

# **Overview of the Error Modeling Annex for the Architecture Analysis and Design Language**

Steve.Vestal@Honeywell.com  
SAE AADL Meeting  
Tucson, AZ  
23 January 2007

# Outline

→ Context

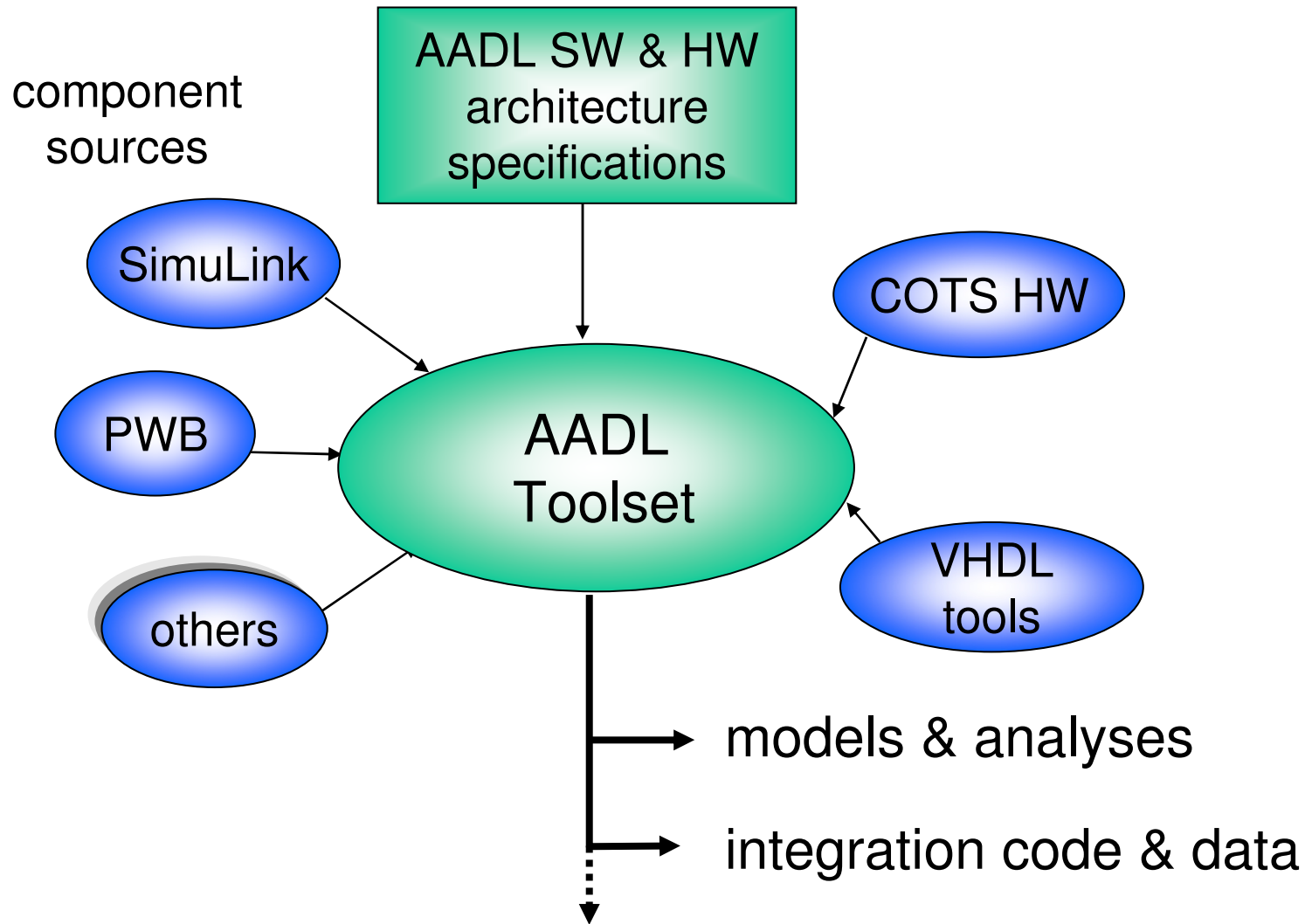
AADL  $\mu$ Overview

Component Error Models

Composing Error Models

Technology Adoption

# Automating Architecture Analysis & Integration



# Example Requirements Addressed by Error Modeling

**Reliability** is the likelihood a system will correctly perform its function over a specified interval of time.

**Integrity** is the likelihood a system will not do anything wrong (or at least unsafe) over a specified interval of time.

**Security** is the likelihood a system cannot be placed into a hazardous state by unauthorized users.

**Availability** is the likelihood a system functions reliably, securely and with integrity when it is desired.

**Maintainability** is the degree to which a system that is not operating reliability, securely, with integrity, or without adequate performance, can be repaired/upgraded to do so.

**Performability** is a measure of the expected utility of a system that may revert to degraded modes of operation in the presence of faults.

**K-fault-tolerance** means the system meets its requirements in the presence of any K faults.

# Terms, Concepts and Referenced Documents

A **fault** is a root cause for an error, e.g. a burned-out transistor in a memory cell in a memory chip.

An **error** is a persistent undesired or erroneous state or condition in a component, e.g. an incorrect word retrieved from a memory with a bad cell.

A **failure** occurs when a component is no longer able to satisfy its nominal specification, e.g. an incorrect word retrieved from a faulty memory cannot be corrected by the provided EDAC.

A fault may persist for awhile before causing an error. This interval of time is called the **fault latency**. During this time the fault is a **latent fault**. E.g., the transistor may be in a DRAM refresh circuit that only corrupts the value during a refresh cycle.

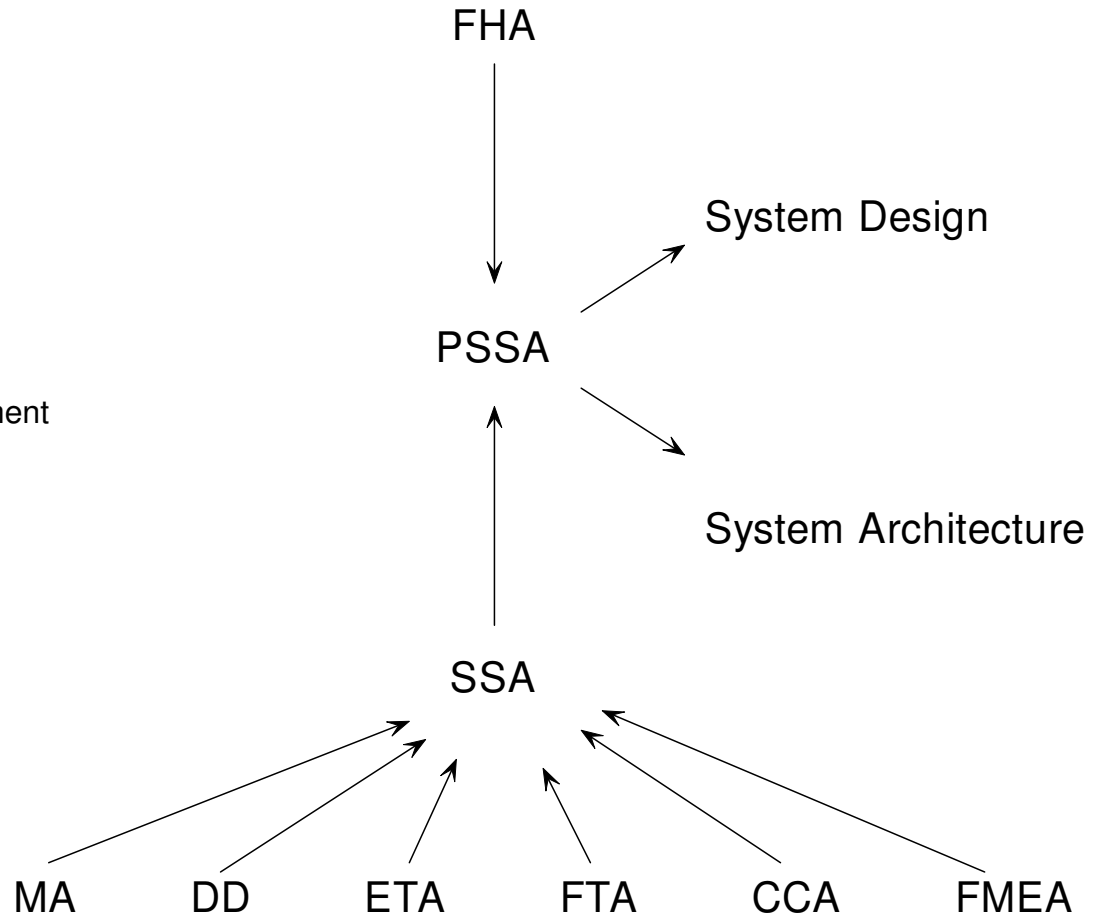
An error may persist for awhile before causing a failure. This interval of time is called the **error latency**. During this time the error is a **latent error**; e.g. the value stored in memory may have been corrupted before the read occurred.

J.-C. Laprie, editor, “Dependability: Basic Concepts and Terminology,” Dependable Computing and Fault Tolerance, volume 5, Springer-Verlag, Wien, New York, 1992.  
Prepared by IFIP Working Group 10.4 on Dependable Computing and Fault Tolerance.

# Terms, Concepts and Referenced Documents

Key:

MA - Markov Analysis  
DD - Dependency Diagrams  
ETA - Event Tree Analysis  
FTA - Fault Tree Analysis  
CCA - Common Cause Analysis  
FHA - Functional Hazard Assessment  
SSA - System Safety Assessment  
PSSA - Preliminary System Safety Assessment  
FMEA - Failure Modes &  
Effects Analysis



Society of Automotive Engineers Aerospace Division, "Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment," SAE Aerospace Recommended Practice ARP4761, Warrendale, PA, Dec 1996.

# Terms, Concepts and Referenced Documents

Radio Technical Commission for Aeronautics, “Software Considerations in Airborne Systems and Equipment Certification,” RTCA DO-178B, Washington, DC, December 1992.

Radio Technical Commission for Aeronautics, “Design Assurance Guidance for Airborne Electronic Hardware,” RTCA DO-254, Washington, DC, April 2000.

US Department of Defense, “Reliability Prediction of Electronic Equipment,” MIL-HDBK-217F, Washington, DC, 2 December 1991.

# Outline

Context

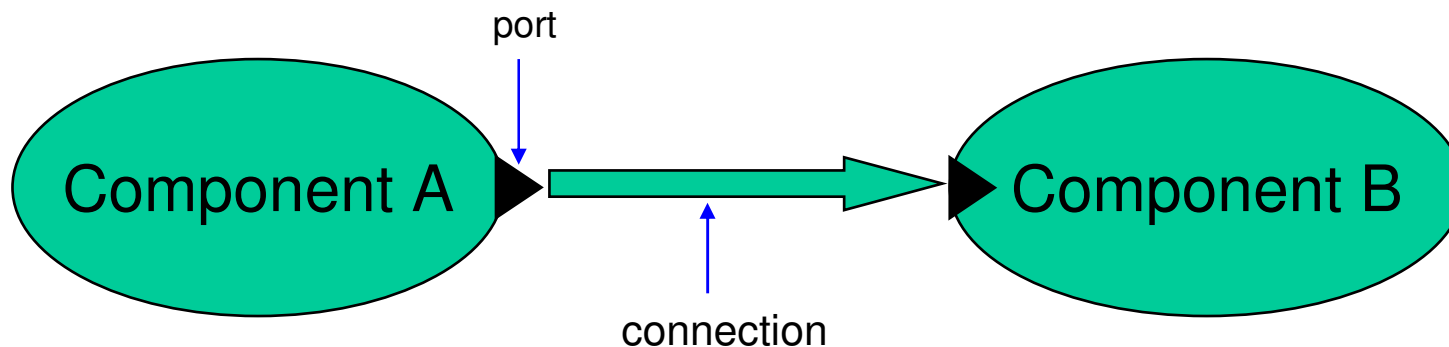
→ AADL  $\mu$ Overview

Component Error Models

Composing Error Models

Technology Adoption

# Connected Components



The AADL defines standard **categories** of **components**:

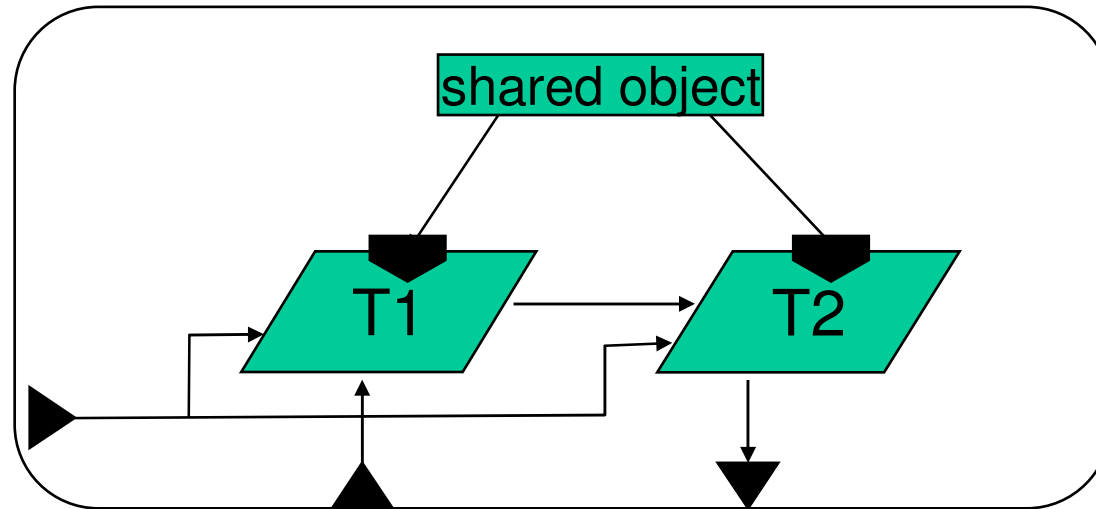
Software: data, subprogram, thread, process

Hardware: memory, bus, device, processor

Both: system

A **connection** between component **ports** declares a flow of control and/or data between components. There are different kinds of ports and connections: data, event, event data, ...

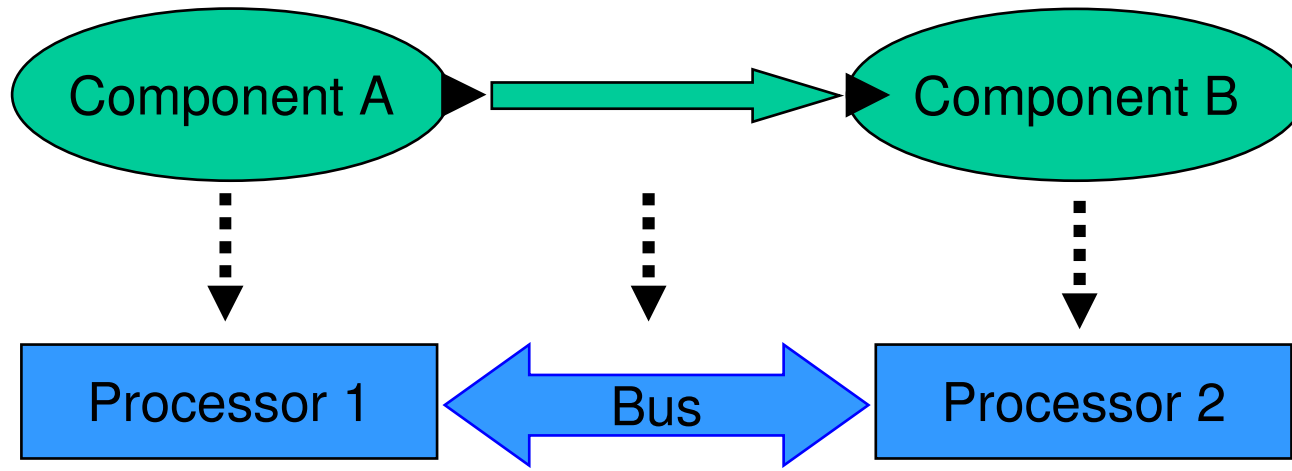
# Shared Objects



A component may declare (in its type) that it requires access to an object declared outside itself

Required accesses may create shared data objects.

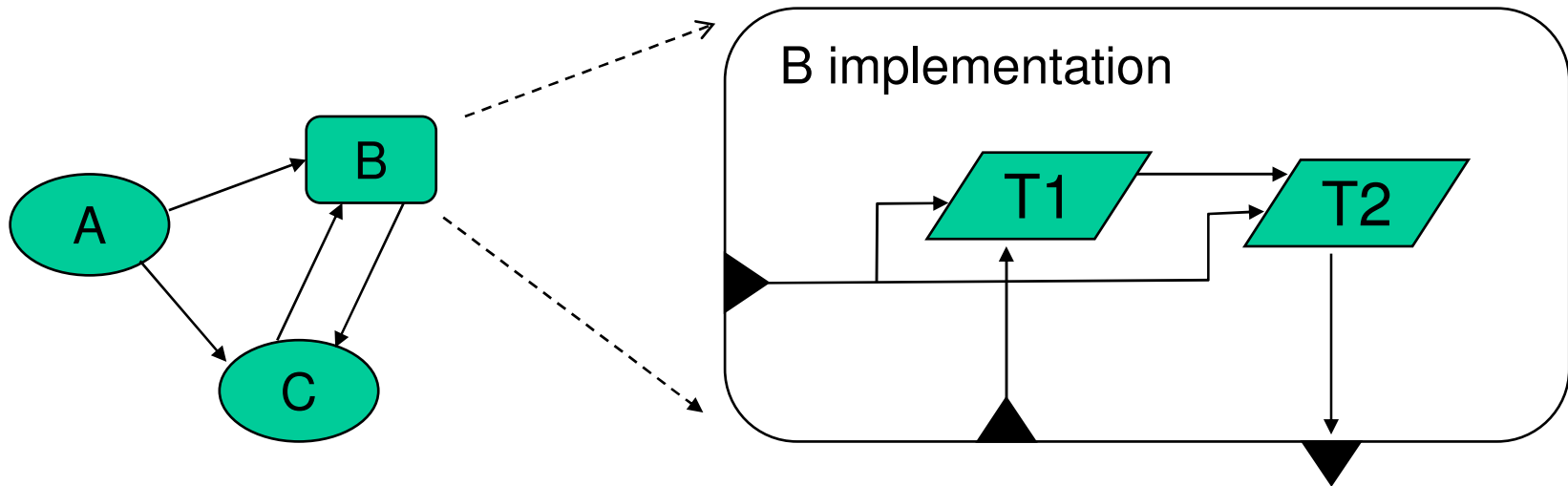
# Systems of Bound Components



A **binding** establishes another kind of control, data and error flow between software and hardware components.

**Property associations** may constrain the legal and required bindings, but bindings need not be completely and explicitly declared by the developer.

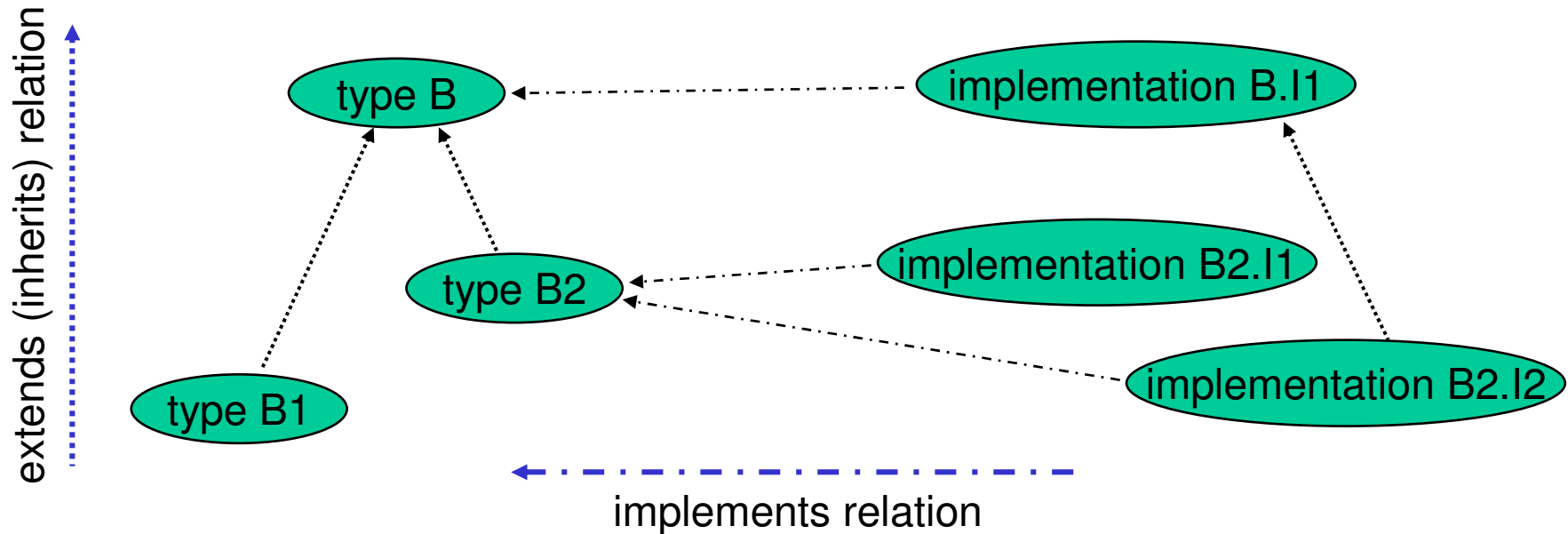
# Hierarchical Specification



A component may have an **implementation**, an internal sub-architecture declared as a set of connected sub-components.

A distinguished root system component may be decomposed into an assembly of connected sub-components, each of which may have an implementation declared as an assembly of connected sub-sub-components, ...

# Type and Implementation Hierarchies



**Types** of components may be declared: category, ports, ...

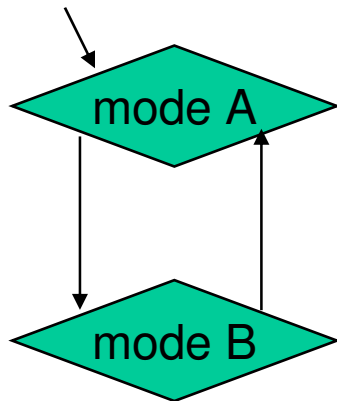
A type may be declared to **extend** (inherit from) another type.

Zero or more **implementations** may be declared for a type: sub-components, connections, ...

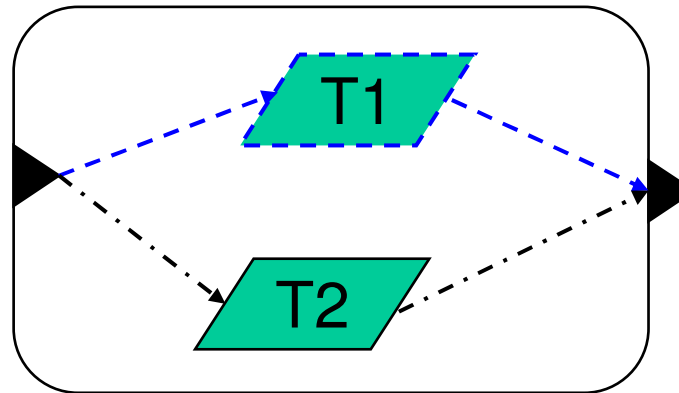
An implementation may be declared to extend another implementation.

(An implementation diagram is a set of connected type icons.)

# Dynamic Architecture Specification



mode transition diagram



implementation diagram

--- active in mode A  
- - - active in mode B

**Modes** and **mode transitions** may be made declared for a component implementation.

The pattern of connections and sub-components active at some point in time may be declared to depend on the current mode of operation.

# Outline

Context

AADL  $\mu$ Overview

→ Component Error Models

Composing Error Models

Technology Adoption

# Error Model Types

An ***error model type*** may declare

***Error states***, which may be used to model e.g.

- Hazards identified during a hazard analysis
- Failure Modes identified during a FMEA

(There must be a distinguished initial error state.)

***Error events***, which may be used to model e.g.

- Internal faults
- Internal repairs

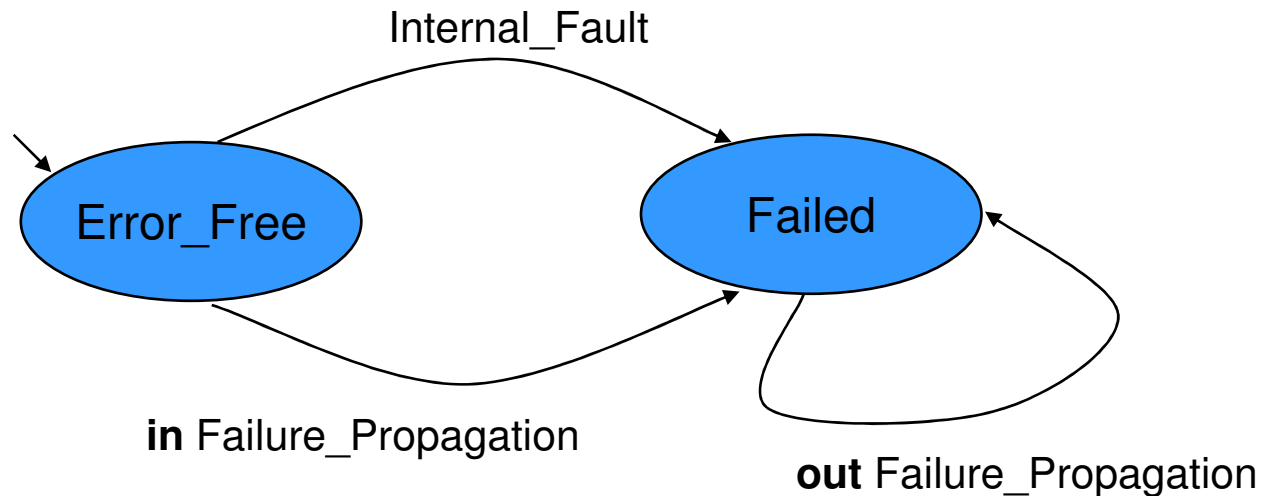
(These are made visible to permit external property associations.)

***Error propagations***, which may be used to model e.g.

- Failure effects

Version 1.0 of the annex does not support type extension.

# Error Model Implementations



**Error model implementations** declare **error state transitions**, i.e. how the error state changes due to:

- Faults and repairs occurring internally within a component
- Error propagations into a component

Transitions are also declared to show which errors are propagated out of a component depending on the current error state of that component.

# Event and Propagation Occurrence Models

An **occurrence** property association may be used to specify a stochastic model for

- error events (internal faults and repairs)
- **out** error propagations

There are two standard occurrence models:

- Fixed probability
- Poisson

Non-standard occurrence models are permitted, e.g. non-Markovian error propagations and repairs

# Component Error Models

**An AADL component implementation declaration may include an error annex section that contains declarations for**

- an error model type**
- an optional error model implementation**
- occurrence property associations**
- guards for incoming and outgoing error propagations**
- associations of error model events with core events**
- abstraction rules to relate a component error model to the composition of the sub-component error models**

# Outline

Context

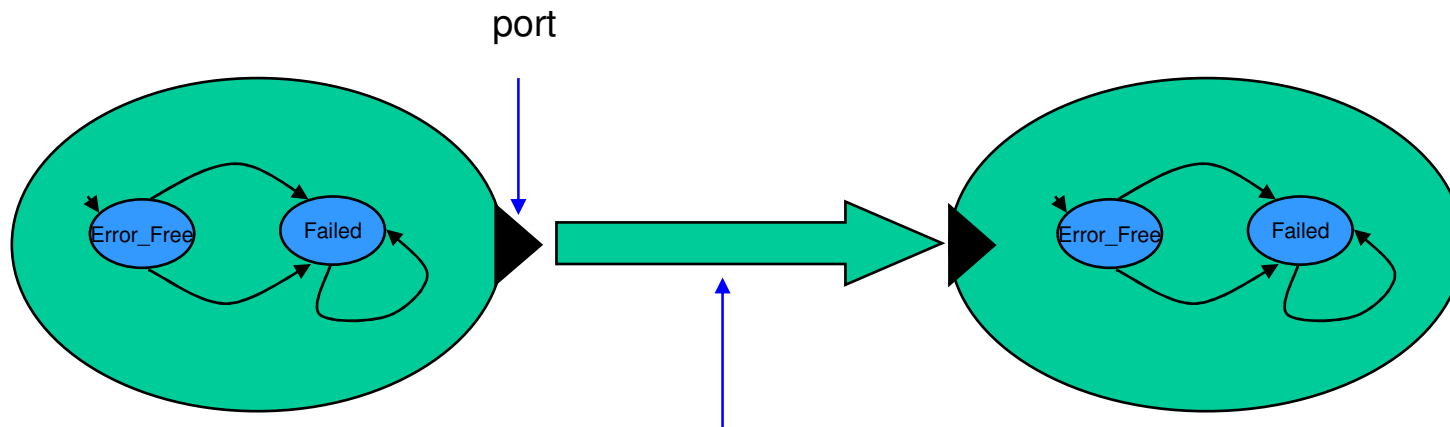
AADL  $\mu$ Overview

Component Error Models

 Composing Error Models

Technology Adoption

# Composing Component Error Models



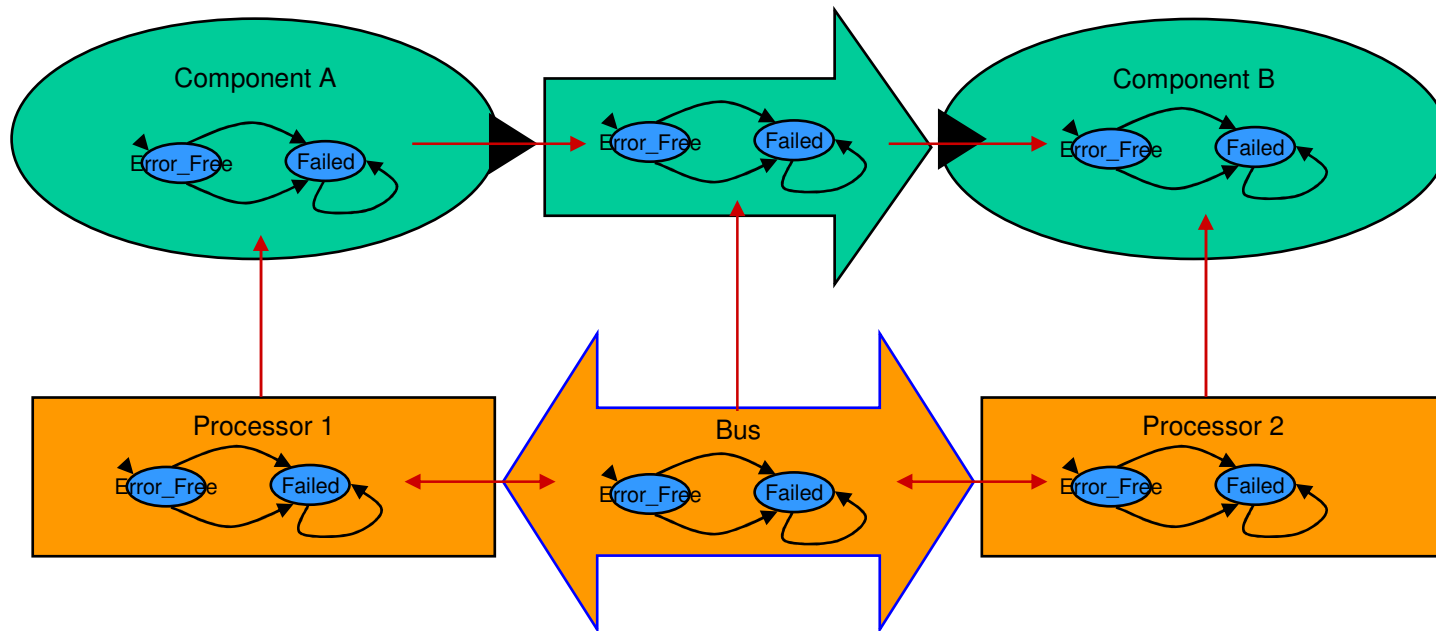
Error propagations may occur through the connection

A component error model is a stochastic automaton.

A system error model is defined as a composition of the concurrent stochastic automata of the components in that system.

Rendezvous between component error models occur when error propagations occur, as modified/guarded by property associations.

# Error Propagation Paths

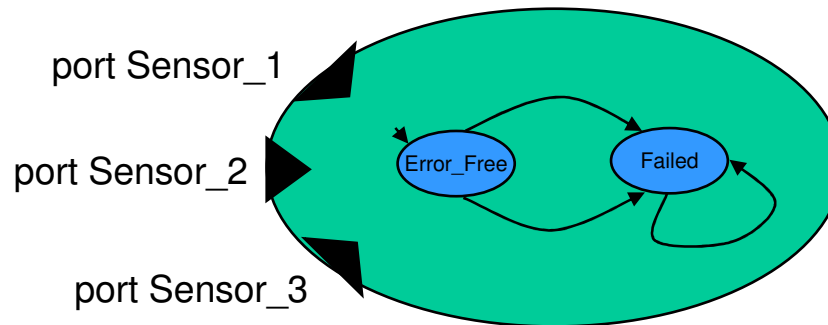


**The possible error propagations between component error models is defined by the structure of the architecture specification.**

**Permissions are given to tailor these definitions depending on error isolation features of the final as-built system, e.g.**

- electrical isolation
- time and space partitioning

# Voting Protocols



Guard\_In => **mask when 2 or more** (Sensor\_1, Sensor\_2, Sensor\_3)  
**applies to** Sensor\_1;

A **Guard\_In** error property allows error propagations arriving through a port or shared object to be either

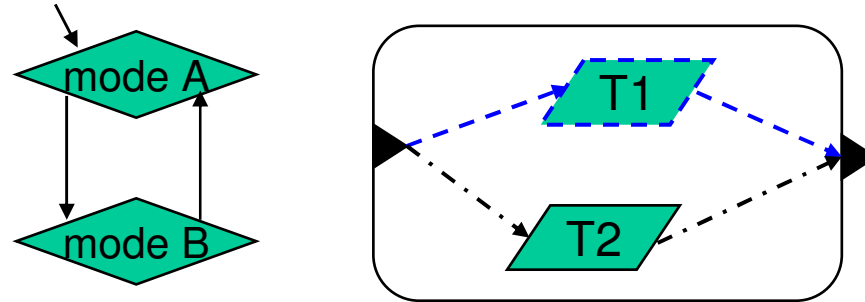
- masked
- mapped to a different kind of error
- propagated unchanged

as a function of the error states of other components that are either

- specified directly as a named error state
- inferable from a named error propagation

A **Guard\_Out** error property allows a similar masking or mapping for errors about to be propagated out of a component (see errata)

# Modal System Error Models



Connections and bindings may be mode-dependent.

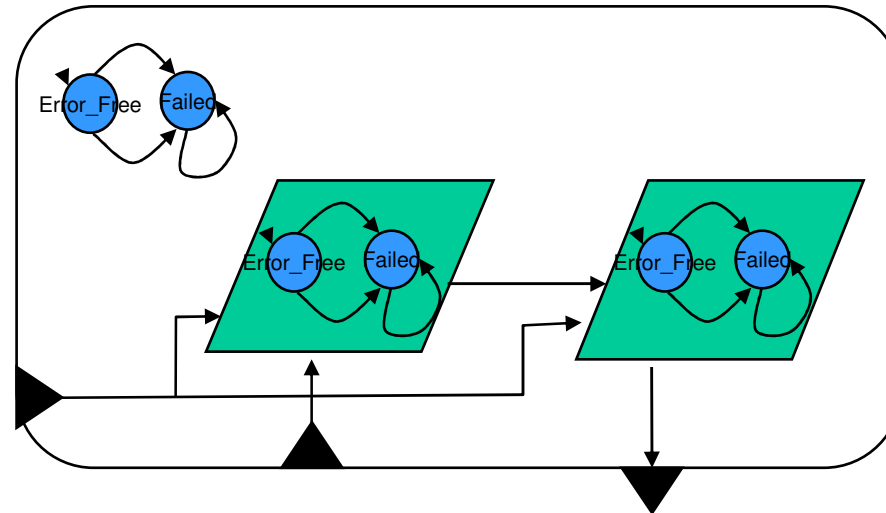
The composition of the component mode transition automata may need to be included as part of the composed system error model.

A **Guard\_Event** error property may specify that certain patterns of error states and propagations will be detected and cause an AADL core event to be raised, e.g. to trigger a mode transition.

A **Guard\_Transition** error property may specify voting protocols for events that trigger mode transitions to specify patterns of erroneous events that will be either

- masked
- cause erroneous mode transitions

# Model Abstraction Hierarchy



If a component and its sub-components have error models, the **Model\_Hierarchy** error property determines whether

- The error state of the component is derived as a specified function of the visible and inferred error states of its sub-components, connections, ports and shared objects.
- The component error model is to be used in the composed system error model in place of the composition of its sub-components, which are now abstracted away and omitted from the system error model.

A **Report** property specifies error states to be evaluated during analysis.

# Outline

Context

AADL  $\mu$ Overview

Component Error Models

Composing Error Models

 Technology Adoption

# Standardization and Tool Support

SAE AS55065/1 containing the Error Model Annex was issued in June 2006.

OSATE/TOPCASED includes support for checking and parsing error model annex specifications.

Several plug-ins are in various stages of development and maturity, e.g.

- SHARPE model generator (Embry Riddle)

- Fault Tree model generator (Honeywell Labs)

- Stochastic Petri Net model generator (LAAS)

# Potential Benefits and Barriers

## Potential benefits:

- Enable quantitative trade-offs and evaluations early in the architecture design process
- Better traceability and assurance of consistency between the design architecture and safety models
- Reduced recurring hardware cost due to more optimized IMA architectures

## Potential barriers:

- Maturity and experience with the methodology
- Maturity of supporting tools
- Multi-organization change in process from models hand-generated by a safety group to models automatically generated from a shared design architecture spec
- Need conservative, assured changes to safety processes