Bindings & Mappings between Two System Hierarchies

Peter Feiler
Jan 2016

Bindings between System Hierarchies

AADL supports a (primary) containment hierarchy

Semantic connections represent flow between and within subtrees

- Managed interaction complexity by requiring connections up and down the hierarchy to restrict arbitrary connectivity
- Note: for subprogram calls we offer both a connection and a mapping specification

Deployment bindings are a mapping from elements of one subtree to elements of another subtree

- The subtrees represent different system layers in a layered architecture with the lower layer typically representing resources to the higher layer
- Bindings represent resource allocation

Binding Issues

Bindings are currently expressed by properties

Binding related properties are not distinguishable from others

- Properties that express bindings
- Properties that relate to bindings

EMV2 propagation paths are derived from bindings and binding points currently are identified by special keyword

Binding properties reach down the instance containment hierarchy

- A primary driver for introducing contained property associations

Binding Specification Proposal

User-definable binding types

- Introduce name and source/target model elements

Binding instances

- Deployment configuration

Binding points

- Binding constraints, properties, propagation paths

Binding Type

Define binding type

- User defined name, sets of source and target categories

```
Binding ProcessorBinding : { thread, thread_group, process} -> {
processor, virtual processor, system }
```

1-to-n binding Alternatives (*):

- example – multiple cores

```
Binding MultiCoreBinding : { thread, thread_group, system } ->* {
processor}
```

1-to-n binding Sequence (+):

- example – connection binding
- Binding to flows in the target architecture represents sequences
- Do we still need sequences here?

```
Binding ProtocolBinding : { port_connection, virtual_bus } ->+ {
bus, virtual_bus, system }
```

Definitions placed in a container similar to property types in property sets

Flows and Bindings

End-to-end flow & flow implementation across virtual and hardware platform elements

- Expressed by end to end flow declaration

Layered virtual buses

- Access connections between virtual buses (interconnected virtual channels)
- Flows across interconnected virtual buses
- Binding of connections to flows

Binding Instances

- Binding of unchangeable source and target hierarchies
- Declared as a configuration section
- Associated with component that contains both source and target subtrees
- Can be applied to implementations and configurations

```
System AS.deploymentconfig configures AS.systemconfig
Bindings
-- single binding target
ProcessorBinding : Appsys.sub.proc.thread1 -> platform.node.cpu1;
ProcessorBinding : Appsys.sub.proc.thread2 -> platform.node.cpu2;
-- multiple alternatives
ProtocolBinding : Appsys.sub.proc.conn1 -> (platform.proto1,
platform.proto2);
-- sequence
ProtocolBinding : Appsys.sub.proc.conn1 -> (platform.network.bus1 +
platform.network.bus2);
```

Partial and Nested Bindings

Partial binding configurations

- Partially configured source and target system
  - Only for those elements that have been configured
- Subset of elements are bound
  - Bindings cannot be overridden

Nested binding configurations

- Subsystem may be internally fully bound
- Configuration that includes binding has been configured in

Visibility of subcomponent hierarchy

- Configurable subsystems use prototype to hide internal hierarchy

Endpoints of Bindings

Source and target of bindings are components

- They act as implicit binding points
- Binding type definition tells us which component categories require binding declarations

Constraints on bindings

- In V2: Allowed_xxx_Binding_Class and Allowed_xxx_Binding on source of binding
- V3 options
  - Property: Allowed_binding_Class => xxx applies to <component>.<bindingtype>
    - Implicit binding point via binding type
    - Property applies to source of binding
    - How to express property on target of binding?
  - Via binding types that limit endpoints to specific component classifiers
  - Via constraint language

Properties on Bindings

Properties on bindings

- On binding point
- On binding
    - Should bindings have names like connections?

Bindings and EMV2

In EMV2 binding related propagation points are identified by

- Special keywords (V2)
  - For sources: processor, memory, connection, binding,
  - For targets: bindings (all bindings with component as target)
- the binding type name (V3)
  - In/out indicates direction of propagation: could be from source to target or vice versa
  - Components involved as source in one binding and destination in another binding of same type
    - Virtual processor is both source and target of processor binding
    - As source and as target it can be on the outgoing and incoming end of propagations

Named Binding Instances

Names on binding instances to be able to associate property values with each instance

```
System AS.deploymentconfig configures AS.systemconfig
Bindings
-- similar to connection declarations
PB1: ProcessorBinding Appsys.sub.proc.thread1 -> platform.node.cpu1;
PB2: ProcessorBinding Appsys.sub.proc.thread2 -> platform.node.cpu2;
```

Explicit Binding Point Definition

Explicit in features section:

- Identifies source or target of binding
- optional classifier(s) restrict the type of target

```
Virtual bus proto1
features
RequiredProtocols: Requires binding ProtocolBinding [ VBOrBusClassifier ];
ProvidedService: Provides binding ProtocolBinding [ VBOrBusClassifier ];
```

- Binding related properties
  - Properties are currently on the component being bound or the target
    - Separate properties for each of the binding types
  - Properties on binding points
    - Number of acceptable binding sources
    - Subset of resource capacity available through binding point