The Montana Toolset: OSATE Plugins for Analysis and Code Generation

Oleg Sokolsky
University of Pennsylvania

AADL Workshop 2005
Paris, France  October 17-18, 2005
Outline

► Goals of the project
► Overview of the Montana toolset
  - Schedulability analysis
  - Charon annex and code generation
► Where do we go from here?
Goals of the project

► Tool support for AADL
  - Important for acceptance of the language

► Modeling and analysis technologies for real-time and hybrid systems
  - Behavior in the architectural context
  - developed over a number of years
  - limited use because of unfamiliar formalisms

► AFOSR STTR
  - Fremont Associates
Outline

► Goals of the project
► Overview of the Montana toolset
  - Schedulability analysis
  - Charon annex and code generation
► Where do we go from here?
Formalism for Real-Time Systems

- ACSR: Algebra of Communicating Shared Resources
  - Resource-sensitive specification
  - Timing constraints
- Analysis of timing properties
  - Schedulability analysis of arbitrary task models
    - Multiple processors and resources, task dependencies, etc.
- Analysis of safety properties
VERSAs

- Implements ACSR semantics
- State-space exploration, deadlock detection,…
- Diagnostics: failing scenarios
ACSR Modeling Principles

- Threads are modeled as state machines
  - Operators for timeout, preemption
  - Timed and instantaneous steps
  - Timed steps access resources

- Concurrency model
  - Exclusive resource access
  - Hand-shake synchronization
Timing and resource analysis

Resource conflict as deadlock

- Allows us to encode task models, scheduling policies, inter-task dependencies, etc.
Translating AADL to ACSR

- Threads are modeled as ACSR processes
  - Based on 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 => resources
- Scheduling protocol determines priorities
- Periodic processes have activators

Scheduling Protocol => EDF
Dispatch Protocol => periodic
VERSA Plugin for OSATE
VERSA 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 code generation
►► Where do we go from here?
Hybrid Automata

- **Continuous dynamics: restricted DAE**
  - Differential equation
    - $x' = -x$
  - Algebraic equation
    - $y = \sin(x)$
  - Invariant
    - $x \geq -10$

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

- Modes describe sequential behavior
- Agents describe concurrency
Charon toolset

- Modeling/type checking
- Reachability analysis
- Simulation
- Code generation
Case Study: Architectural Model

►► Input
  ▪ touch sensors

►► Output
  ▪ desired angles of each joint

►► Components
  ▪ Brain: control four legs
  ▪ Four legs: control servo motors
  ► Instantiated from the same pattern
Case Study: Behavioral Model

- **Control objective**
  - \( v = c \)

- **High-level control laws**
  - \( \dot{x} = -v \)
    - \( x \geq -\text{stride} / 2 \)
  - \( \dot{y} = kv \)

- **Low-level control laws**
  - \( \dot{y} = -kv \)
    - \( x \leq \text{stride} / 2 \)
  - \( \dot{x} = kv \)
  - \( j_1 = \arctan(x / y) - \arccos\left(\frac{x^2 + y^2 + L_1^2 - L_2^2}{2L_1 \sqrt{x^2 + y^2}}\right) \)
  - \( j_2 = \arccos\left(\frac{x^2 + y^2 + L_1^2 - L_2^2}{2L_1 L_2}\right) \)
AADL Annex for Hybrid Systems

Goal

- Develop a comprehensive framework for formal specification of hybrid systems in both architectural and behavioral aspects
  - Architectural specification: AADL
  - Continuous behavior specification: HS formalism via AADL annex
- Integrate existing tools for model composition, analysis, simulation, and code generation in a unified environment
AADL HS Annex: CHARON

- 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 PointsPos Out;
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
  \[ \dot{x} = -v \]
  \[ x \geq -\text{stride} / 2 \]
  \[ \dot{y} = -kv \]
  \[ x \leq \text{stride} / 2 \]

- Low-level control law
  \[ j_1 = \arctan(x/y) - \arccos\left(\frac{x^2 + y^2 + L_1^2 - L_2^2}{2L_1 \sqrt{x^2 + y^2}}\right) \]
  \[ j_2 = \arccos\left(\frac{x^2 + y^2 + L_1^2 - L_2^2}{2L_1 L_2}\right) \]

Process

Thread Group

Thread

Annex Charon

AADL Workshop 2005
CHARON Plugin: Current Status

► Syntax and type check for the CHARON annex
   - Eclipse-style diagnostics in the AADL model
► Conversion to a CHARON model for simulation, analysis, and code generation
   - Analysis tools have to be started manually
Outline

► Goals of the project
► Overview of the Montana toolset
  - Schedulability analysis
  - Charon annex and code generation
► Where do we go from here?
CHARON Plugin: Future Work

► Refined AADL annex definition
  ▪ Based on COTRE state machine annex?
► Tighter integration with analysis tools
  ▪ Automatic invocation
  ▪ AADL-level diagnostics
► Code generation based on target platform specification in the AADL model
  ▪ Period, scheduler, processor type, memory binding, etc.
VERSA Plugin: Future Work

► Complete the translation
► Interface with a better state space exploration engine
► Better user interface
  - Visual representation of failing scenarios
► Interactive simulation
  - Replay of scenarios