What is an AADL Subset?

V. Gaudel†, P. Dissaux*, A. Plantec†, F. Singhoff†, J. Hugues**, J. Legrand*

†University of Brest/UBO, Lab-Sticc, France
*Ellidiss Technologies, France
**Institut Supérieur de l’Aéronautique et de l’Espace/ISAE, France

February, 2013
Introduction (1/2)

Rationale for the Subset annex (February 2012 Meeting)

1. AADL is a rich Language.
2. Each verification/code generation may have specific requirements.
3. Tools that are devoted for a given analysis usually support a subset of AADL.

Addressed problems

1. Use of AADL may lead to some tool interoperability failures.
2. Probably causes a limited use of some AADL tools.

Objectives of the Subset annex (February 2012 Meeting)

1. Increase tool interoperability.
2. Increase confidence of users when they (try to) use tools.
3. Certification toolkits for subset: allow tool designers to check compliance with their products.
4. Allow users to define constraints that are specific to their systems or overall development process.
Problems we try to answer (February 2012 Meeting)

1. What is a subset?
2. How to express it?

Proposition

1. Investigate 3 examples of Subsets.
2. Proposition of a superset from whom all subsets could be defined.
3. Investigate the different kinds of constraints of those subsets.
4. Proposition of an uniform way to describe constraints.
Outline

1. Subset Examples
2. Superset: an AADL Meta-Model
3. Examples of cardinality constraints
4. Mapping towards REAL and Prolog
5. Conclusion
Subset Example 1: Marzhin V1

- **Require**: There is only one Processor component.
- **Require**: The property Actual_Processor_Binding must be specified.
- **Require**: For all processors, property Scheduling_Protocol must have the following values: `POSIX_Fixed_Priority_Scheduling_Protocol`, `Rate_Monotonic_Protocol` or `Deadline_Monotonic_Protocol`.
- **Require**: The property Dispatch_Protocol must have one of the following values: Periodic, Aperiodic, [...], Background.
- **Require**: Properties must be one of the following: `Dispatch_Protocol`, `Period`, `Deadline`, `Priority`, `Compute_Execution_Time`
- ...

- **Authorized**: See AADL-Light Cheat Sheet.
- **Forbid**: There is no abstract component.
- **Forbid**: There is no subprogram call sequence.
- **Forbid**: There is no in-binding.
- **Forbid**: There is no contained property association.
- ...

- **Authorized:** See AADL-Light Cheat Sheet.

- **Forbid:** There is no abstract component.
- **Forbid:** There is no subprogram call sequence.
- **Forbid:** There is no in-binding.
- **Forbid:** There is no contained property association.
- ...
Subset Example 3: Cheddar Subsets

- **Require**: For all threads: Dispatch Protocol must be set to Periodic.
- **Require**: All connections must be Data Port connections.
- **Forbid**: There is no data component.
- **Forbid**: All features must be Data Port.
- **Forbid**: For all Data port, property Timing must have the following values only: sampled, immediate or delayed.

- **Require**: If property `Concurrency_Control_Protocol` has the values `Priority_Ceiling_Protocol` or `Immediate_Priority_Ceiling_Protocol`, Data Ceiling priority must be higher or equal to the maximum value of property Priority of all threads connected to the data component.
- ...

What is an AADL Subset?
Different ways to define subsets:

- Subset: AADL-Light
  - AADL Declarative Model
  - Specifies Authorized/Forbidden parts

- Subsets: Cheddar, Marhzin V1
  - AADL instance model
  - Specifies Restrictions parts.

But of course, they have a common point: AADL Meta-model.
Outline

1. Subset Examples
2. Superset: an AADL Meta-Model
3. Examples of cardinality constraints
4. Mapping towards REAL and Prolog
5. Conclusion
Superset: a meta-model common to all subsets

1. Based on Appendix C for element identifiers
2. And literal descriptions of entities’ attributes
3. Use of multiple inheritance
4. **What is in the superset?**
   - Model of the declarative part of AADL.
   - Instance model can be deduced from this model.
   - Property sets and annexes are considered as parts of the superset.
Meta Model Specification with Platypus

Use of Platypus for prototyping

- Meta-environment based on ISO STEP technology.
- Enables to design, to verify and to validate meta-models written with EXPRESS.
- Enables to implement code generators for EXPRESS meta-model.
- Meta-model elaboration within Platypus
  - EXPRESS is readable
  - The model is checked and evaluated during design
  - Enables multiple inheritance
  - Platypus is already used for code generation with Cheddar
  - We can specify metrics
  - Definition of rules to implement consistency rules
  - Possibility of using this kind of rule for subset definition
What could be a subset?

New Subset Model Proposal

1. Superset is an EXPRESS Meta-model
2. A subset constraint is modeled by an EXPRESS RULES on the superset
3. Then, each subset is declared as a set of EXPRESS RULES on the superset
4. What we assume:
   - A constraint is a cardinality verification
   - Or a composition of cardinality verifications
Outline

1. Subset Examples
2. Superset: an AADL Meta-Model
3. Examples of cardinality constraints
4. Mapping towards REAL and Prolog
5. Conclusion
Examples of cardinality constraints

From literal constraints to cardinality constraints

Summary of encountered constraints:

- There is no \textit{model element}.
- There must be \textit{model element}.
- The value/content of \textit{model element} must be [...]
- \textit{Some property} must be specified.
- For all \textit{model element}, \textit{constraint upon dependent model element}.
Examples of cardinality constraints

From literal constraints to cardinality constraints:

There must be [model element]

- **Forbid**: There is no data component.

```plaintext
RULE No_Data_Instance FOR ( Data_Instance );
WHERE
   R-TT-C2 : SIZEOF ( Data_Instance ) = 0;
END_RULE;
```
Examples of cardinality constraints

From literal constraints to cardinality constraints

There is no [model element]

- **Require:** There is only one Processor component.

```
RULE Only_One_Processor FOR ( Processor_Instance );
WHERE
  RM1 : SIZEOF ( Processor_Instance ) = 1;
END_RULE;
```
Examples of cardinality constraints

From literal constraints to cardinality constraints:

For all \textit{model element}, \textit{constraint upon dependent model element}

- \textbf{Require}: For all threads, the property dispatch protocol must be periodic.

```
RULE Dispatch_Protocol_Must_Be_Periodic FOR ( Thread_Classifier );
WHERE
  RM4_Part3 : SIZEOF ( QUERY ( t <*> Thread_Classifier |
    ( SIZEOF ( QUERY ( p <*> t.properties |
      ( ( p.Property_Name = 'Dispatch_Protocol' ) AND
        ( p.VALUE = 'Periodic' ) ) ) ) ) ) = 0 ) ) = 0;
END RULE;
```
From literal constraints to cardinality constraints:

For all \([\textit{model element}]\), the value/content of \([\textit{model element}]\) must be [...] 

- **Require**: For all processors, property \(\textit{Scheduling\_Protocol}\) must have the following values: \(\textit{POSIX\_Fixed\_Priority\_Scheduling\_Protocol}\), \(\textit{Rate\_Monotonic\_Protocol}\) or \(\textit{Deadline\_Monotonic\_Protocol}\).

```
RULE Scheduling_Proto콜.Must.Be.Posix_FP FOR ( Component.Classifier );
WHERE
  RM3_Part1 : SIZEOF ( QUERY ( c <= Component.Classifier | ( c.category = processor ) AND
    SIZEOF ( QUERY ( p <= c.properties | ( ( p.Property_Name = 'Scheduling.Protocol' ) AND
      ( p.VALUE = 'Posix.Fixed.Priority.Scheduling.Protocol' ) ) ) ) ) ) = 0 ) ) ) ) = 0;
END_RULE;

[...]
( p.VALUE = 'Rate.Monotonic.Scheduling.Protocol' ) ) ) ) = 0 ) ) ) ) = 0;

[...]
( p.VALUE = 'Deadline.Monotonic.Scheduling.Protocol' ) ) ) ) = 0 ) ) ) ) = 0;
END_RULE;
```
From literal constraints to cardinality constraints:

And so on ...

- **Require**: If property `Concurrency_Control_Protocol` has the value `Priority_Ceiling_Protocol`, data Ceiling priority must be higher or equal to the maximum value of property Priority of all threads connected to the data component.

- **Require**: For each Data with `Concurrency_Control_Protocol = Priority_Ceiling_Protocol`, their Ceiling_Priority must be higher or equal to the property Priority of all threads connected to the data.
And so on ...

- **Require**: For each Data with $Concurrency\_Control\_Protocol = Priority\_Ceiling\_Protocol$, their $Ceiling\_Priority$ must be higher or equal to the property $Priority$ of all threads connected to the data.

```plaintext
RULE Ceiling_Priority FOR ( Data );
WHERE
RR13 : ( SIZEOF ( QUERY ( d <=* Data_Classifier | 
  ( SIZEOF ( QUERY ( p <=* Property | 
    ( ( p.Property_Name = 'Concurrency\_Control\_Protocol' ) AND 
      ( p.VALUE = 'Priority\_Ceiling\_Protocol' ) ) ) ) = 1 ) AND 
    ( SIZEOF ( QUERY ( c <=* Access\_Connection | 
      ( ( c.accessed_component = d ) AND 
       ( SIZEOF ( QUERY ( t <=* Thread\_Type | 
         ( SIZEOF ( QUERY ( f <=* t.features | 
           ( f = 
             c.requiring_feature ) AND 
           ( d.ceiling_priority < 
             t.priority ) ) ) ) = 0 ) ) ) ) = 0 ) ) ) ) = 0 );
END_RULE;
```
Outline

1. Subset Examples
2. Superset: an AADL Meta-Model
3. Examples of cardinality constraints
4. Mapping towards REAL and Prolog
5. Conclusion
Mapping towards REAL and Prolog

There is no data component

**EXPRESS:**

```plaintext
RULE No_Data_Instance FOR ( Data_Instance );
WHERE
   R–TT–C2 : SIZEOF ( Data_Instance ) = 0;
END_RULE;
```

**Prolog:**

```plaintext
isSubcomponent(_, _, _, _, 'DATA', _, _, _) → write('error R–TT–C2'); true.
```

**REAL:**

```plaintext
theorem Check_R_TT_C2
   foreach s in System_Set do
      check (Cardinal (Data_Set) = 0);
   end Check_R5_2;
```

- Work in progress.
- Can be produced automatically (e.g. Platypus).
Conclusion

- Problem:
  - What is a subset and how to express it?
  - Is there an uniform way to express the various examples of subsets/constraints?
- Approach:
  - Superset: an AADL meta-model to model the examples of subsets.
  - Can we express constraints of each subset as a cardinality constraint on superset?
- Results:
  - For the considered subset examples, we are able to express all their constraints as cardinality constraints on superset.
- Perspectives/roadmap:
  - Finalize translation of constraints in REAL and Prolog. Relationships with the constraint annex ⇒ next meeting?
  - Express other subsets with cardinality constraints? Oleg?
  - Cardinality may simplify ordering of subset: can we order proposed subsets?
We would like to thank Ellidiss Technologies and Region Bretagne for their support to this project.