AADL Subset Annex

Frank Singhoff et al.

AADL WG meeting

January 2012, Toulouse

Foreword

- **This presentation:** a trial summary on the discussions on this annex since last November.

- **Summary of the discussions on the wiki:**
  http://wiki.sei.cmu.edu/sae-aadl-subcommittee/index.php/Subset_Annex

- **List of topics/questions:**
  1) What are the objectives of this annex?
  2) What is the name of the annex? **(closed)**
  3) What is the scope of this annex?
  4) Is there a potential relationship with AADL constraints languages such as REAL?
  5) How a given profile can/should be expressed?
  6) Do we need pre-defined restrictions/subsets which would be commonly used? **(closed)**
  7) What kind of pre-defined or examples of possible subsets?
  8) What we do not want to see with this annex?
  9) How can we define ordering of subsets?
  10) On which entities do we must specify a subset? on the AADL elements or on instance elements?

Talk overview

1. **Rationale for this annex**
2. **Proposals for this annex**
3. **Examples of subsets**
4. **Roadmap: what's next?**

Rationale for this annex (1/3)

- **Statement addressed by this annex:**
  1. **AADL is a rich language.**
     - Example 1: many supported protocols (e.g. thread communication and synchronization features).
     - Example 2 : different ways to express similar architectures (e.g. producers and consumers synchronization).
     - Example 3: many possible model automatic verifications or code generations.
  2. **Each verification/code generation may have specific requirements**:
     1. **On elements a model must include** : priority property for schedulability analysis, Source_Text for source code generators
     2. **On elements a model must not include:** mode for source code generators, aperiodic threads for schedulability analysis,…
  3. **Tools that are devoted for a given analysis usually support a subset of AADL** and can be unable to handle some other parts of AADL : knowledge embedded into the tools.

Rationale for this annex (2/3)

- **Leads to some tool interoperability failures:**
  - A tool may not work because an AADL model does not contain some mandatory elements, and users don't know which ones.
  - A tool may not work because an AADL model contains elements that can not be handled by the tool, and users don't know which ones.
  - AADL toolchains are difficult to design : a tool may require a subset that is incompatible with the subset of an other tool (analysis tool + source code generator), designers don't know which ones.
- **And probably a limited use of some AADL tools:**
  - Difficult to understand the scope of a tool for users as they are not expert.

Rationale for this annex (3/3)

- **What do we need:**
  - Some means to clearly state the scope of a tool, of an analysis or of a source code generator.
- **What can we expect with the use of this annex:**
  - Increase tool interoperability.
  - Increase confidence of users when they (try to) use tools:
    - Allows users to understand why a tool cannot work with their AADL models.
    - A kind of documentation that define the usability of a tool.
  - AADL should become more accessible to tool vendors and potential users.
  - Certification toolkits for subset: allow tool designers to check compliancy with their products.
  - Allow users to define constraints that are specific to their systems or overall development process.
- **We don’t want to see:**
  - Must not lead to loss of information in AADL models.
  - This annex should not be used as an extension mechanism.
  - The approach should be simple in use, like a pragma restriction. .
  - This annex should not be yet another constraint language.

Talk overview

1. Rationale for this annex
2. Proposals for this annex
3. Examples of subsets
4. Roadmap: what's next?

Proposals for this annex (1/4)

- **Subset is not a new idea:** Ravenscar profile of Ada 2005 allows Ada programmers to statically check compliancy of their programs with real time scheduling analysis methods.

- **What we should or must define in this annex:**
  1. A language to specify a subset?
  2. Do we need to define a pragma concept for AADL?
  3. Some pre-defined/standard subsets? To model typical execution environments or typical verifications/source code generators.
  4. Do we need to define certification tool kits?

Proposals for this annex (2/4)

- **Basic concept of this annex:**

1. **A subset is a set of constraints.**

2. **A constraint models something that:**
   - is either required (e.g. a requirement to apply an analysis).
   - or forbidden (e.g. a restriction to apply an analysis).

3. **Constraints can be:**
   - formally expressed .
   - or more simply, expressed in natural languages.

4. **Scope of a subset: a subset can be used to check compliancy:**
   - of a whole AADL model.
   - or only a part of it (a system may have a small critical part).

Proposals for this annex (3/4)

- **Constraints can be expressed on (scope):**
  - **AADL entities (instance model):** specify which categories/components/connections/mode/flow/… are forbidden/required.
  - **On AADL syntax (declarative model?):** which parts of the AADL BNF is forbidden/required.
  - **On AADL properties (user-defined or standard properties sets such as AADL_Project):** specify which properties are required, or which property values are forbidden/required.
  - **On AADL annexes (standard or user-defined):** which annexes are required/forbidden.

Proposals for this annex (4/4)

- **Examples of constraints proposed during the discussions:**
  - Restrictions/requirements on thread connections/feature.
  - Restrictions/requirements on allowed categories of component, on mode, flow, abstract, inheritance
  - Architectural restrictions (e.g. number of component): one process per system, one system, …
  - Restrictions/requirements on the properties or on the properties set that must be present or is forbidden in an AADL model.
  - Restrictions/requirements on property values such as :
    - List of values that are allowed/forbidden,
    - Constraints mixing the values of different properties ; e.g. values for property A that are allowed/forbidden when property B has a given value
  - etc

Talk overview

1. Rationale for this annex
2. Proposals for this annex
3. Examples of subsets
4. Roadmap: what's next?

Examples of subsets

- **Scope of standard/pre-defined AADL subsets:**
  - Should model either particular model of computation (DDS, synchronous, time triggered, ARINC)
  - Or particular verification/test/certification/analysis tool.
- **List of potential standards/pre-defined AADL subsets:**
  - Ravenscar?
  - HOOD-HRT?
  - DO178C?
  - MILS?
  - What are the relationships with the ARINC653 annex?
- **Ideas? Volunteers to work on others?**
  - **??**
- **Current trials of AADL subsets on the wiki:**
  - AADL-light subset from Brian.
  - HOOD and HOOD-HRT subsets from Ellidiss (Stood tools).
  - Schedulability analysis subsets from Ellidiss/U. Brest (Cheddar/AADLInspector tools): “Ravenscar like”, synchronous data flow, ...

AADL light subset for Brian's BLESS proof tool

- **Constraints on AADL syntax (both on declarative model and instance model?):**
  - no mode
  - no generic
  - no abstract component
  - no flow
  - no inheritance (`extends` just to parse)
  - no refinement
  - no subprogram call sequence
  - no subcomponent access feature
  - no parameter connection
  - no in-binding
  - no contained property association

AADL subsets for HOOD and HRT-HOOD

- **Modeling restrictions for an HOOD subset:**
  - One single System
  - No Bus, Virtual Bus, Memory, Device
  - No Extend, except for Data Components
  - No Provides Data Access, Data Port
- **Additional restrictions for HRT-HOOD subset (kind of Ravenscar):**
  - Only Periodic and Sporadic Dispatch Protocols
- **Methodological constraints (how to organize models):**
  - Separate models for each SW application or library
  - Separate models for deployment (Processors and Virtual Processors)
  - Separate models for genericity (Prototyped Components)
  - More constraining visibility rules

Notes:

*- **"Stood for HOOD"** supports this subset*

*- **"Stood for AADL"** supports multiple Systems, Buses, Memories, Devices and Data Ports*

AADL schedulability subsets for AADLInspector/Cheddar (1/4)

- **Feasibility test, a schedulability analysis method:** example of the worst case response time of tasks feasibility tests (Joseph & Pandia 1986):

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

- **Applicability assumptions:**
  - Periodic tasks, scheduled by fixed priority schedulers.
  - Deadlines are equal to periods.
  - All tasks start at the same time (same release time).
  - **...**

- **Schedulability verification with feasibility tests is difficult to apply with AADL :** we must be sure that the AADL model is compliant with the applicability assumptions.

AADL schedulability subsets for AADLInspector/Cheddar (2/4)

- **Define a set of of AADL subsets:**
  - A subset models a typical thread communication/synchronization mechanism.
  - Mostly constraints on thread communication/synchronization … but not only.
- **For each subset, define feasibility tests that can be applied according to their applicability assumptions.**
- **Verification of a real-time system architecture model by an designer:**
  1. He checks compliancy of his model to one of the subset … which then gives him which feasibility tests he can apply.
  2. Perform verifications with a tool implementing these feasibility tests (e.g. AADLInspector, Cheddar).

AADL schedulability subsets for AADLInspector/Cheddar (3/4)

- **List of investigated subsets:** Synchronous data-flow, “Ravenscar like”, Blackboard, Queued Buffer, Unplugged.

- **Two sets of restrictions for each subset:**
  1. Restrictions commons to all subsets for Cheddar.
  2. Restrictions that are specific for each subset for Cheddar.

- **Subsets are expressed** with both EXPRESS and REAL

- **Subset compliancy tool** can be generated from EXPRESS models with Platypus (Vincent Gaudel’s Phd).

- **Types of restriction we need for these subsets:**
  - Restrictions on thread connections/features allowed in an AADL model
  - Restrictions on categories of component allowed in an AADL model
  - Restrictions on a property that must be present or not in an AADL model
  - Restrictions on property values (allowed/forbidden values, comparison of the values of different properties, conditional restriction depending on the value of a property, …)

AADL schedulability subsets for AADLInspector/Cheddar (4/4)

- **Example of the constraints for "Ravenscar like" from Vincent:**

1. Restrictions on the execution environment:

- R2: for all processors, property Scheduling_Protocol must be only either POSIX_Fixed_Priority_Scheduling, Rate_Monotonic, Earliest_Deadline_First or Deadline_Monotonic
- R3: for all processors, the property Preemptive_Scheduler must be defined
- R4: for all processors, property Scheduler_Quantum must be unspecified or set to 0.

2. Restrictions that are specific to Ravenscar subset:

- R8: for all threads, Dispatch_Protocol must be only either Periodic or Sporadic.
- R9.1 : there is at least one data component.
- R9.2 : only data access connections are allowed.
- R10: each data component must be connected to at least two threads.
- R11: property Concurrency_Control_Protocol must be only either Priority_Inheritance_Protocol, Priority_Ceiling_Protocol or Immediate_Priority_Ceiling_Protocol
- R12: if property Concurrency_Control_Protocol has the values Priority_Ceiling_Protocol or Immediate_Priority_Ceiling_Protocol, data's Ceiling priority must be higher or equal to the maximum value of property Priority of all threads connected to the data component
- R13: No deadlock: if the Priority Inheritance Protocol is used, then two threads can not share more than one data component.

=> Further information on the wiki (including a trial of REAL model from Jérôme).

Talk overview

1. Rationale for this annex
2. Proposal for this annex
3. Examples of subsets
4. Roadmap: what's next?

Roadmap: what's next?

- **Current opened topics:**

  1) What are the objectives of this annex?
  8) What we do not want to see with this annex?

  3) What is the scope of this annex?
  4) Is there a potential relationship with AADL constraints languages such as REAL?
  5) How a given profile can/should be expressed?
  7) What kind of pre-defined or examples of possible subsets?
  9) How can we define ordering of subsets?
  10) On which entities do we must specify a subset? on the AADL elements or on instance elements?

- **Can we close 1 and 8?**

- **How to investigate 3, 4, 5, 7 and 10:** a possible way is to work on few subset examples. *Who is volunteer to produce something?*