Modes in asynchronous systems

J-F. Rolland - M. Filali - J-P. Bodeveix
D. Thomas - D. Chemouil

ASTRIUM - IRIT - CNES

UML-AADL
April 2008
Outline

1. Context
2. Modes in AADL
3. Abstract specification
4. Back to AADL
5. Ongoing work
6. Conclusion
Outline

1. Context
2. Modes in AADL
3. Abstract specification
4. Back to AADL
5. Ongoing work
6. Conclusion
Context of the study

- general study: superpose to AADL precise description standard a formal description.
- provide a formal specification for:
  - simulators.
  - verification tools.
  - code generators.
  - general purpose tools base of model transformations.
- approach: use of TLA.
Modes in real time systems

- Real time systems imply a fixed set of tasks
- Number of real time system have different behaviors → multi-moded systems
- A mode is characterized by :
  - A set of functionalities
  - A set of tasks
  - And a set of active hardware
Example

- Typical example: the aircraft
  - take-off
  - landing
  - cruise

- Another example: Satellite inboard software
  - Launch
  - Operational
  - Safe
  - Low power
Outline

1. Context
2. Modes in AADL
3. Abstract specification
4. Back to AADL
5. Ongoing work
6. Conclusion
Modes in AADL

- Describe different operational states
  - Software and hardware components
  - Connections
  - Properties

- Each component can have different configurations
- Mode transition : a complex behavior
The mode automaton

- **State → Mode**
  - Name
  - One must be initial
- **Transition**
  - Name
  - Associated to an event
  - Deterministic transitions
Mode dependent architecture

- **Subcomponents**
  
  Main\_Gps: `process Gps\_Sender\_Basic
  in modes (Dualmode, Mainmode);`
  
  Backup\_Gps: `process Gps\_Sender\_Basic
  in modes (Dualmode, Backupmode);`

- **Connections**
  
  `data port Main\_Gps\_Position -> Position
  in modes (Dualmode, Mainmode);`
  
  `data port Backup\_Gps\_Position -> Position
  in modes (Backupmode);`

- **Properties**
  
  `bool1 => false in modes (M1);`
  
  `bool2 => true in modes (M2);`
Outline

1. Context
2. Modes in AADL
3. Abstract specification
4. Back to AADL
5. Ongoing work
6. Conclusion
Abstract specification of modes

- Focus on the mode transitions:
  - No scheduling
    - Active
    - Inactive
    - Awaiting Mode
  - No communications
- A simple deterministic mode automaton
Thread types and transition atomicity

- Thread types:
  - Critical threads must terminate in the current mode.
  - Normal threads can be stopped in the current mode.
  - Zombie threads are allowed to survive in the next mode.

- Mode transitions cannot be atomic:
  - `StartModeTransition`
  - `ModeTransition`
  - `EndModeTransition`
Mode transitions

- **Normal behavior**
  - currentEvent = "NoEvent"
  - zombies = ∅

- **Waiting for zombie threads**
  - currentEvent = "NoEvent"
  - zombies ≠ ∅

- **Waiting for critical threads**
  - currentEvent ≠ "NoEvent"
  - zombies = ∅
**Mode transitions**

- **Normal behavior**
  - $currentEvent = "NoEvent"$
  - $\land zombies = \emptyset$

- **Waiting for zombie threads**
  - $currentEvent = "NoEvent"
  - $\land zombies \neq \emptyset$

- **Waiting for critical threads**
  - $currentEvent \neq "NoEvent"
  - $\land zombies = \emptyset$

- **Thread Transition**
- **Start Mode Transition**
- **End Mode Transition**
- **Mode Transition**

**Thread Transition**

**Start Mode Transition**

**End Mode Transition**

**Mode Transition**

**Start Mode Transition**

**Thread Transition**
Correspondance with AADL modes

- In AADL mode may be defined at different hierarchical levels
- Only one level for the abstraction: flat automaton
- Equivalent to the mode automata of the instance model
Scheduling abstraction
Properties correspondence

- Synchronized property:
- Active_thread_handling_protocol property:
- Urgency property:
Properties correspondence

- **Synchronized property:**
  - Periodic threads
  - Mode switch only occurs at the hyperperiod of synchronized threads
  - In AADL V1: a boolean value
  - Evolution: A list of transition name

  → Corresponds to critical threads

- **Active_thread_handling_protocol property:**

- **Urgency property:**
Properties correspondence

- **Synchronized property:**

- **Active_thread_handling_protocol property:**
  - Define the protocol used to process buffers of the thread
  - Allow specific thread to end their execution in the new mode

  → Equivalent to zombie threads

- **Urgency property:**
Properties correspondence

- Synchronized property:
- Active_thread_handling_protocol property:
- Urgency property:
  - Used in the dispatch of aperiodic thread
  - Used to choose a mode transition
  → Corresponds to the priority of the transition
Status of the specification

- Model of AADL execution platform:
  - scheduling (Fixed priorities)
  - shared resources (IPCP)
  - Timed communications through ports and shared resources
  - Modes

- Checked properties
  - Schedulability
  - Size of buffers
  - Integrity of shared data
  - No deadlock
Conclusion

- formal description of AADL modes
  - abstraction of AADL modes
  - concretization of the abstraction

- perspectives:
  - provide early verifications for models based on modes.
  - study some implementations schemes.
thank you for your attention.
Questions?