

“Advancing Traffic Efficiency and Safety  
through Software Technology”

# Behavior and Environmental modeling

David Servat (CEA LIST)  
Carl-Johan Sjöstedt (KTH)



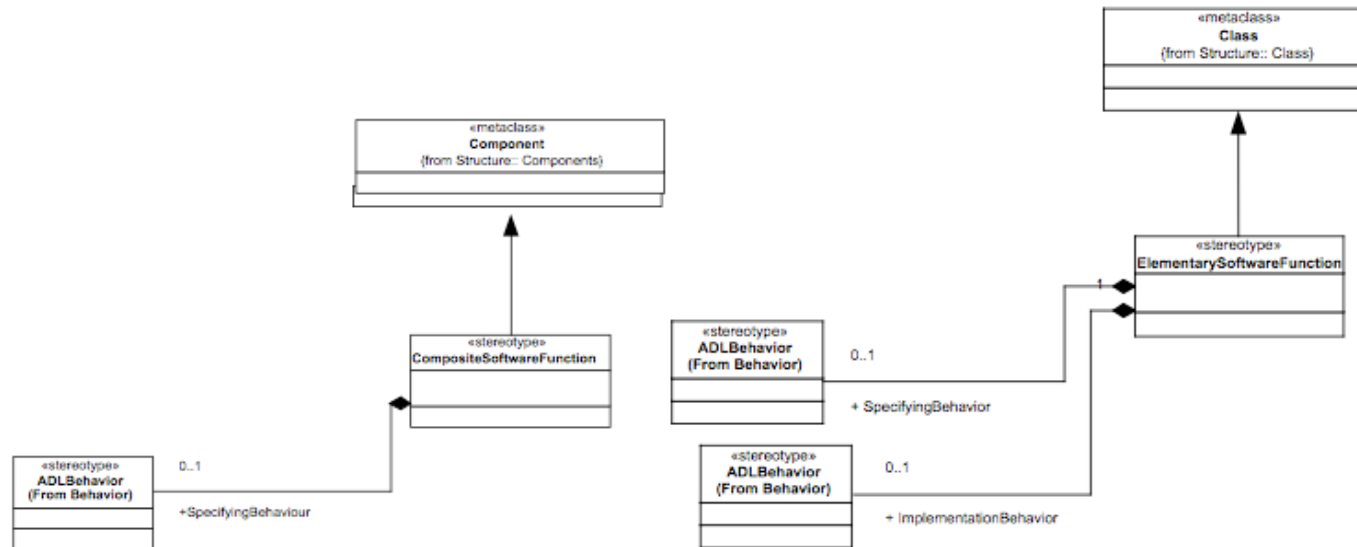
# Foreword

- Results of work task 3.2
- Main objectives
  - **Definition of elementary function behavior**
  - **Composition of elementary functions**
  - **Environmental modeling**
  - **Relation to external tools, such as Simulink, ASCET, etc.**
  - **Error behavior (will be dealt with in the next talk)**
- Three starting points
  - **The precise, rather constrained execution semantics of elementary function**
  - **The distinction between native and external behavior (inherited from EAST-AEE)**
  - **The idea to link up with UML2 behavioral constructs**

# Outline of the talk

1. Account on semantics investigation
2. Behavioral constructs presentation
3. Examples
4. Autosar projection
5. Environmental modeling
6. Perspectives

# EAST-ADL1 behavioral approach



Elementary functions are the behavioral units, they are the only functions that get both a specification behavior and an implementation behavior

Two types of behavior are introduced: *ExternalBehavior* (defined via third-party tool), *NativeBehavior* (supposedly defined by EAST-ADL constructs, mainly targeting UML2 framework, not developed in EAST-ADL1)

Elementary functions are executed on the assumption of synchronous execution (read inputs from ports, compute, write outputs on ports)

They are composed in higher level, composite functions which only have a specification behavior

The whole approach is in keep with ADLs (Architecture Description Languages) which favor a structural (architectural) representation of systems, rarely using behavioral diagrams...

## Prioritized goals resulting from new requirements

- Native definition of behavioral model  
This is an important enhancement for EAST ADL2 in order to support modeling and analysis of behavior with non-proprietary tools and enable model exchange.
- Relation of behavioral model with AUTOSAR behavior  
At least the design level of EAST ADL2 must have a clear mapping of behavior to AUTOSAR. The purpose of EAST ADL2 is to provide a functional definition of the EE systems that are independent of software architecture. It is thus necessary to be able to model behavior in a way that is really traceable to AUTOSAR behavior.
- Definition of behavioral model for external tools and alignment toward EAST-ADL  
Proprietary tools like Simulink and Ascet are likely to dominate automotive system development for the foreseeable future. It is therefore important to define how these behavioral representations are integrated in the EAST ADL2 system model
- Semantic for behavioral definition of environment function  
Environment models, or plant models, are necessary to verify the system definitions by co-simulation with the plant. It is thus necessary to support interfacing between the EE system and plant, and to support continuous behavior in time and value.

## Prioritized goals resulting from new requirements (seq.)

- Composition of behavioral models considering internal and external behavioral models

EAST ADL2 is meant to support re-use and integration of models from different developers. It is therefore necessary to support behavioral composition, i.e. to be able to put together several functions and have a predictable result regarding the common behavior.

- Error modeling and error behavior integration

Error behavior modeling is to some extent different from other behavioral definitions.

Rather than describing the nominal behavior such as data transformation and output data and events for the system functions, abstract error behavior is captured. The representation is focused on errors and how they propagate, based on which it is possible to see if a certain error has an effect in other parts of a system.

- Timing model of behavioral model

With the assumption of synchronous execution, timing model does not influence the internals of ADL functions. Once triggered, each function executes in a single-shot fashion, and the timing model does not constrain events during execution. The timing model has to support modeling of timing properties on the level of the composed system

# Perspectives

- ✓ The overall behavior and environmental model constructs were established in EAST-ADL2 including providing a mapping between Simulink and UML continuous/discrete behavior models
- ✓ Investigation led to find out possible mismatches in semantics w.r.t. to UML, can be dealt with:
  - ✓ more careful definition of the semantics attached to the stereotypes
  - ✓ building blocks (library of specific actions) for the definition of native behaviors inside activity diagrams (work started for environmental modeling and Simulink import/export)
- ✓ Definite need to lead some work in close loop with simulation tools (as was done for ErrorBehavior)

Further work in ATESST2!!

# UML2 behavior : a nice match in principle

**UML2 behavior** can be either:

- an *OpaqueBehavior*: (language, body) pair of properties, e.g. piece of code written in C, externally defined function, etc.
- an *Activity*: description of behavior in terms of control/data flows among a group of interconnected actions of various granularity - call to behaviors are possible. Based on UML2 action library, this provides main constructs of programming languages
- a *StateMachine*: description of behavior in terms of event/time driven behavior triggering, suitable also for protocol specification

All can be combined via actions calling various forms of behavior

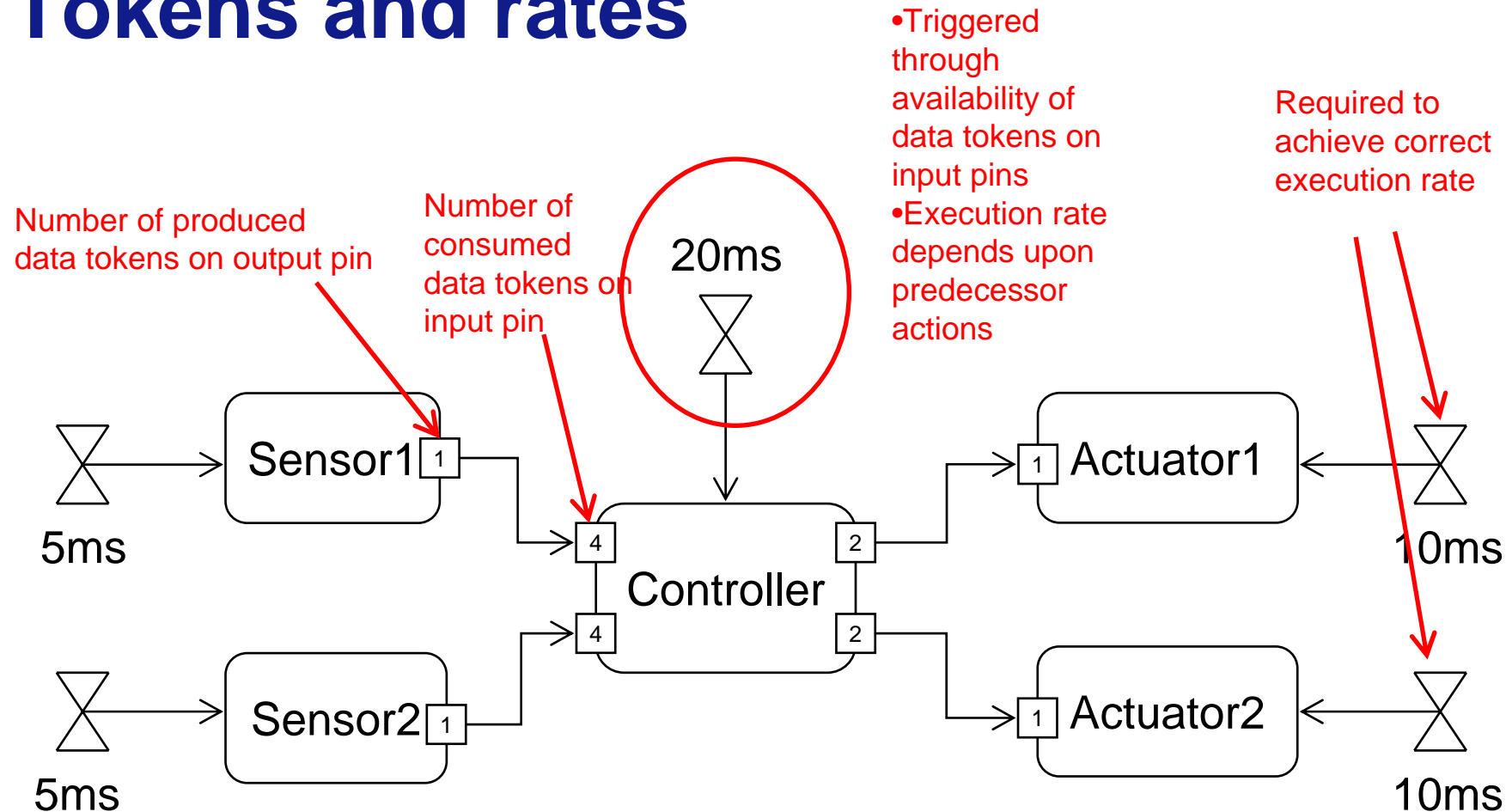
## **HOWEVER**

- ✓ Triggering mechanism is mainly object-oriented (i.e. message passing, operation/signal calls), which calls for dedicated semantics description for other triggering mechanisms
- ✓ Activity diagrams though graphically appealing and resembling what is common use in the domain *has* rather constrained semantics

# Multi-rate systems

- Appear as control engineer pattern at FDA (design level) for a composite function
- Require specific semantics some conflict with UML2
  - “Last-is-best” semantics of data tokens on output pins of Actions/ activity parameter nodes
  - Initialization of Actions includes provision of initial data tokens on output pins
  - Explicit triggering of Actions at a specific rate vs.number of data tokens emitted
- No definite, complete and sound solution can be found here
  - Use of specific stereotypes on Pins + disregarding UML2 token semantics: a quick and dirty solution yet cannot be used to simulate behavior
  - Use Seq. Diagrams (see Time modeling in relation to Marte)

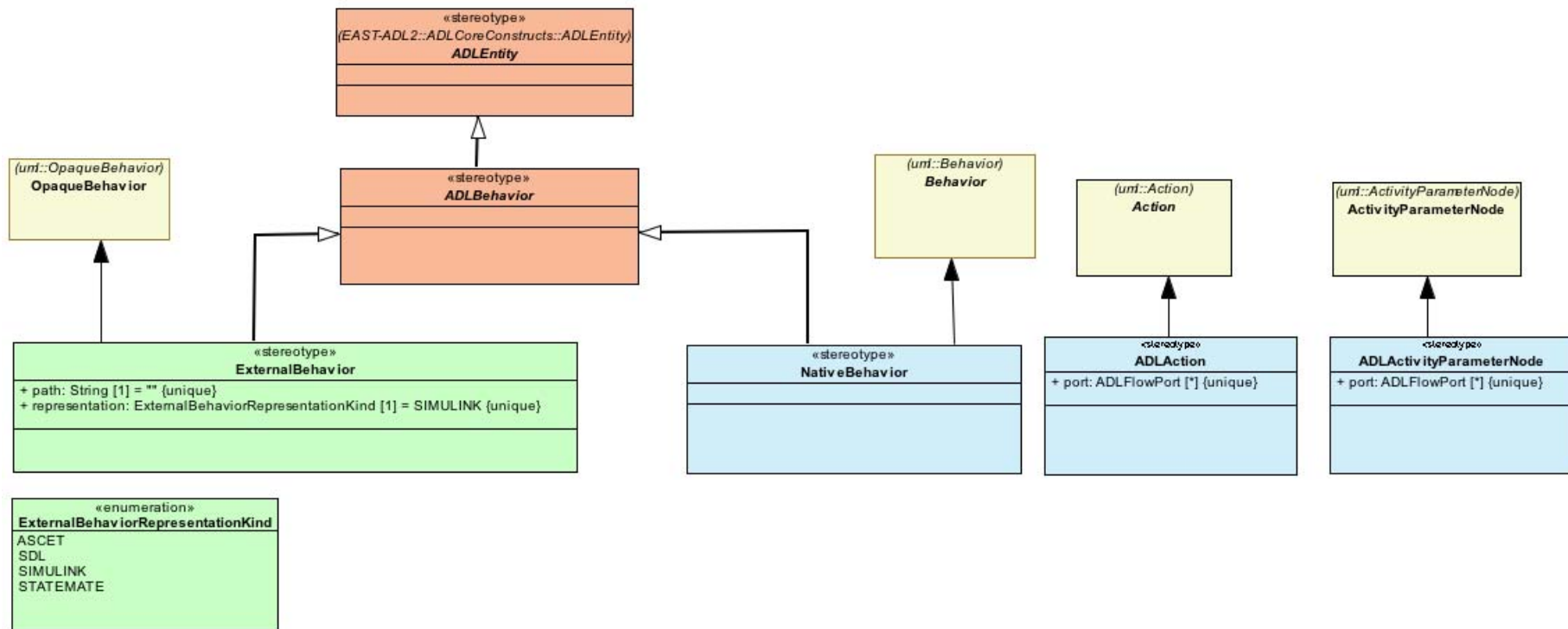
# Tokens and rates



# Summary

- Semantics issue: distinguish between
  - sound and prescribed way of modeling in EADL
  - partial solution for alternative ways of modeling, needing more investigation if analysis and/or simulation is looked for
- Fill in attributes in the domain model so that all information may be kept in a composite structure diagram which acts as our primary representation mode
- Add UML2 Behaviors to the elementary and composite functions:
  - Activity, StateMachine (we do not have an example though), OpaqueBehavior (for the link towards external tools), Interaction
- For composite functions: behaviors are spec!
- Allow for the time being multiple ways of expressing behavior, e.g. Interaction and Activity, let users choose what is best for their needs, provide examples in the deliverable to account for what either means enable / disable

# Behavior constructs 1/2



Distinction between external (ref. to file) and native, def. of modeling shortcuts

# Modeling shortcuts

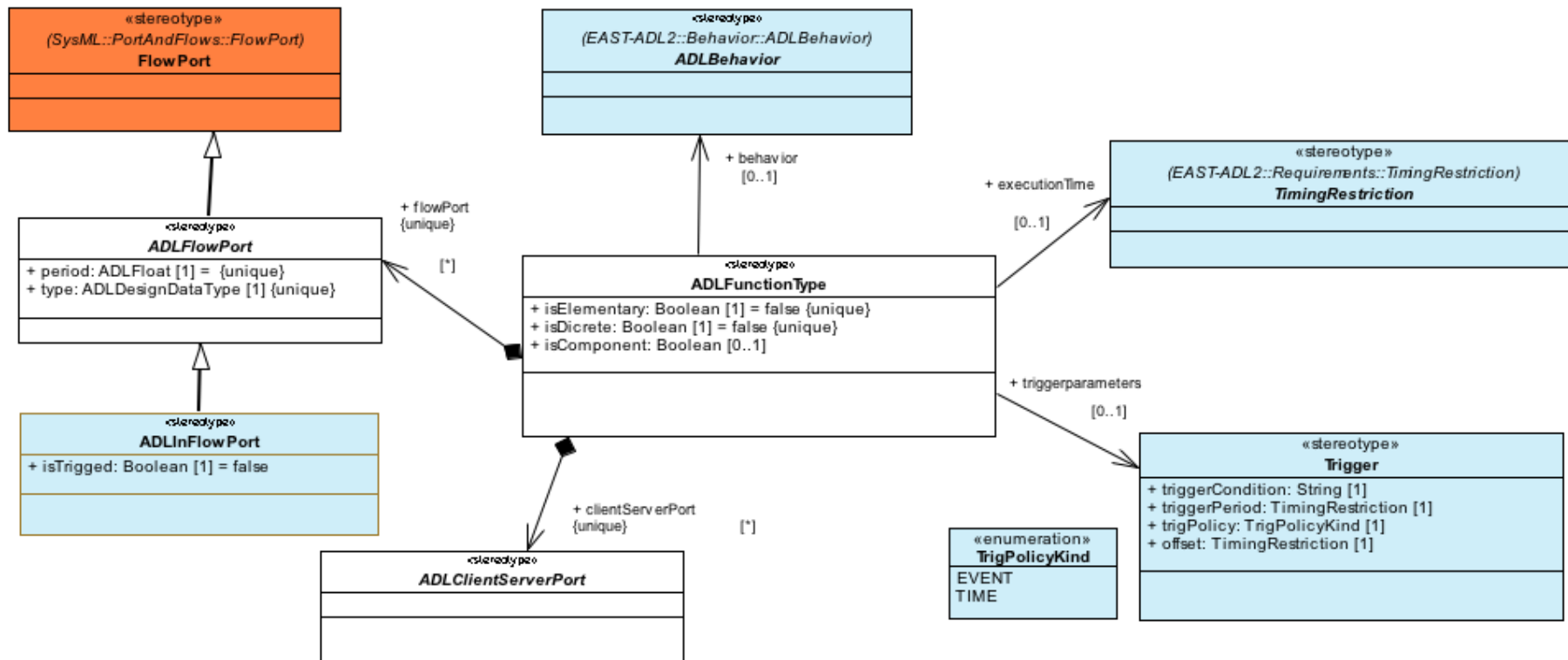
Several levels should be allowed to cope with users with various needs/skills w.r.t. to UML, namely to cope with data access on ports

- “*NativeBehavior*” and “*ExternalBehavior*” granted implicit access
- “*ADLFlowPort*” referencing list of operations directly without use of *Interfaces*
- “*ADLAction*” to reference ports on actions
- “*ADLActivityParameterNode*” to reference ports as parameters of activities
- “*ADLPin*” to reference ports on input/output pins of actions

Possible extensions:

a set of actions, e.g. *ReadDataFromPortAction*, regrouped in a *ModelLibrary*, part of the profile, for the real-experienced user and for sake of precision semantics - help show what is intended by our modeling shortcuts (MARTE approach btw), also extended with SignalProcessing (e.g. Simulink) basic actions

# Behavior constructs 2/2

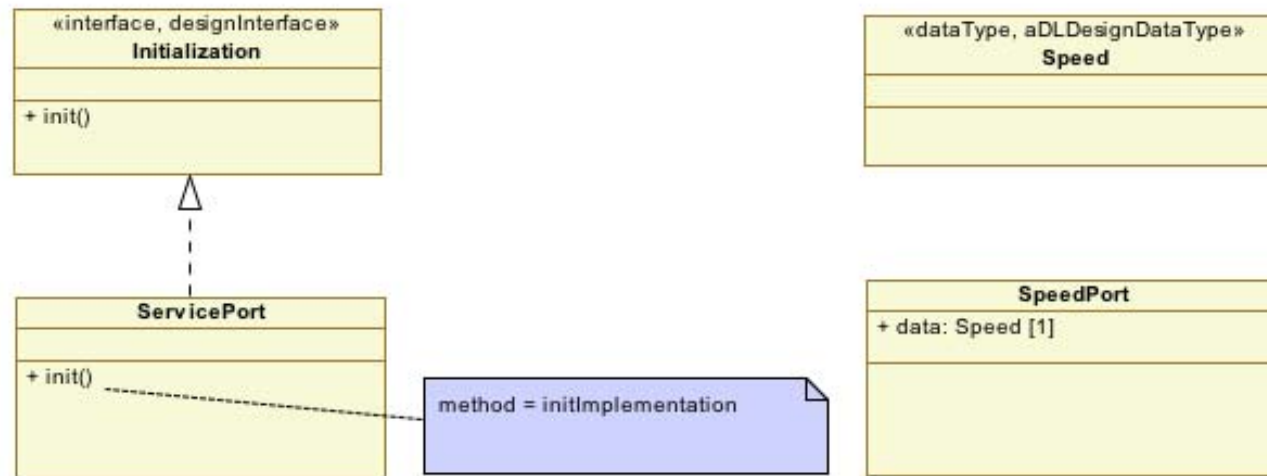


Link with ADLFunctionType: behavior, triggers, and triggered flow ports

# Trigger modes

Trigger: {periodic, event}	Defines the triggering policy. If this is set to 'time', then TriggerPeriod must be set. If this is set to 'event', then ADLInflowPorts tagged isTriggered = true. In <i>both</i> cases, TriggerCondition has to be defined.
TriggerPeriod: TimingRestriction	The trigger period of the ElementarySoftwareFunction, if periodic triggering is used.
TriggerCondition: String	A Boolean expression that must evaluate to true for this ElementarySoftwareFunction to execute. This value is used both for periodically and for event triggered Elementary-Software-Functions.
Offset: TimingRestriction	The offset in ms from the period start time. Applies to periodic triggering policy only (The ElementarySoftwareFunction becomes eligible for execution Offset time units after the period start time)
ExecutionTime: TimingRestriction	The execution time from the start (activation) to completion for the ElementarySoftwareFunction. ExecutionTime is therefore a tentative value that may be subject to back-annotation

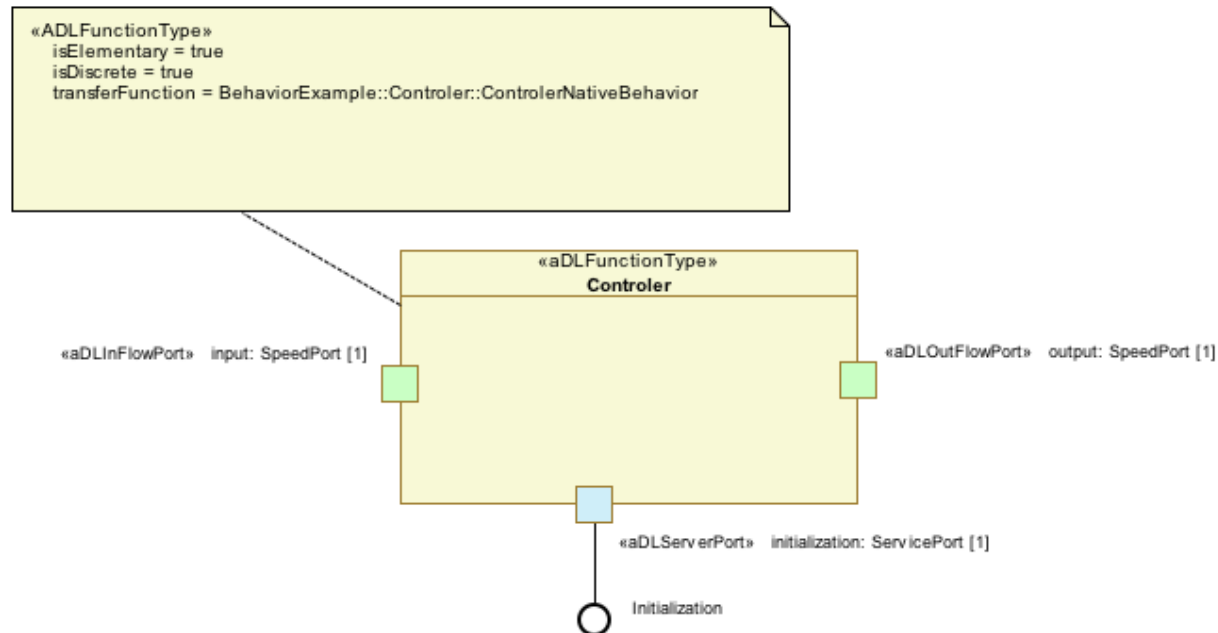
# Example 1/5



Preliminary steps: definition of some port types

- Service ports
- Flow ports

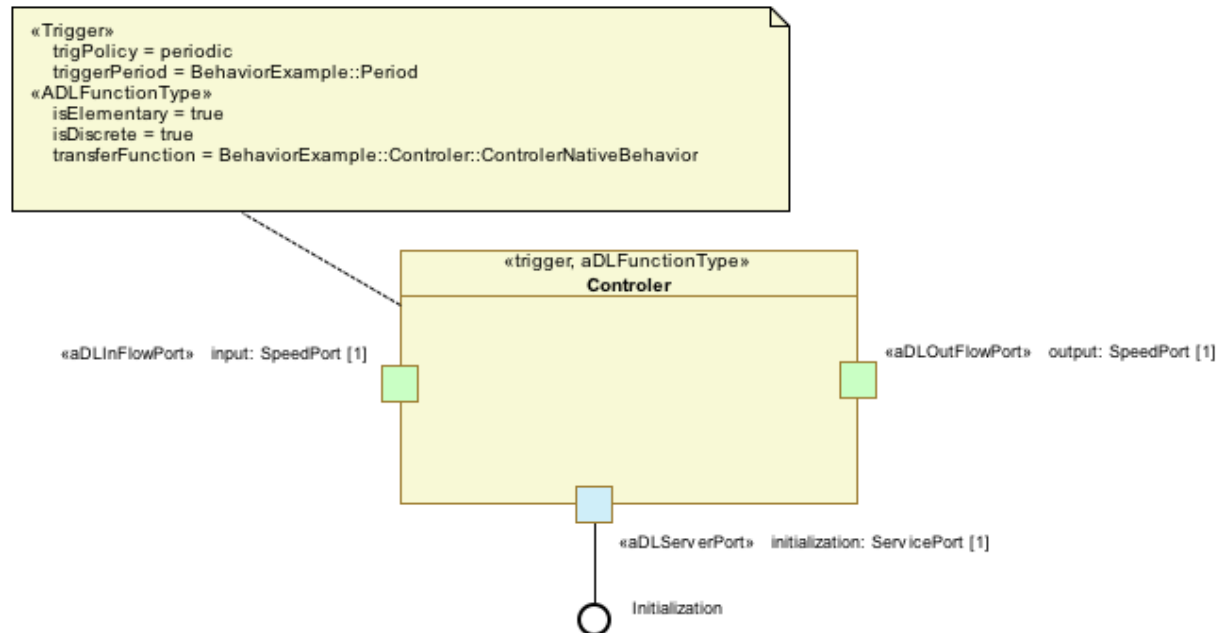
## Example 2/5



Elementary function with flow port (in/out) and server port

A behavior has been attached to the function

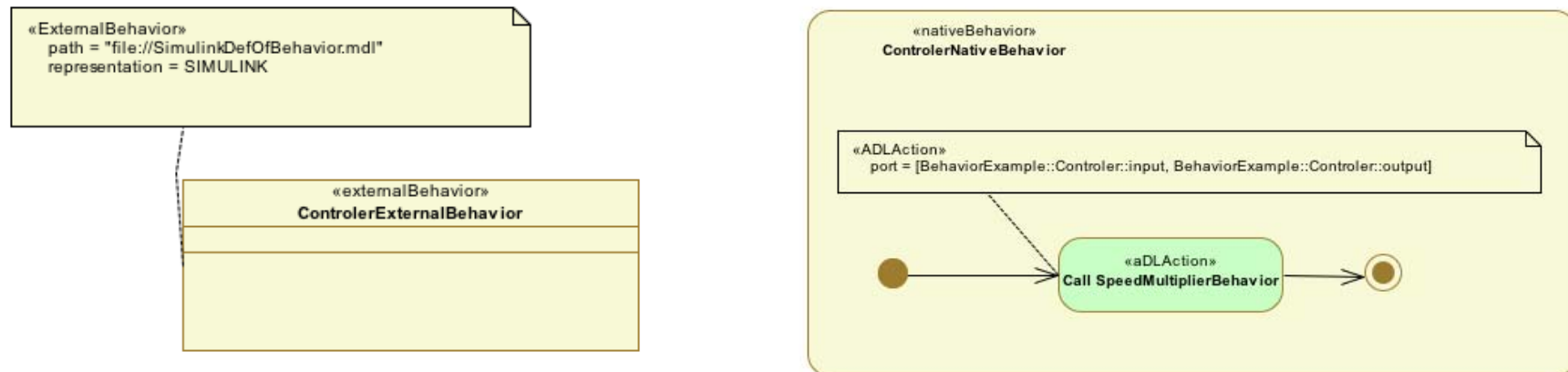
## Example 3/5



Adding triggering information to the function

The ADLInFlowPort should be tagged isTriggered=true if trigPolicy set to Event (not the case here)

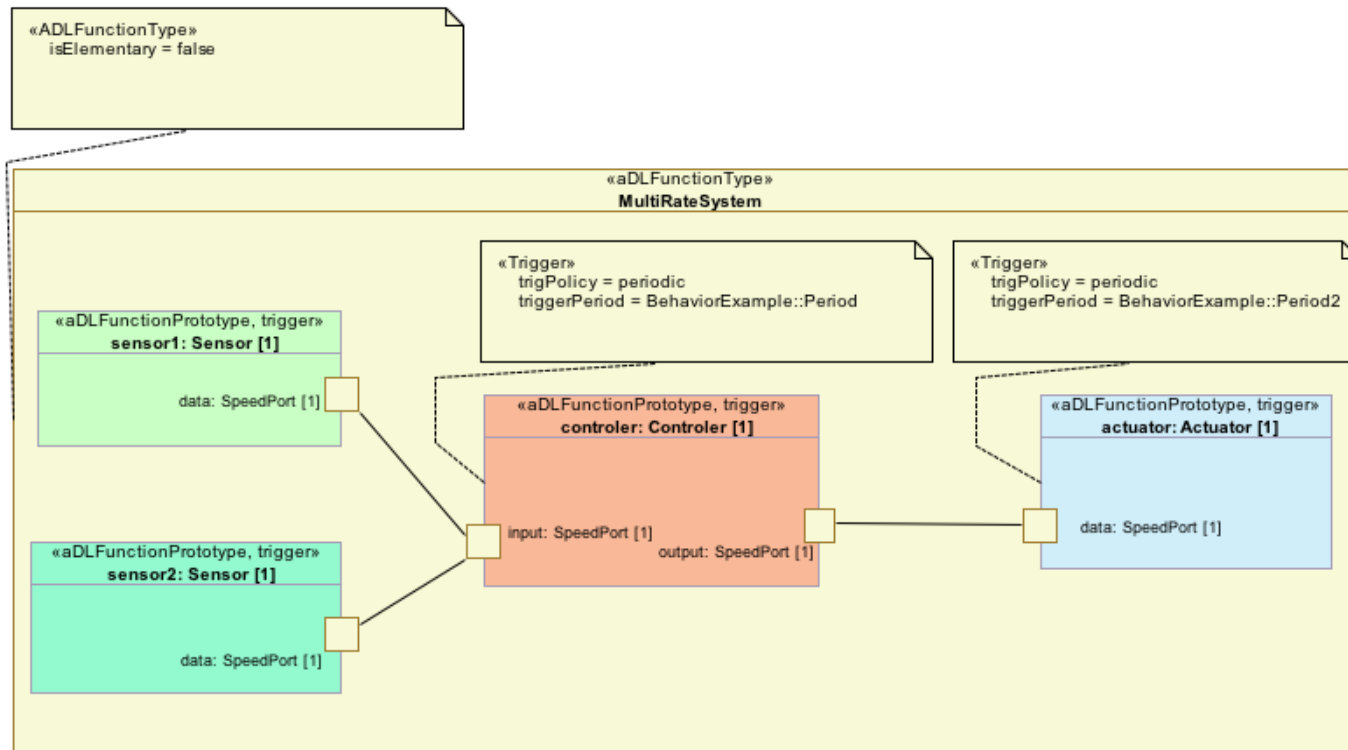
## Example 4/5



An opaque, external behavior (ref. to external file)

A native behavior defined as UML activity (use of modeling shortcuts)

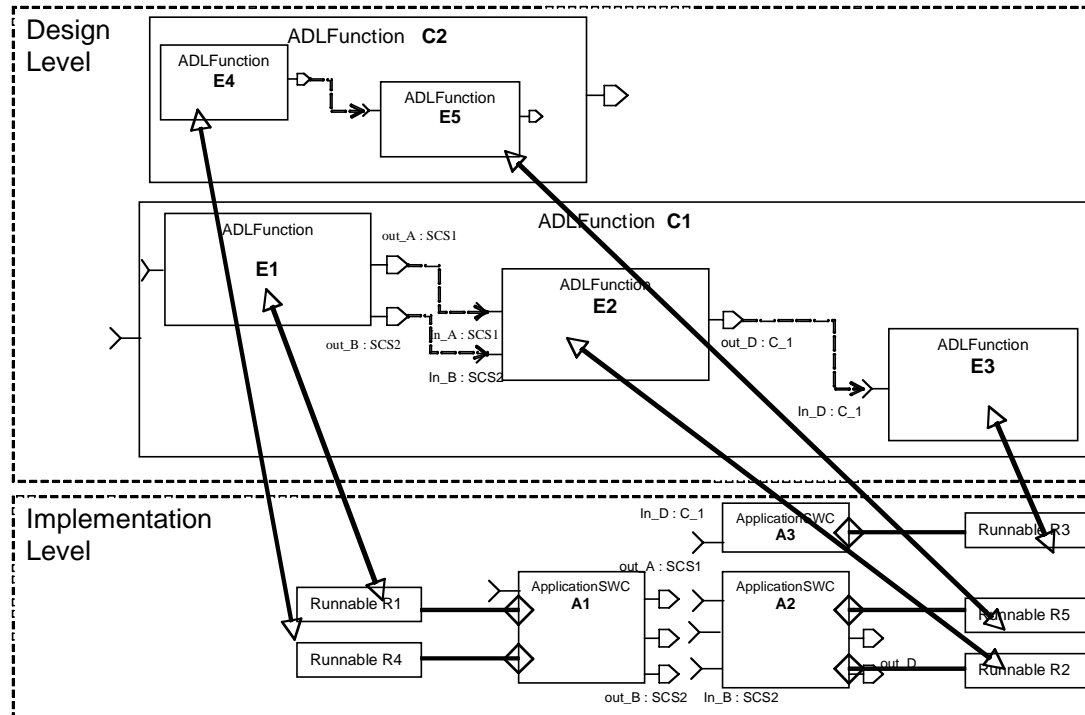
# Example 5/5



Composite function example e.g. multi-rate system

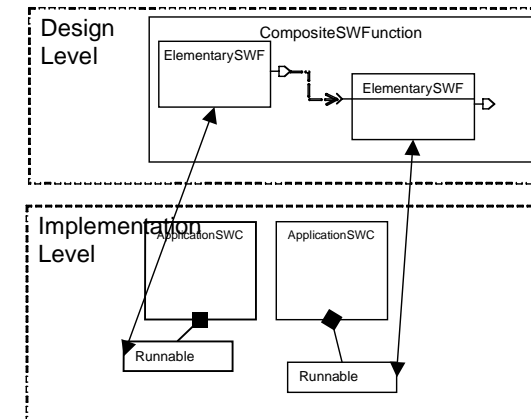
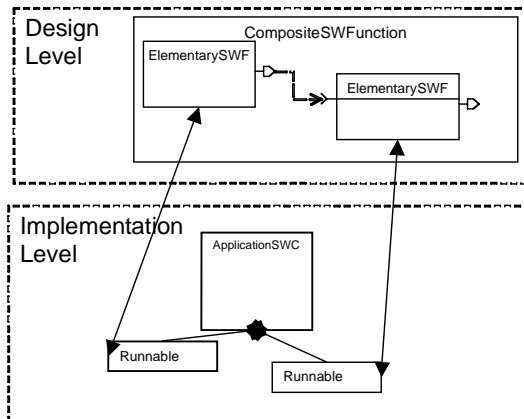
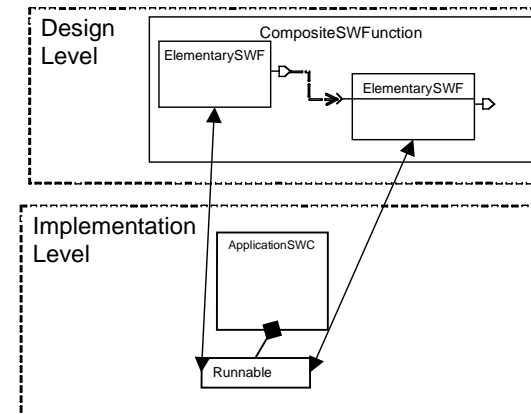
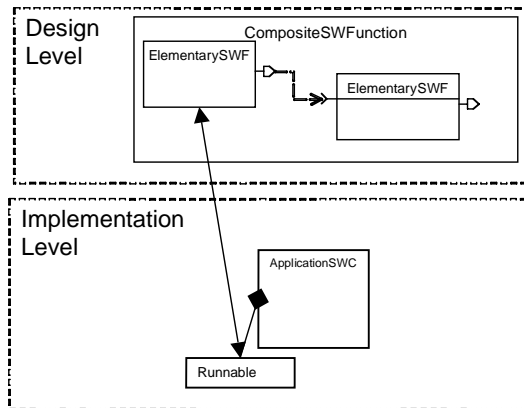
Triggering information put on the function prototypes

# Autosar projection

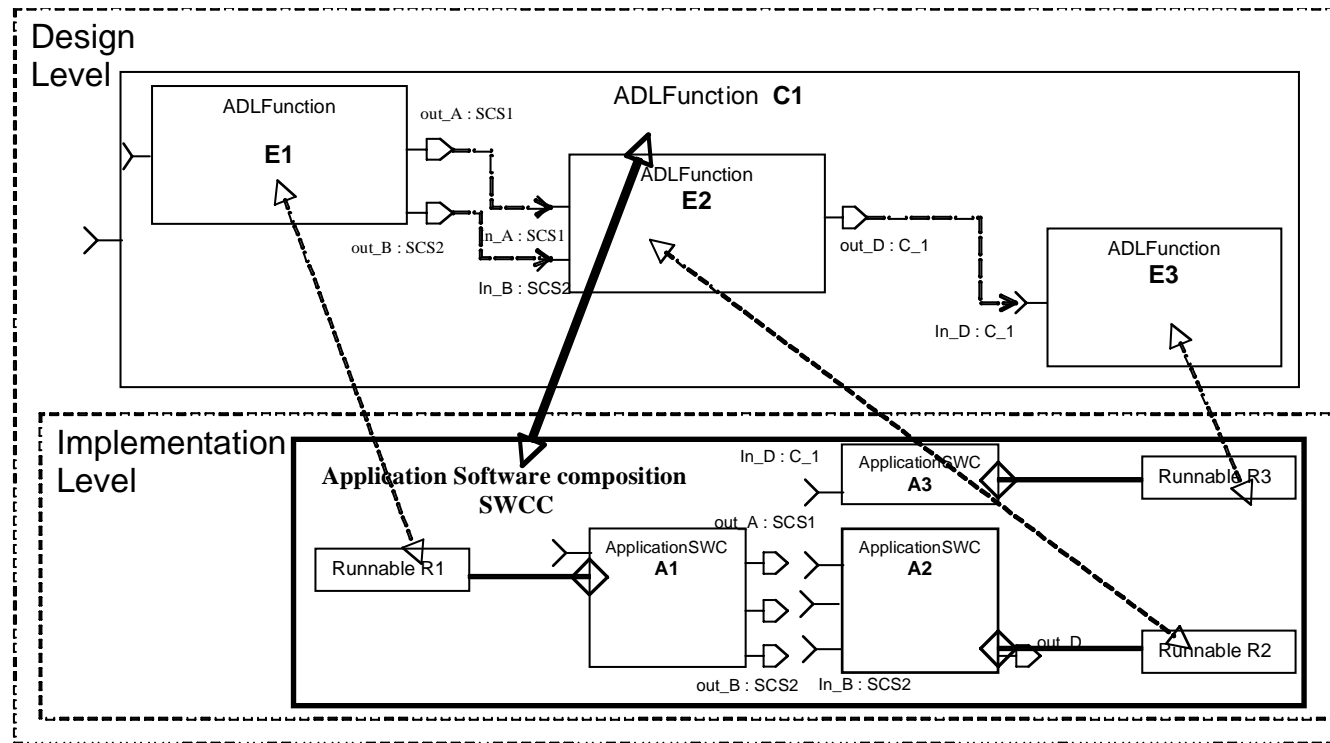


Architecture may be different because of change of focus: functional->implementation  
 Several mapping solutions can be thought of  
 ✓Functions to runnables here

# Alternative mappings



## Another solution: Composite functions vs. AR composition



Examples of the ADLRealization relation:  
 Dashed is AR runnable to ADLFunction  
 Solid is AR component or composition to ADLFunction

# Projection of Trigger modes to Autosar

RTE events	ADLFunction triggers
Timing Event for periodical execution	TrigPolicy: periodic TriggerPeriod:value Offset:none TriggerCondition:NA
DataReceivedEvent	TrigPolicy: event TriggerPeriod:value Offset:none TriggerCondition:ADLFlowPort
OperationInvokedEvent	TrigPolicy: event TriggerPeriod:value Offset:none TriggerCondition:ADLClientServer
DataSendCompleteEvent	Not Applicable (see text justification)
WaitPoint	Not Applicable (see text justification)

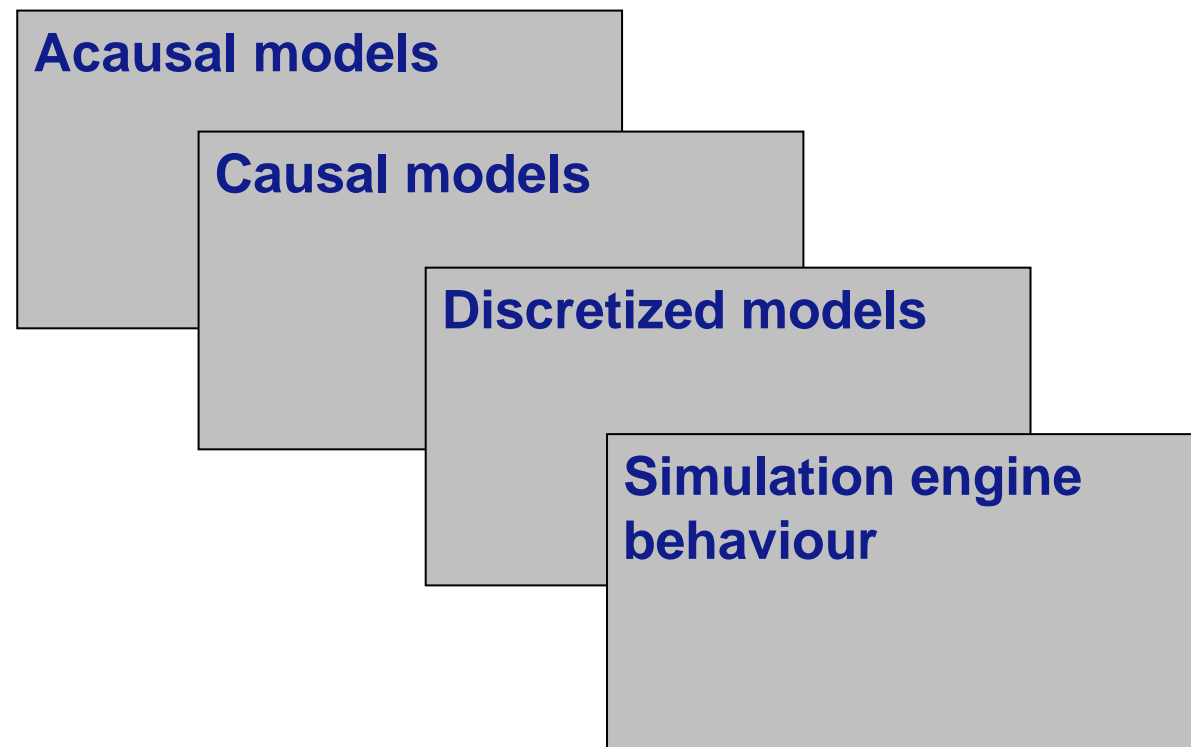
# Environmental modeling

- Environment = plant models + EE environments
  
- Here, EAST-ADL approach to plant models will be presented, i.e. continuous-time systems with which the embedded system interacts.

# Requirements

- The ADL shall support environment models in different levels of details
- The ADL shall provide HIL compatible representations of the relationships between environment model and sensors & actuators
- The ADL shall support re-use and variants of the parts of an environment model
- The ADL should support environment models for electronics/software environment, such as test equipment, road infrastructure, fellow vehicles etc.

# Different realization levels for plant models



# Acausal models

- Equation based modeling: SysML parametric diagrams, Modelica, VHDL-AMS
- Enables re-use of components, easy to connect physical components to each other.
- SysML parametric diagrams and Modelica blocks are incompatible, which was shown in a paper presented at the EOOLT workshop in 2007 by ATESSST. The Modelica community has developed its own ModelicaML profile.
- No acausal representation is in the EAST-ADL language today, ADL DesignConstraint (Physical) could be used for equations

# Causal models

- Causal models are continuous models, also referred to as signal flow graphs. Used in Simulink, ASCET, bond graphs, formalized in Ptolemy.
- In SysML, continuous activity diagrams can be modelled. In a publication we have shown that this is not a suitable representation of causal continuous systems.
- By setting the parameter "isDiscrete" to false, continuous functions can be defined in EAST-ADL.

# Simulation engine behavior

- A proof-of-concept representation of Simulink models in UML has been developed. A Simulink to UML structure conversion has been developed using the Atlas Transformation Language, ATL
- By using ATL, it is convenient to provide different mappings of Simulink models to ADL entities.
- The Simulink simulation engine, including ordering of blocks, solver, time marching, can be represented in UML, which has been shown in a publication. This is to be further investigated.

# Perspectives

- ✓ The overall behavior and environmental model constructs were established in EAST-ADL2 including providing a mapping between Simulink and UML continuous/discrete behavior models
- ✓ Investigation led to find out possible mismatches in semantics w.r.t. to UML, can be dealt with:
  - ✓ more careful definition of the semantics attached to the stereotypes
  - ✓ building blocks (library of specific actions) for the definition of native behaviors inside activity diagrams (work started for environmental modeling and Simulink import/export)
- ✓ Definite need to lead some work in close loop with simulation tools (as was done for ErrorBehavior)

See you soon for ATESST2 results!!