Objectives

- Provide overview of propose UML Profile for SAE AADL
- Review key features of UML Domain Model for SAE AADL
- Demonstrate profile AADL through use on existing avionics system
Outline

- Extending UML
  - Draft UML Domain Model for AADL
  - AADL/UML Profile
  - AADL/Example
  - Summary of Part II

AADL/UML Relationship

- Extensible AADL Annexes UML Working Groups
- To Be submitted to OMG for Adoption
- AADL Core
- AADL UML Profile
- UML 2.0
- UML 1.4 Detailed design
- Safety
- Dependability
- RT Scheduling
- UML-RT Performance Timeliness
- Investigating overlap & integration

Peter Feiler of the Software Engineering Institute, Carnegie Mellon University. Used with permission.
Extending UML

- UML provides modeling concepts & notations for typical software modeling projects
- Users may need
  - Additional features and/or notations
  - Non-semantic information attached to models
- UML core concepts can be extended or specialized by users
  - 3 built-in extension mechanisms
    - Stereotype
    - Constraint
    - Tagged Value
  - Can be used separately or together
- Can extend UML metamodel by explicitly adding new metaclasses & other meta-constructs
  - Depends on modeling tools or use of meta-metamodel facility

UML Profile

- Specializes UML for specific domain or process
- Predefined package of
  - Stereotypes
  - Tagged values
  - Constraints
  - Stereotype-specific icons
- Does not extend UML by adding any new basic concepts
- Provides conventions for applying & specializing standard UML to particular environment or domain
- (As defined in UML 1.5)
Stereotypes

- Classify model elements at object-model level
  - Instances of stereotyped element behave as if they were instances of new metamodel classes whose form is based on existing "base" metaclasses

- Augment UML classification mechanism based on built-in UML metamodel class hierarchy

- Adds "virtual" UML metaclasses with new
  - Semantics
  - Meta-attributes
  - Property lists
  - Constraints
  - Graphical representation

Stereotypes (Cont.)

- Names of new stereotypes must not clash with
  - Names of predefined metamodel elements
  - Names of other stereotypes

- A model element can be marked by 1 stereotype
  - Also called "classified by" or "stereotyped"
  - Stereotype can be constructed as specialization of other stereotypes
  - Receives features & semantics defined for stereotype

- Intent is that tools & repositories be able to manipulate stereotyped element
  - Same as ordinary element for most editing & storage purposes
  - Differentiating it for certain semantic operations, such as well-formedness checking, code generation, or report writing
Property Lists & Tagged Values

- Any modeling element may have “arbitrary” information attached in form of property list

- Property List consists of tag-value pairs
  - Tag is user-definable unique name string for property
  - Value is string
    - “Arbitrary” from UML’s perspective
    - May be constrained by definer
    - May be meaningful to tools

- Stereotype may require specific
  - Set of tags
    - "pseudo-attributes"
  - Optional default values
    - "constraints"

Constraints

- Semantic condition or restriction
  - Boolean expression associated with model element(s)
    - Must be true for the model to be well formed
  - Assertion not an executable statement
  - Certain constraints are predefined in UML

- 3 forms
  - Invariant
  - Precondition
  - Postcondition

- May be expressed in UML’s Object-Constraint Language (OCL)

- May be associated with specific stereotype to define semantics
Benefits of Extending UML

- Architects can represent system architecture graphically using commonly available UML tools
- UML tool developers can add advance support for AADL to existing tools rather than developing new tools
  - e.g. safety analysis
- Software designers can take defined architecture & refine software components
  - rather than common practice of re-creating architecture in software development tools
- System integrators should have easier time integrating
  - Software components generated by UML tools, or hand-code based on UML specification
  - Executive and architectural glue code that is generated by AADL tool
  - Target hardware.

Outline

- Extending UML
- Draft UML Domain Model for AADL
- AADL/UML Profile
- AADL/Example
- Summary of Part II
Overview of the UML Profile for the
Architecture Design & Analysis Language

UML Domain Model of AADL v0.99 (Balloted)

Overview

Model Data Types

Basic Concepts

Annexes

global

Core Concepts

UML Domain Model of AADL v0.99 (Balloted)

Core Concepts

Named-Element (from Namespace)

Element (from Element)

Namespace (from Namespace)

Package (from Packages)

Port-Group-Type (from Port-Group Type)

Component-Classifier (Non Component Classifiers)

Parameter-Specification (Non Component Specification)

Component-Instance-Specification (Non Component Specification)

Feature-Instance (from Feature Instances)

System-Instance (Non Component Instance)

Subcomponent (Non Component Instance)

Component-Instance (Non Component Instance)

Property-Type (from Property Definition)

Mode (from Modal Behavior)

Property-Sef (Non Property Sets)

Property-Definition (from Property Definitions)

Property-Set (Non Property Sets)

Parameter-Specification (from Subprogram Features)

Named-Element (from Namespace)

Feature-Specification (Non Features)

Behavior (from Behavior)

Instance (from Instance Model)

Feature-Instance (Non Feature-Instances)

Component-Instance (Non Component Instance)
Overview of the UML Profile for the Architecture Design & Analysis Language

UML Domain Model of AADL v0.99 (Balloted)

AADL Architecture Specification

UML Domain Model of AADL v0.99 (Balloted)

System Instance
Overview of the UML Profile for the Architecture Design & Analysis Language

UML Domain Model of AADL v0.99 (Balloted)

Components

- Component-Classifier
- Extensible-Element (from Extensible & Redefinable Elements)
- Namespace (from Namespace)
- Component-Type (from Component Types)
- Realization (from Extensible & Redefinable Elements)
- Component-Implementation (from Component Implementations)
- Type (frommg e class)
- realization (for Realizable Element)
- implementation (for Component Implementation)
- Feature-Specification (from Features)
- Annex-Subclause (from Annexes)
- Flow (from Flows)
- Property-Definition (from Property Definitions)
- Property-Specification (from Properties-Specification)
- DefinedProperties (from Properties)
- DefinedFlows (from Flows)
- DefinedAnnexes (from Annexes)
- DefinedFeatures (from Features)
- DefinedFlows (from Flows)
- DefinedAnnexes (from Annexes)
- DefinedFeatures (from Features)
- DefinedProperties (from Properties)
- DefinedFeatures (from Features)
- DefinedProperties (from Properties)
- DefinedFeatures (from Features)
- DefinedProperties (from Properties)
UML Domain Model of AADL v0.99 (Balloted)

Component Types

Component-Type

+ rootAccessingClassifier
  + rootDefiningClassType

+ definedRequiredComponent
  + requiredComponent
  + inheritedRequiredComponent

+ /accessingClassifier
  + /providingClassifier

+ /definedProvidedSubcomponent
  + /providedSubcomponent
  + inheritedProvidedSubComponent

Component-Instance-Specification (from Component Specification)
Overview of the UML Profile for the Architecture Design & Analysis Language

Sea World Aviation Conference 2004

Copyright 2001-4. Absolute Software Co., Inc.
3 November 2004
Overview of the UML Profile for the Architecture Design & Analysis Language

UML Domain Model of AADL v0.99 (Balloted)

Ports

+ /complementaryBasicPortSpec
| (subsets complementaryPortSpecification, derived, optional)
| Basic-Port-Specification
| + direction : Port-Direction
| 0..*
| + /complementaryBasicPortSpecification
| | (subsets complementaryPortSpecification, derived, optional)
| | Basic-Port-Specification
| | + direction : Port-Direction
| | 0..*
| + /equivalentBasicPortSpecification
| | (subsets equivalentPortSpecification, derived, optional)
| | Basic-Port-Specification
| | + direction : Port-Direction
| | 0..*

Event-Port-Specification
Data-Port-Specification
Event-Data-Port-Specification

Sample Constraints (Invariants)

- **Component Classifier**
  - a component classifier is owned by exactly 1 namespace, if it is not defined in a package then it is owned by the anonymous namespace.
  - this classifier’s category must appear in the applicableCategories of all definedPropertiesComponent Type

- **Component Instance**
  - required and provided components must be data or bus types

- **Component Implementation**
  - name must be unique relative to other implementations of the same component type
  - category must be same as the category of the type it realizes component type
  - Subcomponents is the union of inheritedSubcomponents and defineSubcomponents
  - inheritedSubcomponents is the union of subcomponents of the Component Implementation this implementation extends and requiredComponent
  - requiredComponents has the same set of components as the requiredComponents of the type this implementation realizes

Will be defining in UML's OCL
Outline

- Extending UML
- Draft UML Domain Model for AADL
- AADL/UML Profile
- AADL/Example
- Summary of Part II

UML Model of AADL v0.99 (Balloted)

Stereotype Mapping for Component Types
Outline

- Extending UML
- Draft UML Domain Model for AADL
- AADL/UML Profile
- AADL/Example
- Summary of Part II

Flight Manager in AADL Graphical Notation

- Navigation Sensor Processing
- Integrated Navigation
- Guidance Processing
- Flight Plan Processing
- Aircraft Performance Calculation
- Fuel Flow
- Nav sensor data
- Nav data
- Nav sensor data
- Navigation data
- Performance data
- Flight Plan data
- To Partitions
- From Partitions
Flight Manager Example Using UML Profile
Classifier View: Navigation Package

<<AADLData>>
Sensor_Data_Type
---------
AADLProperties
Source_Data_Size = 10 B

<<AADLData>>
Integrated_Data_Type
---------
AADLProperties
Source_Data_Size = 32 B

<<AADLData>>
Signal_Data_Type
---------
AADLProperties
Source_Data_Size = 100 B

Flight Manager Using UML Profile
Classifier View: Navigation Package (Possible Alternative Representation)
Flight Manager Example Using UML Profile
Classifier View: Guidance Package

```
<<AADLPackage>>
Guidance

<<AADLThread>>
Controller_Type

----------
AADLFeatures
<<AADLPort>> Nav_Data in Navigation::Integrated_Data_Type
<<AADLPort>> FP in Flight_Management::Flight_Plan_Type
<<AADLPort>> Nav_Signal_Data out Guidance_Data_Type

AADLProperties
Period = 50ms -- i.e. 20Hz
Dispatch_Protocol = Periodic

<<AADLData>>
Guidance_Data_Type

----------
AADLProperties
Source_Data_Size = 100 B

<<AADLPackage>>
Guidance
```

Flight Manager Example Using UML Profile
Classifier View: Aircraft Performance Package

```
<<AADLPackage>>
Aircraft_Performance

<<AADLSubprogram>>
Calculate_Performance

----------
AADLFeatures
<<AADLParameter>> Plan : in Flight_Management::Flight_Plan_Type
<<AADLParameter>> Nav_Data : in Navigation::Integrated_Data_Type
<<AADLParameter>> Fuel_Flow : in Fuel_Management::Fuel_Data_Type
<<AADLParameter>> Performance : out AC_Performance_Data_Type

<<AADLSubprogramImplementation>>
Calculate_Performance.F15_Implementation.

----------
AADLProperties
Compute_Execution_Time = 25ms

<<AADLSubprogramImplementation>>
Calculate_Performance.Harrier_Implementation.

----------
AADLProperties
Compute_Execution_Time = 35ms
```
Flight Manager Example Using UML Profile
Classifier View: Aircraft Performance Package
(cont.)

<<AADLPackage>> Aircraft_Performance

<<AADLData>> Performance_Data_Type

AADLFeatures
<< AADLSubprogram>> Calculate : Calculate_Performance

<<AADLRealization>>

<<AADLData>> Performance_Data_Type.F15_Implementation

AADLFeatures
<< AADLSubprogram>> Calculate : Calculate_Performance.F15_Implementation

AADLProperties
Source_Data_Size = 20 KB

<<AADLRealization>>

<<AADLData>> Performance_Data_Type.Harrier_Implementation

AADLFeatures
<< AADLSubprogram>> Calculate : Calculate_Performance.Harrier_Implementation

AADLProperties
Source_Data_Size = 40 KB

<<AADLThread>> Calculator_Type

AADLFeatures
<< AADLPort>> Nav_Data : in Navigation::Integrated_Data_Type
<< AADLPort>> FP_Data : in Flight_Management::Flight_Plan_Type
<< AADLPort>> Fuel_Data : in Fuel_Management::Fuel_Data_Type
<< AADLPort>> Performance_Data : out Performance_Data_Type

AADLProperties
Dispatch_Protocol = Periodic

<<AADLThreadImplementation>> Calculator_Type.F15_Implementation

AADLFeatures
<< AADLPort>> Performance_Data : refines to Performance_Data_Type.F15_Implementation

AADLProperties
Period = 250ms -- i.e. 4Hz

<<AADLThreadImplementation>> Calculator_Type.Harrier_Implementation

AADLFeatures
<< AADLPort>> Performance_Data : refines to Performance_Data_Type.Harrier_Implementation

AADLProperties
Period = 250ms -- i.e. 4Hz
Flight Manager Example Using UML Profile
Behavior View: Aircraft Performance Package’s Calculator_Type.Base Thread Impl.

<<AADLThreadImplementation>>
Calculator_Type.Base

<<AADLMode>>
Default_Mode

AADLCallSequences
Default_Periodic_Calls : AC_Performance_Data_Type.Calculate( FP_Data, Nav_Data, Fuel_Data, Performance_Data);

Flight Manager Example Using UML Profile
Classifier View: Flight Management Package

<<AADLPackage>>
Flight_Management

<<AADLData>>
Flight_Plan_Type

<<AADLPortGroup>>
Flight_Sensor_Group_Type
AADLPorts
<<AADLPort>> Nav_Data in Navigation::Sensor_Data_Type
<<AADLPort>> Fuel_Flow in Fuel_Management::Flow_Data_Type

<<AADLPortGroup>>
Flight_Status_Group_Type
AADLPorts
<<AADLPort>> Nav_Data out Navigation::Nav_Data_Type
<<AADLPort>> Guidance_Data out Guidance::Guidance_Data_Type
<<AADLPort>> Flight_Plan out Flight_Plan_Type

<<AADLThread>>
Flight_Planner_Type
AADLPorts
<<AADLPort>> Guidance_Data in Guidance::Guidance_Data_Type
<<AADLPort>> FP_Data out Flight_Management::Flight_Plan_Type
<<AADLPort>> Performance_Data out Performance_Data_Type

<<AADLProcess>>
Flight_Manager
AADLPorts
<<AADLPortGroup>>
Sensors in Right_Sensor_Group_Type
<<AADLPortGroup>>
Status : out Right_Status_Group_Type
AADLProperties
Source_Data_Size => 500 KB;
Flight Manager Example Using UML Profile

Classifier View: Flight Management Package (cont.)

Generic Flight Mgr. Impl.

Flight Manager Example Using UML Profile

Classifier View: Flight Manager Implementation (cont.) – Nesting & Association
Flight Manager Example Using UML Profile
Classifier View: Flight Management Package (cont.)

F15 & Harrier Flight Mgr. Impl.

Outline

- Extending UML
- Draft UML Domain Model for AADL
- AADL/UML Profile
- AADL/Example

Summary of Part II
SAE AADL Summary (cont.)

UML Profile

- **Symbiotic Relationship**
  - System Architects can graphically using commonly available UML tools to represent software & hardware architecture
    - Can make use of UML tools that provide capabilities like simulation of state models
  - UML tool developers can easily integrate AADL tools for advance analysis techniques rather than developing new tools
    - e.g. safety analysis
  - Software designers can take defined architecture & refine software components
    - Rather than common practice of re–creating architecture in software development tools
  - System integrators should have easier time integrating
    - Software components generated by UML tools, or hand–code based on UML specification
    - Executive & architectural glue code that is generated by AADL tool
    - Target hardware

References