Gamma Statechart Composition Framework

András Vörös, Bence Graics, Vince Molnár

Budapest University of Technology and Economics,
Fault Tolerant Systems Research Group
Complex and Critical Systems

Railway, Automotive, Aerospace

Industry 4.0, Fog and Edge systems

Correctness is important!
Mission

Expressive & efficient design
- Use \textit{composition} to model complex systems
- Use \textit{interfaces} to define services and communication

Code generation for the whole system
- Create code for components (e.g. with Yakindu)
- Gamma will generate \textit{code for the composition}

Build models of high quality
- Continuous \textit{validation}
- Transformations for \textit{formal verification}
Approach

- Component design
- Implementation
Approach

Component design

Implementation

Crossroad with traffic lights...
Approach

Component design

Implementation

Toggled by a timer...
Approach

Component design

Implementation

3-phase
(green → yellow → red → green)
Approach

Component design

Police may interrupt to trigger blinking yellow

Implementation
Approach

Unsafe state: Both directions GREEN

Component design

Implementation
Approach

Component design

Implementation
Approach

Component design

Implementation
Approach

Component design

Implementation
Approach

Component design

Implementation
Approach

Component design

Implementation
import TrafficLightCtrl
import Controller

specification Crossroad {
    component CrossroadComponent := {
        port control : requires Control
        port police : requires PoliceInterrupt
        port priorityOutput : provides LightCommands
    
    components {
        controller : ControllerDeclaration
        priority : TrafficLightCtrlDeclaration
        secondary : TrafficLightCtrlDeclaration
    }
    bindings {
        control -> controller.Control
        police -> controller.PoliceInterrupt
        priorityOutput -> priority.LightCommands
    }
    channels {
        [controller.PriorityControl]-o-[priority]
        [controller.SecondaryControl]-o-[secondary]
        [controller.PriorityPolice]-o-[priority.PoliceInterrupt]
        [controller.SecondaryPolice]-o-[secondary.PoliceInterrupt]
    }
}
Approach

System design

Component design

Implementation
Approach

Formal language for statecharts

**Validation**: non-determinism, race conditions

**Verification**: transforming to UPPAAL

**Test generation**: Unit tests for the component
Approach

1. **System design**
2. **Component design**
3. **Implementation**
4. **Formal semantics for composition**
5. **Intermediate formal model**
6. **Component verification**
7. **System verification**
8. **Validation of the composition**
Approach

Goal: correct system design

Validation, Formal verification, Testing
Approach

Goal: correct system design

Validation, Formal verification, Testing

System design → Formal semantics for composition → System verification

Intermediate formal model

Component design

Implementation

Interchangeable
Case-study

Model-based Demonstrator for Smart and Safe Systems (MoDeS3)

Distributed safety-critical system
  – Prevent accidents

Model-based development of the safety-logic
  – Communication intensive
  – Real-time
  – Developed with the Gamma framework – proven correctness