 2020.

BY-NC-ND

Robot Skills Models

- Organize the software architecture into functions, or skills

⇝ DSL for Skill Modeling... and tools[4]

Photo by Matthew Henry

Skill models

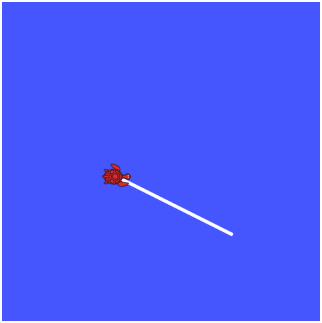
©Charles Lesire

[4] Charles Lesire et al. "Formalization of Robot Skills with Descriptive and Operational Models". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Las Vegas, NV, USA, 2020.

BY-NC-ND

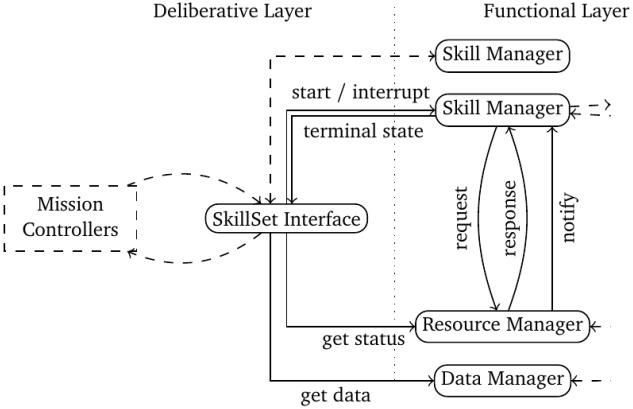
Robot Skills DSL: Turtlesim

```
type {
  Pose
  Pen
  Bool
}

skillset turtlesim {
  data position: Pose

  function isSafe(p: Pose): Bool

  resource simulator {
    initial UNAVAILABLE
    extern UNAVAILABLE -> AVAILABLE
    extern AVAILABLE -> UNAVAILABLE
  }

  skill teleport {
    progress=0
    input target: Pose
    precondition sim_avail: resource=(simulator==AVAILABLE)
    invariant sim_inv: resource=(simulator==AVAILABLE)
    mode {
      ARRIVED {}
      COLLISION {}
    }
  }
```



BY-NC-ND

Robot Skills Execution Tools



BY-NC-ND

Robot Skills Tools

and also:

- interface library for integration into StateMachines and BehaviorTree models
- translation towards PDDL models
- translation towards NuSMV State Machines

BY-NC-ND

Outline

BY-NC-ND

ASPiC

- Task-level (or Mission) Programming:
  - Expressive "language" with complex operations
  - Management of failures in task execution
  - Petri nets as an underlying formal model
  - Execution control based on playing the resulting Petri net model

[5] Charles Lesire and Franck Pommereau. "ASPiC: An Acting System Based on Skill Petri Net Composition". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid, Spain, 2018. doi: 10.1109/IROS.2018.8594328.

BY-NC-ND

ASPiC

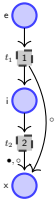
- Task-level (or Mission) Programming:
  - Expressive "language" with complex operations
  - Management of failures in task execution
  - Petri nets as an underlying formal model
  - Execution control based on playing the resulting Petri net model
- ASPiC[5]:
  - Petri nets with Control-Flow semantics
  - Colored PN with control-flow tokens • and ○
  - Control-flow places $P^{\bullet\circ}$: entry, internal, exit
  - Modelling of *skills* using specific CFPN
  - composition operators that (tend to) preserve properties by construction

[5] Charles Lesire and Franck Pommereau. "ASPiC: An Acting System Based on Skill Petri Net Composition". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid, Spain, 2018. doi: 10.1109/IROS.2018.8594328.

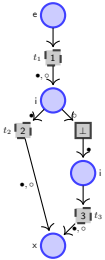
BY-NC-ND

ASPiC Operators

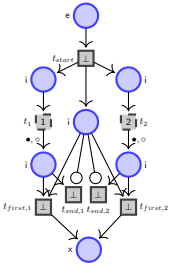
- Composition operators to build up plans from skills
- Integrate exception propagation
- Each operator modelled as a CFPN with placeholder transitions
- Operators: sequence, choice, concurrency, if/then/else, race concurrency, loop/retry



$N_{\odot}$



N_{ite}



$N_{\otimes}$

BY-NC-ND

ASPiC Properties

- Well-formedness
  - preserved when connecting with a handler
  - preserved by operators
- Cleanness of control-flow

$$\forall M : M_0 \xrightarrow{*} M \wedge M(p_{\mathsf{x}}) \neq \emptyset \Rightarrow$$
$$M(p_{\mathsf{x}}) \in \{\{\bullet\}, \{\circ\}\} \wedge (p \in P^{\bullet} \setminus \{p_{\mathsf{x}}\} \Rightarrow M(p) = \emptyset)$$

- Control-safe: $P^{\bullet}$ is safe
- Cleanness and control-safety are not ensured by all operators (typically $N_{\otimes}$)

BY-NC-ND

Conclusion

Being able to guarantee a correct behavior, from mission-level to functional components, with models compliant to the actual execution.

BY-NC-ND