

SysML to AADL Bridge



Automating the Translation of SysML into AADL for Analysis and Refinement

Hazel Shackleton

hazel.shackleton@adventiumlabs.com

Steve Vestal

steve.vestal@adventiumlabs.com

DISTRIBUTION A. Approved for public release: distribution unlimited.

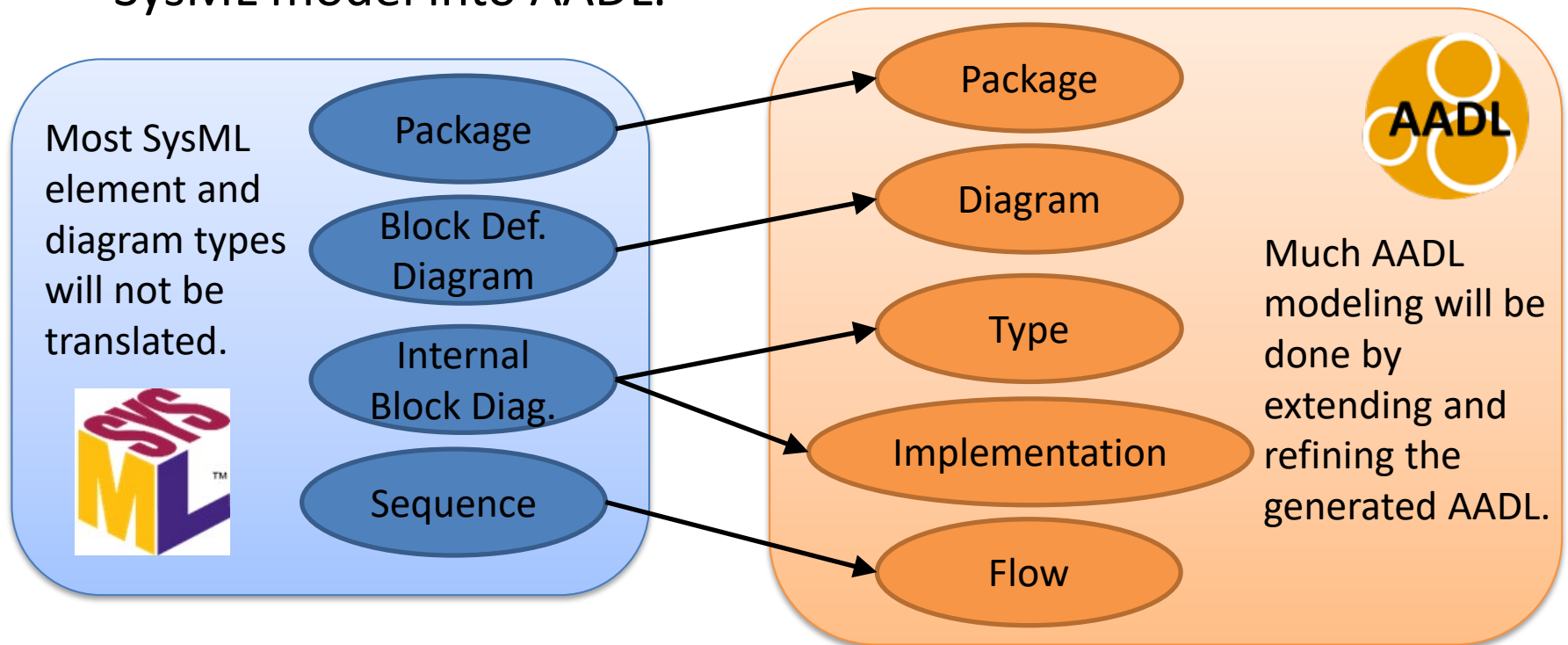
This material is based upon work supported by the U.S. Army Development Command (DevCOM), U.S. Army Combat Capabilities Development Command Aviation & Missile Center (DEVCOM AvMC), ATTN: Security Office, Bldg 401, Lee Blvd, Fort Eustis, VA 23604-5577 under contract no. W911W6-17-D-0003. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. Army.

Agenda

- Design goals & supported workflows
- Demo
 - Using the profile
 - Code generation
 - Analysis and round-trip engineering
 - Model refinement & integration
- Ongoing work (behavior & property mapping)
- Questions

Targeted Portions of SysML Only

- Allow users to annotate portions of SysML models using an AADL Profile and automatically translate those portions of the SysML model into AADL.



Clear Semantics

- Transition SysML specification from system engineering phase of development into standardized AADL semantics for ACVIP/AADL virtual integration analysis.
- Provide a common (across multiple programs, contractors) AADL Profile based on the AADL standard that is amenable to extension for additional AADL properties and annexes (for integration of additional analysis tools).

Provided SysML AADL Profiles:

- Core AADL stereotypes and properties
- Extension profiles for ARINC 653, MILS, SESSAF (STPA), FASTAR, MADS
MagicDraw/Cameo Enterprise Architecture and Sparx Enterprise Architect

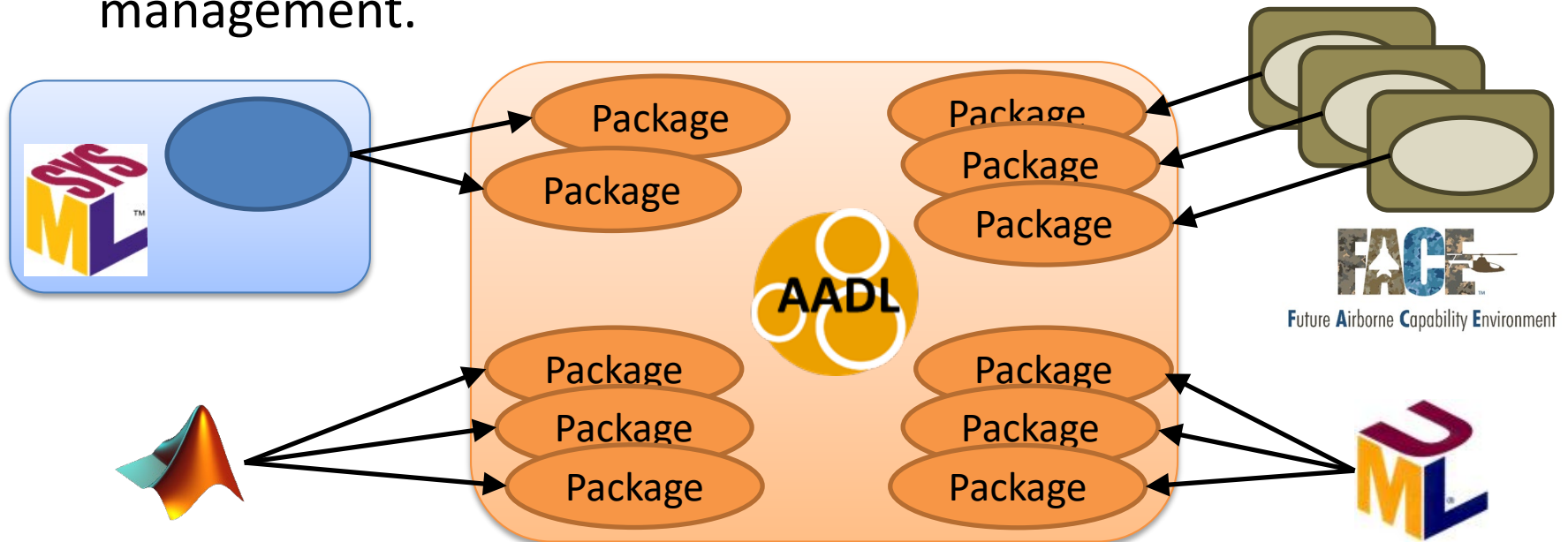
Mapping of Core AADL Stereotypes

(Excerpt from Modeling Guidelines)

AADL stereotype	SysML element	Generated AADL declaration(s)	AS5506C
AADL_Package	package	package in a file	4.2
AADL_Abstract	block	abstract type + implementation	4.6
AADL_Data	block	data type + implementation	5.1
AADL_Subprogram	block	subprogram type + implementation	5.2
AADL_Thread	block	thread type + implementation	5.3
AADL_Thread_Group	block	thread group type + implementation	5.4
AADL_Process	block	process type + implementation	5.5
AADL_Processor	block	processor type + implementation	6.1
AADL_Virtual_Processor	block	virtual processor type + implementation	6.2
AADL_Memory	block	memory type + implementation	6.3
AADL_Bus	block	bus type + implementation	6.4
AADL_Virtual_Bus	block	virtual bus type + implementation	6.5
AADL_Device	block	device type + implementation	6.6
AADL_System	block	system type + implementation	7.1
AADL_Feature	port	feature (in a type declaration)	8.1
AADL_Feature_Group	port	feature group (in a type declaration)	8.2
AADL_Feature_Group	interfaceBlock	feature group type + implementation	8.2
AADL_Access	port	provides/requires access (in a type declaration)	8.6-8.8
AADL_Flow	interaction	flow source, path, sink, end-to-end	10

Workflow Support

- Virtual integration of computer system architecture model occurs in an AADL modeling environment.
- Modularized, textual, BNF-based exchange formats have been proven very useful for model exchange and configuration management.



Modeling Style Support

- SysML standards and conventions, natural for SysML aficionados.
- AADL standards and conventions, natural for AADL aficionados.
- Clear mapping from stereotyped SysML elements with tagged value properties to the AADL generated from those elements and properties.

Provided Documentation:

- SysML AADL Profile and Modeling Guidelines
- SysML example models

SysML AADL Profile and Modeling Guideline Topics

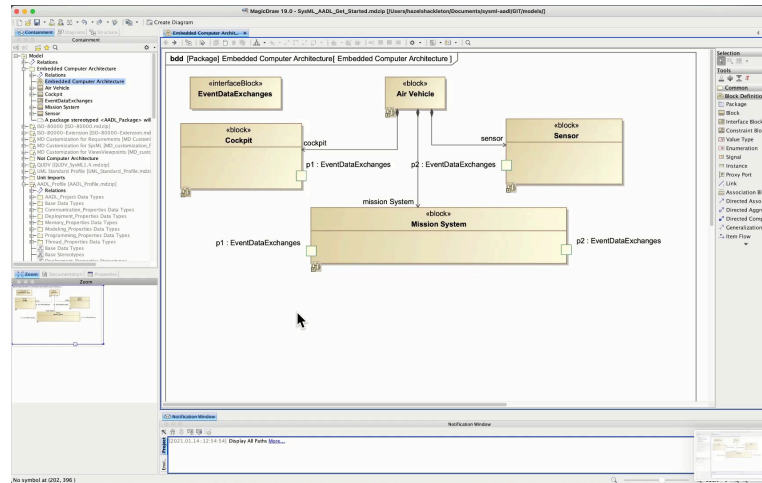
- Names and Name Spaces
 - Allocation and Binding Relations
 - ✓ Packages, Blocks, Types, and Implementations
 - ✓ Tagged Values and Properties
 - ✓ Ports, Features, and Connections
 - ✓ Interactions and Flows
 - ✓ Annex Profiles
-
- Not covered today
 - ✓ At least briefly discussed

(Quick Shout Out/Thank You)

- Our work was accomplished with oversight by a SysML AADL Working Group consisting consisting of Army Aviation system integrators and suppliers.
- Other work we looked at:
 - MARTE
 - Open Archive Toulouse Archive Ouverte (OATAO)
 - ExSAM Profile, 2011 (University of Oslo, Norway)
 - SCADE Architect to AADL translator
 - Rockwell Collins SysML to AADL translator v1.1 2016
 - SLICED AADL Profile for UML state machine diagrams

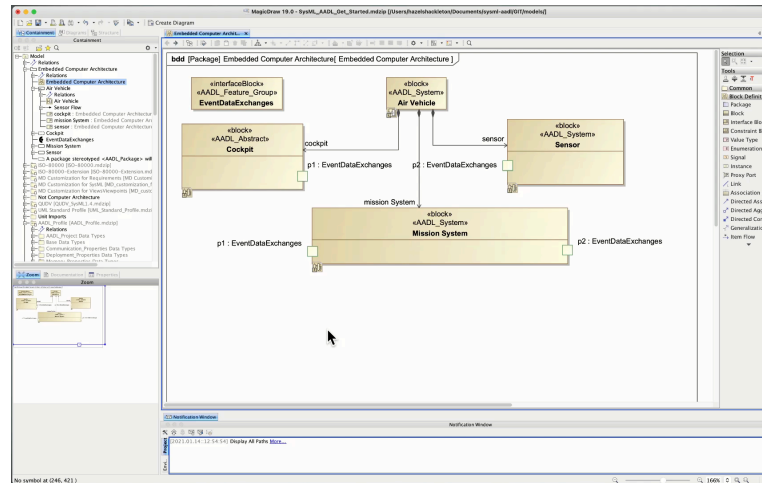
Selecting Elements for Translation

- Using AADL Profile to select Packages and Components for translation.
- Annotating the Block Definition Diagram.



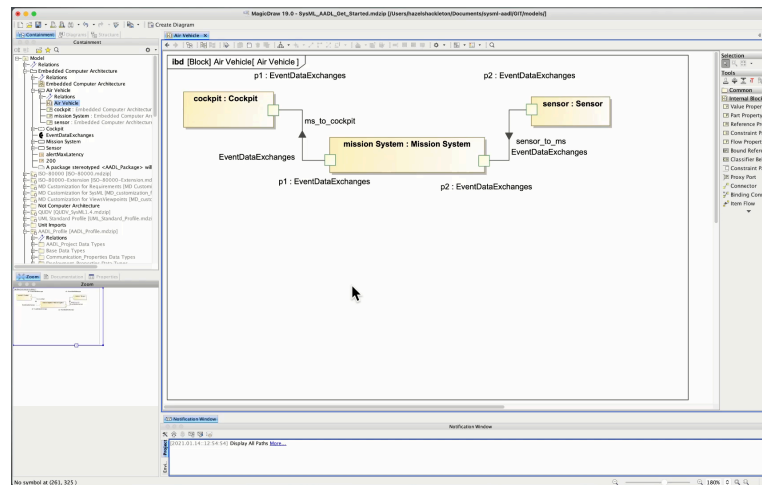
Annotating with Tag Values

- Adding detailed port and connections information using Stereotype Tags.
- Annotating the Internal Block Diagram.



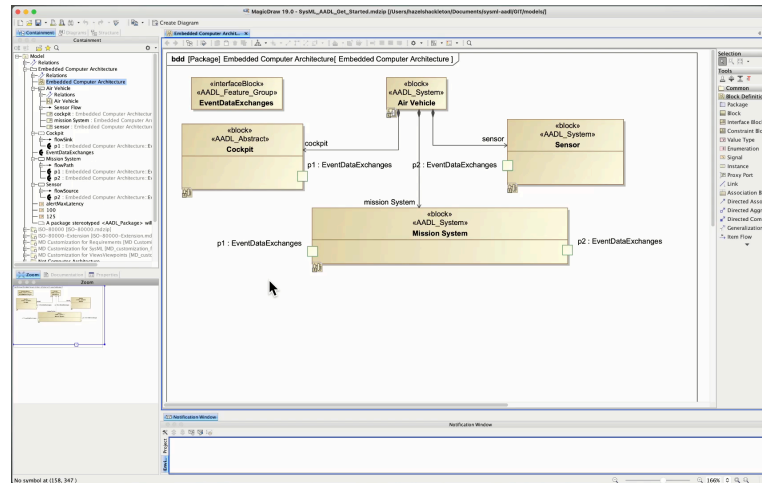
Adding End to End Flows

- Creating a sequence diagram to add a flow through existing components.



Round-trip Engineering

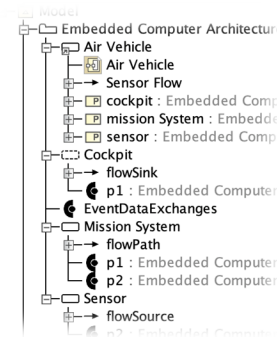
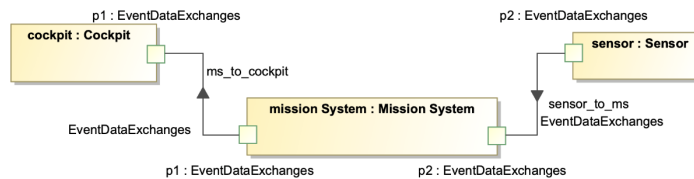
- Generation of AADL project.
- Mapping back from AADL to SysML using GUIDs.



Split Between SysML & AADL

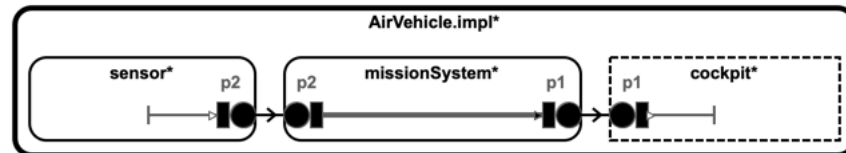


- Detailed AADL Analysis: where should additional property values and model details be added?
 - Option 1: Expand SysML model with refinements, new components, new tags/properties.
 - Option 2: Refine generated AADL model.



```

-- from SysML Air Vehicle (_19_0_2_c3f0303_1587146874690_365944_426)
system implementation AirVehicle.impl
subcomponents
  cockpit: abstract Cockpit; -- from SysML cockpit (_19_0_2_c3f0303_1587146874690_365944_426)
  missionSystem: system MissionSystem; -- from SysML missionSystem (_19_0_2_c3f0303_1587146874690_365944_426)
  sensor: system Sensor; -- from SysML sensor (_19_0_2_c3f0303_1587146874690_365944_426)
connections
  mstocockpit: feature group missionSystem.p1 -> cockpit.p1;
  sensortoms: feature group sensor.p2 -> missionSystem.p2; --
flows
  SensorFlow: end to end flow
    sensor.flowSource ->
      sensortoms ->
        missionSystem.flowPath ->
          mstocockpit ->
            cockpit.flowSink; -- from SysML Sensor Flow (_19_0_3_31c)
properties
  Latency => 100 ms .. 175 ms applies to SensorFlow;
end AirVehicle.impl;
    
```



Option 1: SysML Refinement

Example of use: Want to provide the property at the SysML level.

- Generalization/Inheritance refinement all supported by the translator.
- Create custom profiles for a missing AADL Property Sets.
 - Detailed instructions in provided documentation.
 - Any tag defined in a profile named “AADL_<something>_Profile” will be translated into an AADL property if a SysML model element has that tag.

Provided Example Model:

- Architecture_Refinement_Example

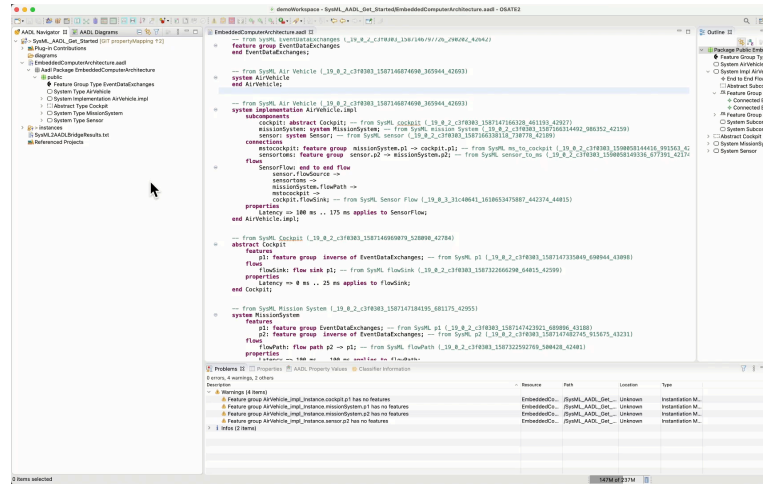
Provided SysML AADL Profiles:

- Core AADL stereotypes and properties
- Extension profiles for ARINC 653, MILS, SESSAF (STPA), FASTAR, MADS
MagicDraw/Cameo Enterprise Architecture and Sparx Enterprise Architect

Options 2: AADL Refinement

Example of use: Complex properties defined during the embedded design phase with added analyses as the fidelity of the model increases.

- Refine Mission System & Air Vehicle to include Error Modeling using EMV2 annex.



```

-- from SyML Air_Vehicle_19_R_2_378383_158714674698_365844_42853
and AirVehicle;

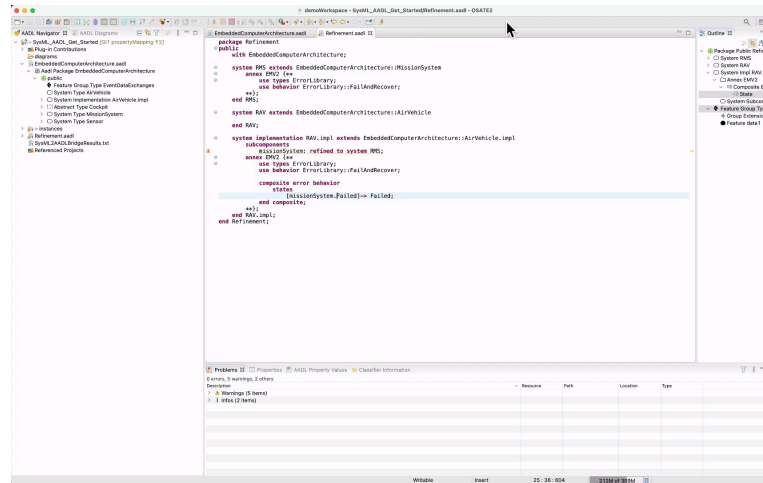
-- from SyML Air_Vehicle_19_R_2_378383_158714674698_365844_42853
abstract Cockpit1;
  feature group Cockpit1;
  sensor system Sensor1;
  flow path p1;
  properties
    latency => 100 ms .. 175 ms applies to SensorFlow;
end Cockpit1;

-- from SyML Cockpit_19_R_2_378383_158714698079_328998_427841
abstract Cockpit2;
  feature group Cockpit2;
  flow path p2;
  properties
    latency => 4 ms .. 25 ms applies to flowPath;
end Cockpit2;

-- from SyML Mission_System_19_R_2_378383_1587147181435_481179_429551
and MissionSystem;
  feature group EventDataChanges;
  feature group Sensor;
  flow path p1;
  flow path p2;
  properties
    latency => 100 ms .. 175 ms applies to SensorFlow;
end MissionSystem;
  
```


AADL Integration

- Integrate existing project feature types into generated AADL model.



```

package Refinement
with EmbeddedComputerArchitecture;
end package;

system RMS extends EmbeddedComputerArchitecture::MissioSystem
  error RMS;
  use type ErrorLibrary;
  use behavior ErrorLibrary::FailAndRecovery;
end RMS;

system SAV extends EmbeddedComputerArchitecture::AIRVehicle
end SAV;

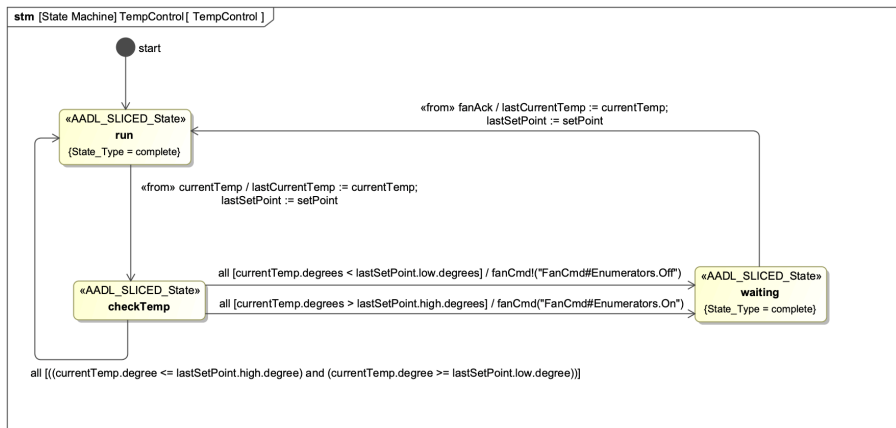
system Implementation_RMS_lapl extends EmbeddedComputerArchitecture::AIRVehicle_lapl
  subcomponents
  missioSystem: refined to system RMS;
  error RMS;
  use type ErrorLibrary;
  use behavior ErrorLibrary::FailAndRecovery;
end Implementation_RMS_lapl;

composite error behavior
  error
  [!MissioSystem_Palid]= Failed;
end composite;
end;
end RMS_lapl;
end Refinement;
  
```

Highlighted Ongoing Work (1 of 2)

- Behavior & Annexes under GUMBO (Grand Unified Modeling of Behavioral Operators) SIBR (Adventium and Kansas State).

State Machines



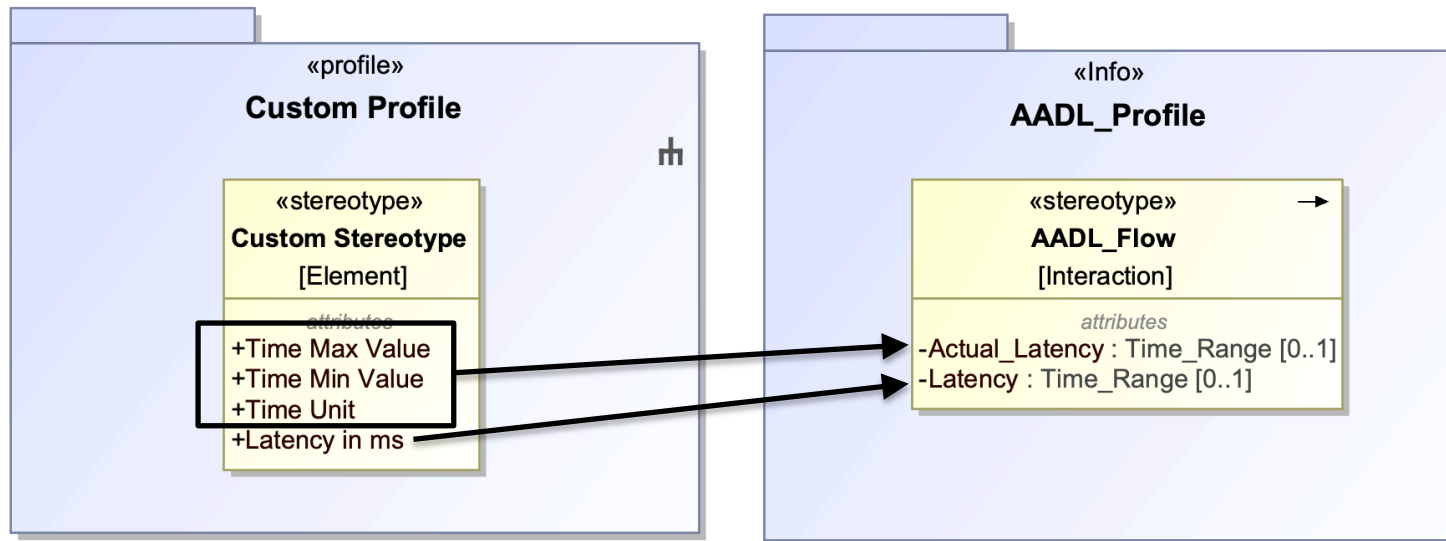
behavior_specification annex

```

-- from SysML TempControl (_19_0_3_31c40641_1601579128489_223139_43824)
thread implementation TempControl.impl
  -- from SysML behavior_specification (_19_0_3_31c40641_1603210321659_196478_43244)
  annex behavior_specification {**
    variables
      lastCurrentTemp: TempSensor::Temperature.impl; -- from SysML lastCurrentTemp (_19_0_3_31c40641_1605552704202_6640)
      lastSetPoint: SetPoint.impl; -- from SysML lastSetPoint (_19_0_3_31c40641_1605552856851_590884_42692)
    states
      start: initial state; -- from SysML start (_19_0_3_31c40641_1603210321780_889054_43277)
      run_GEN: complete state; -- from SysML run (_19_0_3_31c40641_1603210321781_504339_43278)
      checkTemp: state; -- from SysML checkTemp (_19_0_3_31c40641_1603210339288_606172_43302)
      waiting: complete state; -- from SysML waiting (_19_0_3_31c40641_1603210346516_999912_43330)
    transitions
      start --[]-> run_GEN; -- from SysML go (_19_0_3_31c40641_1603210321781_334721_43279)
      run_GEN --[ on dispatch currentTemp ]-> checkTemp
      {
        lastCurrentTemp := currentTemp;
        lastSetPoint := setPoint
      }; -- from SysML operate (_19_0_3_31c40641_1603210494219_991483_43435)
      --checkTemp --[ currentTemp.degree < lastSetPoint.low.degree ]-> waiting
      {
        fanCmd!("FanCmd#Enumerators.Off")
      }; -- from SysML sendOff (_19_0_3_31c40641_1603210641193_308048_43499)
      --checkTemp --[ currentTemp.degree > lastSetPoint.high.degree ]-> waiting
      {
        fanCmd!("FanCmd#Enumerators.On")
      }; -- from SysML sendOn (_19_0_3_31c40641_1603211054626_193106_43673)
      checkTemp --[]-> run_GEN; -- from SysML TempOk (_19_0_3_31c40641_1603211187988_570345_43706)
      waiting --[ on dispatch fanAck ]-> run_GEN
      {
        lastCurrentTemp := currentTemp;
        lastSetPoint := setPoint
      }; -- from SysML getAck (_19_0_3_31c40641_1603211323329_76081_43840)
    };
  };
end TempControl.impl;
  
```

Highlighted Ongoing Work (2 of 2)

- Mapping other existing profile tag values or properties to generated AADL properties.
- User must ensure mapping is semantically correct.



Unsupported AADL Features in Bridge

- Connection bindings that are ordered lists (routings)
- System operating modes
- Prototypes
- Subprogram parameters
- Subprogram call sequences
- Annex sub-languages (e.g., EMV2 Annex, Behavior Specification Annex)*
- Multiple implementations for a type must be declared as extensions to both the type and the implementation.

*Ongoing work

Questions?

<https://camet.adventium.com>

- SysML AADL Profiles
 - Core AADL stereotypes and properties
 - Extension profiles for ARINC 653, MILS, SESSAF (STPA), FASTAR, MADS
MagicDraw/Cameo Enterprise Architecture and Sparx Enterprise Architect
- Documentation
 - SysML AADL Profile and Modeling Guidelines
 - SysML example models
- *SysML to AADL Bridge Tool*
MagicDraw/Cameo Enterprise Architecture and Sparx Enterprise Architect