

# Using Aspect-GAMMA in the Design of Embedded Systems

M. Mousavi, G. Russello, M. Chaudron, M. Reniers, T. Basten  
A. Corsaro, S. Shukla, R. Gupta, D. C. Schmidt

Eindhoven University of Technology  
University of California at Irvine, and San Diego  
Virginia Tech.

# Overview

1. Introduction
2. Exploring Aspects
3. Case study
4. Conclusion and Future Works

# Introduction

1. Separation of Concerns:
  - (a) Focused Design
  - (b) Lighter Verification
  - (c) Localized Change
  - (d) Abstraction from Modelling Methods
2. AoP and Multi-Dimensional SoC Programming

# Contributions

1. Support for Separation of Concerns
2. Extension to Design Phase
3. Establishing a Formal Basis

# Exploring Aspects: Roadmap

Aspects of Distributed Real-Time System Design:

- Computation

# Exploring Aspects: Roadmap

Aspects of Distributed Real-Time System Design:

- Computation
- Coordination

# Exploring Aspects: Roadmap

Aspects of Distributed Real-Time System Design:

- Computation
- Coordination
- Timing

# Exploring Aspects: Roadmap

Aspects of Distributed Real-Time System Design:

- Computation
- Coordination
- Timing
- Distribution

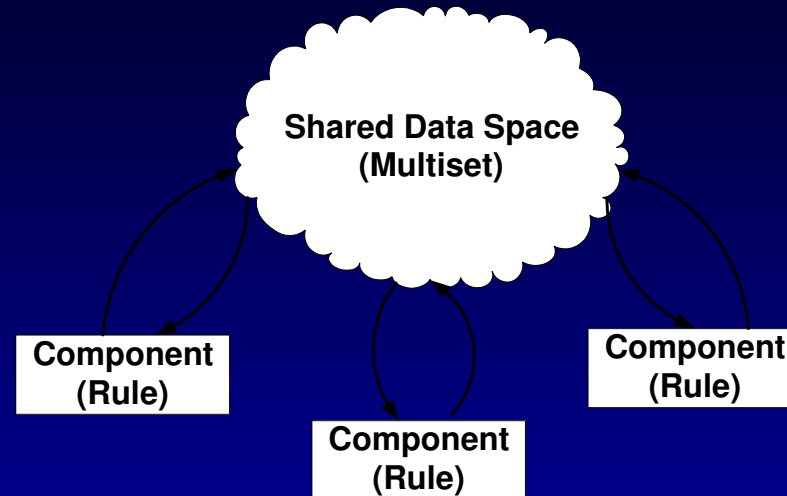
# Exploring Aspects: Roadmap

Aspects of Distributed Real-Time System Design:

- Computation
- Coordination
- Timing
- Distribution
- Hardware Resources, etc...

# Distributed Real-Time Design

- Computation: GAMMA



$\text{RuleName} = \text{MultisetExp} \mapsto \text{MultisetExp} \Leftarrow \text{Condition}$

# Distributed Real-Time Design

- Computation: GAMMA
- Coordination: Schedules

Schedule ::= RuleName |  
RuleName  $\curvearrowright$  Schedule |  
Schedule ; Schedule |  
Schedule || Schedule |  
 $\mu$ RecursionVar. Schedule |  
RecursionVar

# Distributed Real-Time Design

- Computation: GAMMA
- Coordination: Schedules
- Timing: Intervals

$$T_{\text{RuleName}} = \text{Interval}$$

# Distributed Real-Time Design

- Computation: GAMMA
- Coordination: Schedules
- Timing: Intervals
- Distribution: Relations, mapping data types and functionalities to locations

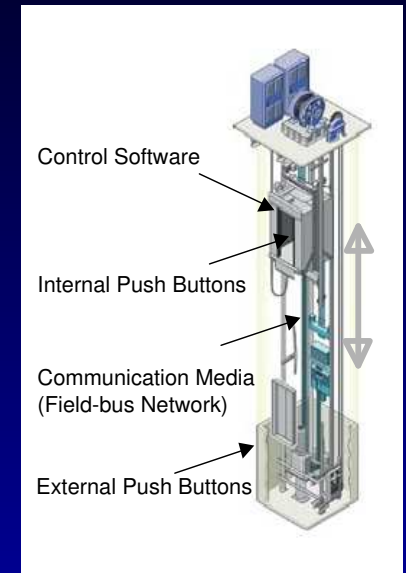
StaticDist :  $(R \cup T) \rightarrow \mathcal{P}(P)$ ,

R : The Set of Rule Names,

T : The Set of Data Types,

P : The Set of Location

# Elevator: Functionality

$$\begin{aligned} \text{ElevatorSystem} &= \{ \\ \text{extRequest} &= ((\text{extStop}, i), \text{off}) \mapsto \\ &\quad ((\text{extStop}, i), \text{on}), \\ \text{moveUp} &= (\text{cf}, i) \mapsto (\text{cf}, i + 1) \Leftarrow \dots, \\ \text{load} &= ((\text{extStop}, i), \text{on}) \mapsto ((\text{extStop}, i), \text{off}) \Leftarrow \dots \\ \text{unload} &= ((\text{inStop}, i), \text{on}) \mapsto ((\text{inStop}, i), \text{off}) \Leftarrow \dots \} \end{aligned}$$


# Elevator: Coordination

$$\text{ElevatorSchedule} = \text{Environment} \parallel$$
$$(\mu X. \text{ServiceUp} ; \text{ServiceDown}; X)$$
$$\text{Environment} = (\mu X. \text{inRequest} \parallel X) \parallel$$
$$(\mu X. \text{extRequest} \parallel X)$$
$$\text{ServiceUp} = \mu X. ((\text{load} \parallel \text{unload}) ;$$
$$(\text{moveUp} \curvearrowright (\text{moveUp} ; X)))$$

# Elevator: Timing

$$T_{\text{inRequest}} = T_{\text{extRequest}} = [0, 0]$$

$$T_{\text{moveUp}} = T_{\text{moveDown}} = [\text{StepTime}, \text{StepTime}]$$

$$T_{\text{load}} = T_{\text{unload}} = [\text{MinService}, \text{MaxService}]$$

# Elevator: Distribution

$\text{StaticDist}(\text{type}(\text{extStop}, i), \text{status}) = \{\text{floor}_i\}$

$\text{StaticDist}(\text{type}(\text{inStop}, i), \text{status}) = \{\text{elevator}\}$

$\text{StaticDist}(\text{type}(\text{cf}, i)) = \{\text{elevator}\}$

$\text{StaticDist}(\text{extRequest}) = \{\text{floor}_i \mid 0 \leq i \leq \text{MaxFloor}\}$

for each rule other than extRequest:

$\text{StaticDist}(\text{rule}) = \{\text{elevator}\}$

# Conclusion

- A specification suite for design of distributed real-time systems
- Support for separation of concerns
- Formal semantics for mechanized verification using current verification methods and tools

# Ongoing Work and Future Goals

- Mechanized proof: A prototype of proof mechanization under PVS
- From aspects toward implementation: A middleware implementation with the support for separation of concerns