The Montana Toolset: Formal Analysis of AADL Specifications

Oleg Sokolsky
University of Pennsylvania

ARO URI Review Meeting
May 10-11, 2005
Outline

• Goals of the project
• Overview of AADL
• Overview of the Montana toolset
  - Schedulability analysis
  - CHARON annex and hybrid systems simulation
• Summary
Goals of the project

• Tool support for AADL
  - Important for acceptance of the language
• Application of our technologies
  - Modeling and analysis technologies for real-time and hybrid systems, developed over a number of years
  - Limited use because of unfamiliar formalisms
• Partially supported by Phase I AFOSR STTR
The team

- University of Pennsylvania
  - Oleg Sokolsky
  - Jesung Kim
  - Insup Lee
  - Valentina Sokolskaya
- Fremont Associates
  - Duncan Clarke
Outline

- Goals of the project
- Overview of AADL
- Overview of the Montana toolset
  - Schedulability analysis
  - Charon annex and hybrid systems simulation
- Summary
AADL: An Emerging Standard

• Architecture Analysis and Design Language
  - SAE Standard AS5506
• Features:
  - Component-based modeling
    • Hardware and software components
  - Execution semantics: hybrid automata
  - Extensible via annexes
• Tool support: OSATE
  - Eclipse plug-in developed by SEI
Thread Hybrid Automata

Diagram showing states and transitions such as:
- Dispatch
- Ready
- Executing
- Running
- Error detected
- Complete

Transitions include:
- Abort (Process)
- Unblock on releaseResource
- Return server subprogram
- Preempt
- Resume
- Block on getResource
- Call server subprogram
- Background enter(Mode)
- Background exit(Mode)

States with conditions like:
- $\delta c = 0$
- $c = 0$
- $c = 1$

Assert $t = $ Deadline on all outgoing edges.
Thread States

Inactive
Not member of current mode

Active
Member of current mode

Uninitialized Thread
Initialize

Initialized Thread

Activate
Active In New Mode:

Active In New Mode:
Activate Complete:

Active
Dispatch:

Suspended

Compute

Fault:
Repeated:
Recovered:

Deactivate

Deactivate Complete:

Inactive

Inactive

Inactive

Inactive

Deactivate Complete:

Deactivate

Activate

Inactivate

Inactivate Complete:

Initialized Thread

Terminate:

Finalize

Finalize Complete:

Terminated Thread

Thread State

Thread State with Source Code Execution

Application Source Entrypoints
Application as Plug-in
AADL Model Example
Outline

• Goals of the project
• Overview of AADL
• Overview of the Montana toolset
  - Schedulability analysis
  - Charon annex and hybrid systems simulation
• Summary
ACSR and the VERSA Toolkit

• Analysis of timing properties
  - Schedulability analysis of arbitrary task models
    • Multiple processors and resources, task dependencies, etc.
  - ACSR: Algebra of Communicating Shared Resources

• VERSA:
  - Implements ACSR semantics
  - State-space exploration, deadlock detection,…
  - Diagnostics: failing scenarios
Schedulability analysis with ACSR

System is schedulable if its ACSR model has no deadlocks

- Process structure reflects execution semantics
- Resources represent execution constraints
- Priorities encode scheduling policies
VERSA Plugin for OSATE

OSATE / ECLIPSE

AADL model

scenarios

AADL to ACSR translation

VERSA

diagnostics

VERSA plugin
Translating AADL to ACSR

- Threads are modeled as ACSR processes
  - Based on AADL thread semantic automaton
- Processors and access connections are modeled as resources
- Event and data connections are modeled as communication channels
Example: Cruise Control

- Standard example (from OSATE release)

+ auxiliary processes for bookkeeping
Example: Cruise Control

- Processor and connection bindings determine resources
- Scheduling protocol determines priorities
- Periodic processes have activators

Scheduling_Protocol => EDF
Dispatch_Protocol => periodic
VERSAS Plugin: Current Status

- AADL to ACSR translation is defined for most of AADL:
  - Periodic, aperiodic, and sporadic threads, event and data event connections, server subroutines, mode switches, ...
- Implementation supports a subset of AADL:
  - Periodic threads
  - RMS or EDF schedulers
Outline

• Goals of the project
• Overview of the Montana toolset
  - Schedulability analysis
  - Charon annex and hybrid systems simulation
• Summary
Hybrid Automata: Formalism for Hybrid System Models

- **Continuous dynamics**: Mathematical equation
  - Differential equation
    - $x' = 1 \ (dx/dt = 1)$: constant increase
    - $x' = x \ (dx/dt = x)$: exponential increase ($x = e^t$)
  - Algebraic equation
    - $y = \sin(x)$
  - Invariant
    - $x \geq -10$

- **Discrete control**: Finite State Machine
  - State: dynamics
    - $x' = 1, x' = -1, x' = x, x' = -x, \ldots$
  - Transition: switching of dynamics
    - $x' = 1 \quad (x > 10) \rightarrow x' = -1$
Syntax: Modes and Agents

- **Modes** describe sequential behavior
- **Agents** describe concurrency
Charon toolset: control panel
Quadruped Robot

- **Input**
  - touch sensors

- **Output**
  - desired joint angles

- **Components**
  - Brain: control legs
  - Four legs: control servo motors
    - Instantiated from the same pattern
Four-legged Robot: Behavioral Model

- **Control objective**
  \[- \mathbf{v} = \mathbf{c} \]

- **High-level control laws**

  \[
  \begin{align*}
  \dot{x} &= -v \\
  x &\geq -\text{stride} / 2
  \end{align*}
  \]

  \[
  \begin{align*}
  \dot{y} &= kv \\
  x &\leq \text{stride} / 2
  \end{align*}
  \]

- **Low-level control laws**

  \[
  j_1 = \arctan(x / y) - \arccos\left(\frac{x^2 + y^2 + L_1^2 - L_2^2}{2L_1 \sqrt{x^2 + y^2}}\right)
  \]

  \[
  j_2 = \arccos\left(\frac{x^2 + y^2 + L_1^2 - L_2^2}{2L_1L_2}\right)
  \]
AADL Extension for Hybrid Systems

• Motivation
  - ADLs lack support for formal specification of continuous behavior.
  - Hybrid system specification languages do not concern the target system architecture.

• Goal
  - Develop a comprehensive framework:
    • Architecture: AADL
    • Continuous behavior: HS formalism via AADL annex
  - Integrate existing tools for modeling and analysis tools in a unified environment.
Thread components may have an annex for hybrid systems description written in CHARON.

CHARON annex can reference port names declared in the AADL model to define continuous flow over them.

Sub-modes declared in the annex library can be shared by different thread components.

```plaintext
thread leg_control
  features
    joints: port group LegJointsIn;
    coords: port group PointsPosOut;
end leg_control;

thread implementation leg_control.charon
  annex Charon {**
    mode getup = leg_moveXY(...);
    mode walk = leg_movePhase(...);
    trans from getup.e to walk do {...}
  **}
end leg_control.charon;

annex Charon {**
  mode leg_moveXY(...) {...}
  **}
```
Example: Four-legged Robot

- Control objective
  \[ v = c \]
- High-level control law
  \[ \dot{x} = -v \quad x \geq -\text{stride} / 2 \]
  \[ \dot{y} = -kv \]
  \[ \dot{y} = kv \quad x \leq \text{stride} / 2 \]
  \[ \dot{x} = kv \]
- Low-level control law
  \[ j_1 = \arctan(x/y) - \arccos\left(\frac{x^2 + y^2 + L_1^2 - L_2^2}{2L_1 \sqrt{x^2 + y^2}}\right) \]
  \[ j_2 = \arccos\left(\frac{x^2 + y^2 + L_1^2 - L_2^2}{2L_1 L_2}\right) \]
CHARON Plugin

• **Current status:**
  - Syntax and type checking for the annex
    • Eclipse-style diagnostics in the AADL model
  - **Conversion to a CHARON model**
    • Interface to CHARON tools for simulation, analysis, code generation, ...

• **Future work:**
  - Beyond CHARON
    • A more generic HS annex (HSIF?)
  - Tighter integration with CHARON toolkit
Outline

• Goals of the project
• Overview of the Montana toolset
  – Schedulability analysis
  – Charon annex and hybrid systems simulation
• Summary
Summary

• Technology transfer effort
  - Proof-of-concept implementations
  - Commercialization interest
• AADL offers an opportunity to bring formal modeling into the mainstream
  - Uniform tool framework
  - Formalism details are hidden behind AADL front end
• Much remains to be done