The AADL behavior annex - experiments and roadmap

R. B. França[1] J-P. Bodeveix[1] M. Filali[1] J-F. Rolland[1]
D. Chemouil[2] D. Thomas[3]

[1] Institut de Recherche en Informatique de Toulouse
Université Paul Sabatier
Équipe ACADIE

[2] Centre National d'Études Spatiales

[3] EADS Astrium Satellites

UML&AADL 2007

Outline

Outline

The ArchiDyn study

Goals:

- Describe a complete flight software in AADL
- Integration of AADL in an existing process
- Identification of architectural patterns
- Evaluation of the language (AADL + behavioral annex)

Conclusion of this study

Concerning the 2006 version of the behavioral annex:

- the annex is too simple, we needed:
  - composite states
  - concurrent states
  - Multiples synchronizations
  - a more precise definition of communication timings

The behavioral annex

- A simple state-transition system
- Describe the behavior of thread and subprograms
- Actions can read and/or write ports
- Start in an initial state and end in a complete state
- Complete state is the starting state for the next dispatch
- Perform (eventually remote) procedure calls

Example

```
thread AVB_HDLR
features
  timer: in event port;
  int_it: in event port;
  eof_it: in event port;
  aocs_ack: out event port;
  dor_ack: out event port;
properties
  Dispatch_Protocol => Sporadic;
  Period => 10ms;
end AVB_HDLR;
```

```
thread implementation AVB_HDLR.i
annex behavior_specification {**
states
  s0: initial complete state;
  s1, s2, s3: complete state;
transitions
  s0 -[ timer? ]-> s1;
  s1 -[ int_it? ]-> s2 {aocs_ack!;};
  s2 -[ int_it? ]-> s3 {dor_ack!;};
  s3 -[ eof_it? ]-> s0;
**};
end AVB_HDLR.i;
```

Outline

Composites states

- Refinement of AADL modes
- Hierarchical automata

```
system implementation AOCS_FUNCTION.N0
 modes
  IDLE: initial mode; ASH_MODE: mode;
  NM_MODE: mode; OCM_MODE: mode;
 annex behavior_specification {**
  mode transitions
   IDLE -[ TC.ASH_REQ? ]-> ASH_MODE;
   NM_MODE -[ TC.OCM_REQ? ]-> OCM_MODE;
   ASH_MODE, OCM_MODE -[ TC.NM_REQ? ]->
NM_MODE;
```

```
   composite mode ASH_MODE
    states
     STAB: initial state;
     STRAP, SLO: state;
    transitions
     STAB -[ on Bdot_Control_Law_Converged ]->
STRAP;
     STRAP -[ on Sun_Presence ]-> SLO;
     SLO -[ on Sun_Presence = false ]->
STRAP;
   end ASH_MODE;
   ...
  **};
end AOCS_FUNCTION.N0;
```

Concurrent states

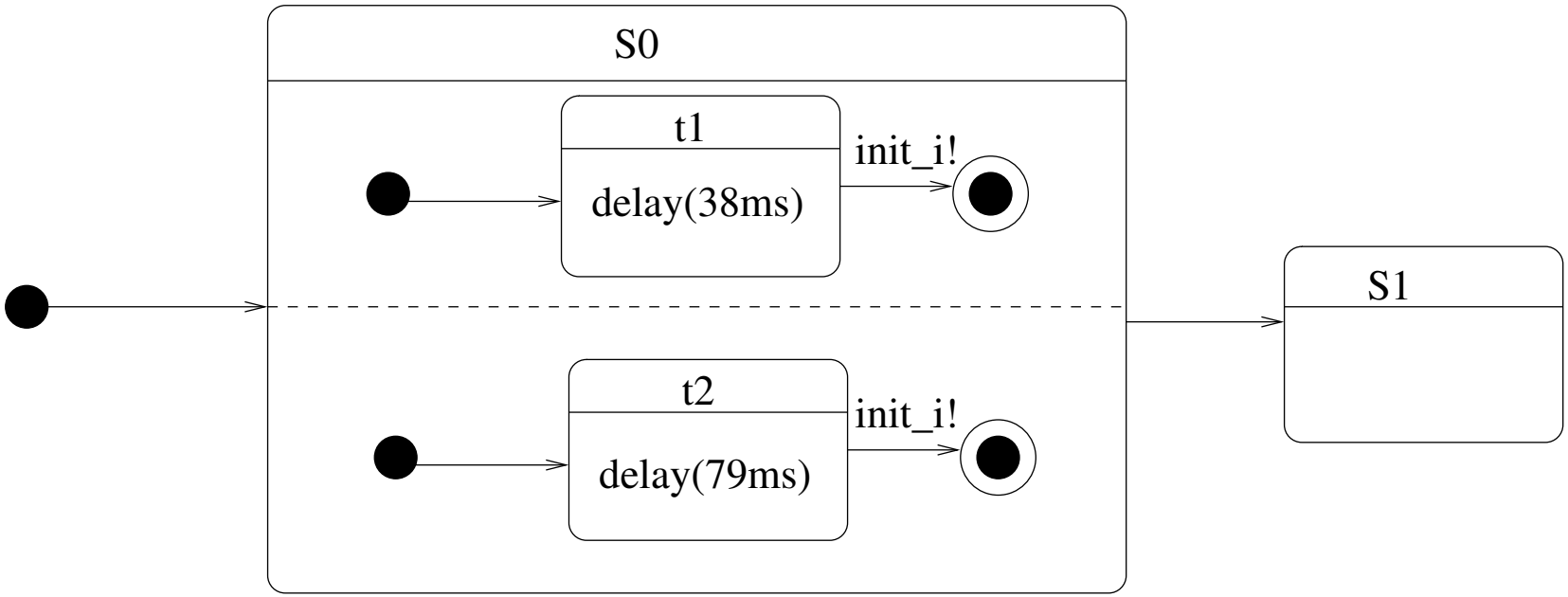
- Introduce logical parallelism
- No run-time threads
- Reduction of the number of states

Concurrent state - example

```
thread avb_bus
features
  int_it: out event port;
  eof_it: out event port;
properties
  Dispatch_Protocol => Periodic;
  Period => 125ms;
end avb_bus;

thread implementation avb_bus.i
annex behavior_specification {**
states
  s0: initial concurrent state;
  s1: complete join state;
transitions
  s0 -[ ]-> s1;
concurrent state s0
composite state st1
  states
    t1: initial exit state;
  transitions
   t1i -[ ]-> t1i{delay(38ms);int_it!;
end t1;
composite state st2
  states
    t2: initial exit state;
  transitions
   t2i -[ ]-> t2i{delay(79ms);int_it!;
end t2;
 ...
end s0;
**};
end avb_bus.i;
```

Concurrent state - example



Multiple Synchronizations

- Dispatch on reception of multiple signals
- Dispatch on timer
- Supported by several OS

```
thread implementation PL_CYC.i
annex behavior_specification {**
states
  RQ: initial state;
  wait_PLB, ACC, DA: complete state;
transitions
  RQ -[ start? ]-> wait_PLB { computation(2ms); };
  wait_PLB -[ PLB_ACK? ]-> ACC { computation(4ms); };
  ACC -[ PF_END? & AOCS_END? ]-> DA { ... };
**};
end PL_CYC.i;
```

Requested extension to AADL

- Modification of the dispatch mechanism
  - Dispatch condition expressed as a boolean function on ports
  - Bounded wait (support for timeout)
- More precise support for communication timing:
  - Date of emission or reception

Example

```
thread implementation PL_CYC.i
annex behavior_specification {**
states
  RQ: initial complete state;
  s1, s2 : state;
transitions
  RQ -[ start? ]-> s1 { computation(2ms); sync1 ! };
  s1 -[ ]-> s2 { computation(4ms); sync2 !};
  s2 -[]-> RQ { sync3 ! };
**};
end PL_CYC.i;
```

Outline

Conclusion

- Extension of the behavioral annex (concurrent and hierarchic states)
- Proposition of extension for AADL
- Integration of a static analyzer in Topcased
- Perspectives
  - How to implement the “delay” with AADL constructions
  - Define simplification for hierarchical and concurrent states
  - Formalize the link between AADL and the behavioral annex

Ongoing work around AADL

Formal semantics of the AADL execution model in TLA+

- TLA+ : a formal language based on set theory and temporal logic
- We consider a subset of AADL (Threads, Data, Ports, Shared variables)
- Model checking using TLC

Ongoing work around AADL

- A mapping to Real-Time Java
  - Implementation of the execution model in RTSJ
  - Based on the semantics defined in TLA
- AADL to Fiacre
  - Intermediate language for model checking purpose
  - Synchronous communication
  - Timed transition systems semantics
  - Hierarchical components