The AADL Behavioral annex : an example of use

Frank Singhoff, LISyC/UBO

UML-AADL workshop'2010

Verification with the real-time scheduling theory (1/5)

- **Real-time systems:**
  1. Functions of real-time systems may have timing constraints.
  2. Example of timing constraints : deadlines
  3. **Functions must meet timing constraints: how to check them at design time ?**

- **Timing analysis with real-time scheduling theory (also called “Rate Monotonic Analysis”):**
  1. **Modeling functions:** simplified models of task = processor demand + deadline (e.g. periodic task model).
  2. **Use of well known task schedulers** (e.g. Rate Monotonic).
  3. **Verification:** either by simulation or by feasibility tests.

Verification with the real-time scheduling theory (2/5)

- **Example of feasibility test to perform verification** : worst case response time of a set of periodic tasks, sharing resources, scheduled by Rate Monotonic, (Joseph & Pandia 1986):

$$R_i = C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{P_j} \right\rceil \cdot C_j \qquad \leq deadline_i$$

- **Verification method:**
  - Compute Ri, the worst case response time of task i as a sum of:
    1. Ci, worst case execution time of task i (or task capacity).
    2. Bi, worst case waiting time of task i for shared resources.
    3. Waiting time for the processor due to scheduling of the other tasks.
  - Compare Ri with the task i deadline.

- **Weakness of this approach** : low accuracy of Ci and Bi may lead to compute a too pessimistic Ri.

Verification with the real-time scheduling theory (3/5)

```
PROCESS IMPLEMENTATION a_process.I
SUBCOMPONENTS
 thread1 : THREAD a_thread.I;
 thread2 : THREAD a_thread.I;
 resource1 : DATA a_data.I {
Concurrency_Control_Protocol=>PCP};
END a_process.I;
```

```
THREAD IMPLEMENTATION a_thread.I
PROPERTIES
 Dispatch_Protocol => Periodic;
 Deadline => 10 ms;
 Period => 10 ms;
 Compute_Execution_Time=> 1 ms ..7 ms;
 Bound_On_Data_Blocking_Time => 7 ms;
END a_thread.I;
```

Ci → Compute_Execution_Time

Bi → Bound_On_Data_Blocking_Time

- AADL Model without Behavioral annex specification:
  - Verification is possible.
  - But : no thread behavior specification => we assume that threads use the shared resource during all their execution time, which leads to a pessimistic Bi.
  - May imply a pessimistic worst case response time (Ri)
  - Behavioral annex provides a mean to increase accuracy of the model.

Verification with the real-time scheduling theory (4/5)

```
PROCESS IMPLEMENTATION a_process.I
SUBCOMPONENTS
 thread1 : THREAD a_thread;
 thread2 : THREAD a_thread;
 resource1 : DATA a_data {
Concurrency_Control_Protocol=>PCP};
END a_process.I;
```

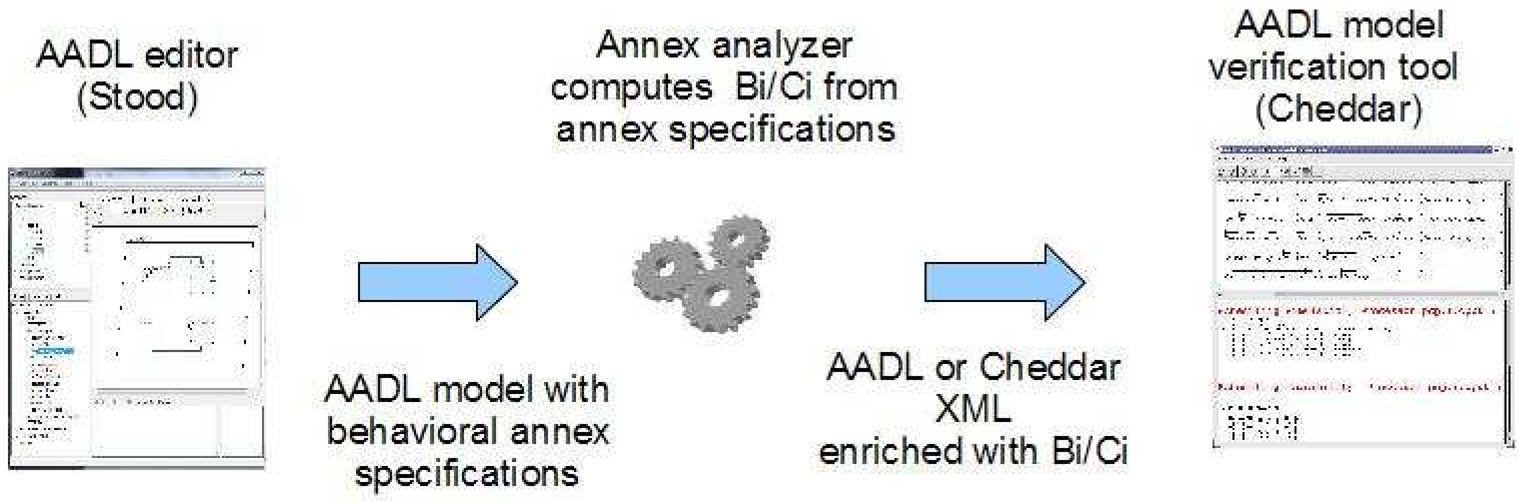
```
THREAD IMPLEMENTATION a_thread.I
PROPERTIES
 Deadline => 10 ms;
 Compute_Execution_Time => 1 ms .. 7 ms;
 Bound_On_Data_Blocking_Time => 7 ms;
 ANNEX Behavior_Specification {**
  states s : initial complete final state;
  transitions t : s -[on dispatch]-> s
   {computation(4 ms); resource1!<;
    computation(3 ms); resource1!> };
**};
END a_thread.I;
```

Ci

Bi

- AADL model with Behavioral annex specification:
  - Specification of the thread behavior: specify when threads use the shared resource (allocate and release time of the resource).
  - Less pessimistic Bi approximation (Bi=3 and not 7), and then more accurate worst case response time (Ri).

Verification with the real-time scheduling theory (5/5)



- AADL and its annexes are standardized, and make possible tool interoperability.
- Example of an AADL Tool chain (Ellidiss & Univ. Of Brest):
  1. STOOD or ADELE AADL editor (both core AADL & annex specifications)
  2. Static analysis of Behavioral annex specifications. Produce enriched Cheddar XML with shared resource accesses (Bi).
  3. Analysis with feasibility tests by Cheddar.

Conclusion of the panel

- **AADL allows architecture designers to:**
  - Model, verify and generate embedded real-time systems architectures.
  - Assume an iterative development process, and allows model refinements.

- **AADL core language and annexes:**
  - Core language defines system architectures.
  - Annexes give different view points of system architectures (e.g. verification, code generation, …).
  - Examples of annex: ARINC 653, behavioral, data modeling, error, …

- **AADL has been used as a pivot language in numerous research projects:** IST-ASSERT, TOPCASED, SPICES, ANR Flex-eWare, MOSIC, SAVI AVSI, …

- **Many AADL interoperable tools:** OSATE, Furness, TINA, CPN/AMI, Lotos, BIP, GreatSPN, STOOD, ADELE, Ocarina, Cheddar, …

- **Further readings:** http://www.aadl.info