AEROSPACE
Assuring Space Mission Success

Application of the Architectural Analysis and Design Language (AADL) for Quantitative System Reliability and Availability Modeling

Myron Hecht, Alex Lam, and Chris Vogl

Presented to

International Symposium on Software Reliabiltiy Engineering

Mysuru, India

November, 2009

Outline

- Motivation
- Introducing AADL
- AADL Error Annex
- AADL Modeling Environment
- AADL transformation tool
- Sample Application
- Future Plans

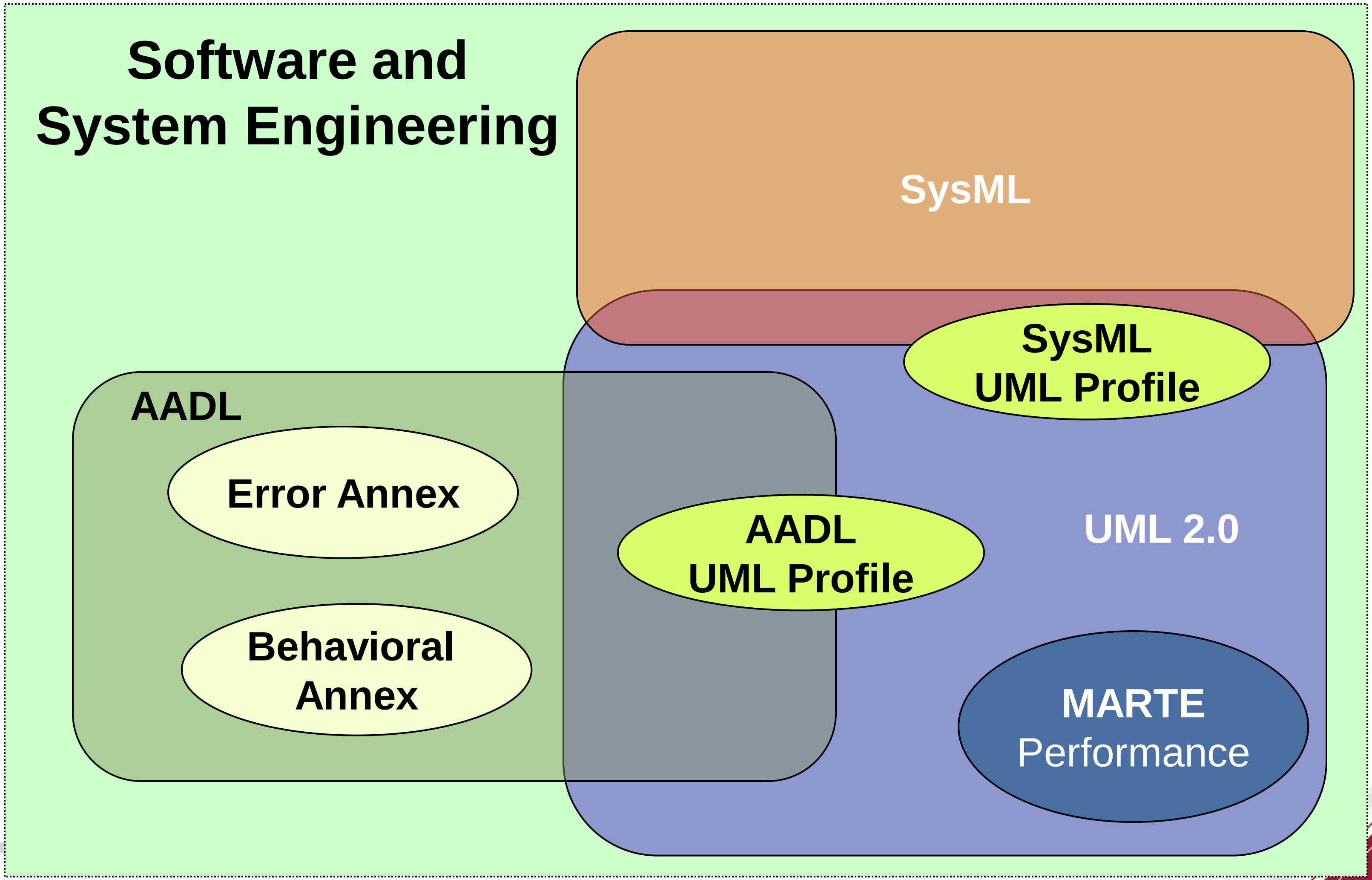
Motivation

- Need: Support to make better decisions on system architectures
- Target systems: Space vehicle and other constrained computing environments
- Development phase: Architectural decisions made during the early design impact
- Decisions supported:
  - *Extent and type of redundancy*
  - *Tradeoffs of reliability vs. Weight, power, and functional capability*
  - *Failure rate and recovery time requirements*
  - *Strategies for recovering from control and payload computing disruptions*
  - *Handling failure propagation and common mode failures*

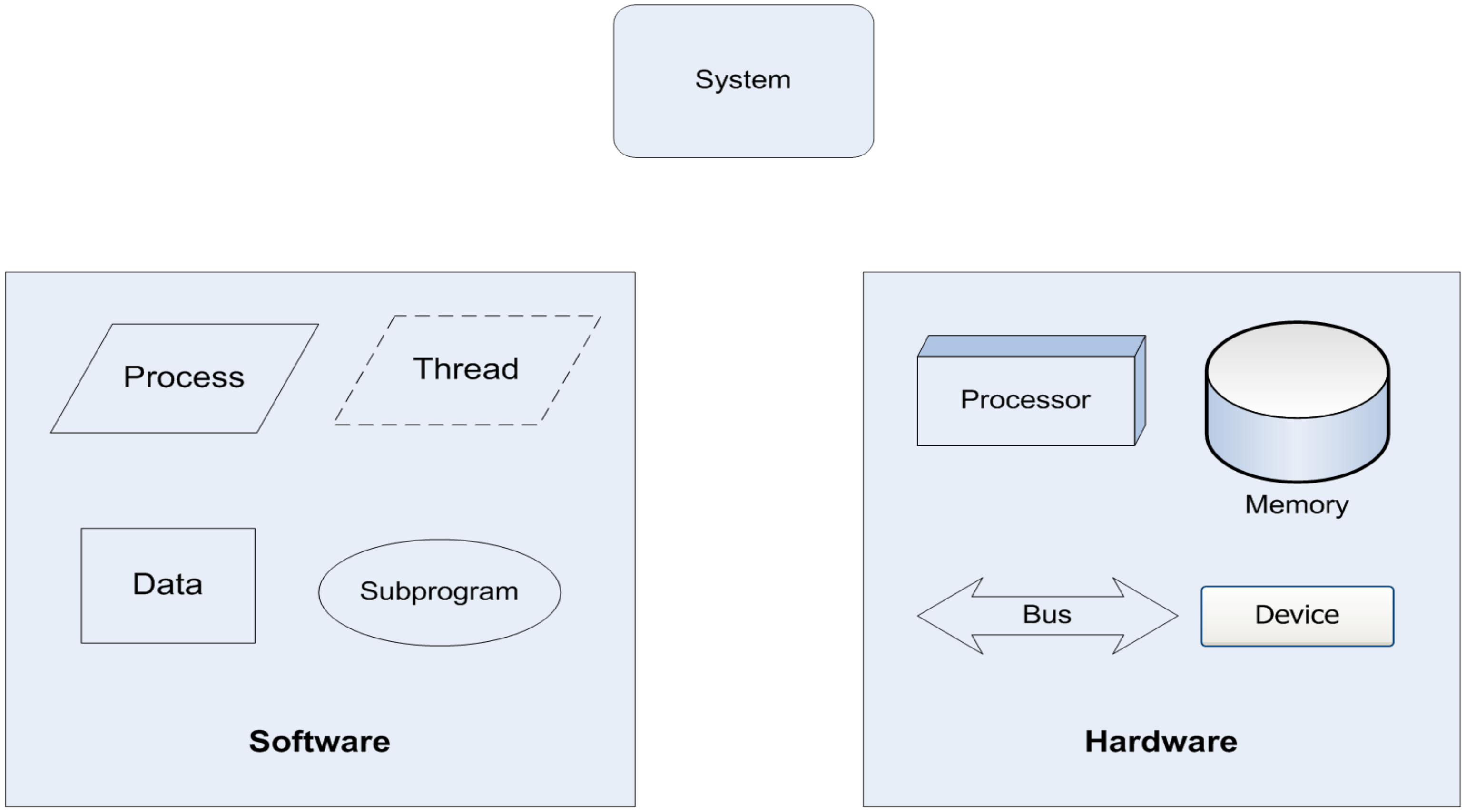
Introducing the Architecture Analysis & Design Language (AADL)

- Society of Automotive Engineers (SAE) Aerospace Standard AS5506 (2004)
  - *Preceded by more than a decade of development under the DARPA Meta-H program*
- Includes representations of software, computational hardware, and system components for
  - *specifying and analyzing real-time embedded systems,*
  - *mapping of software onto computational hardware elements.*
- Effective for model-based analysis and specification
  - *Evolved from DARPA Meta H project*
  - *Highly structured, defined semantics allows for modeling and analysis*
- Annex libraries define extensions to the core language concepts and syntax
  - *Error Annex of particular interest*

AADL/UML/SysML Relationship

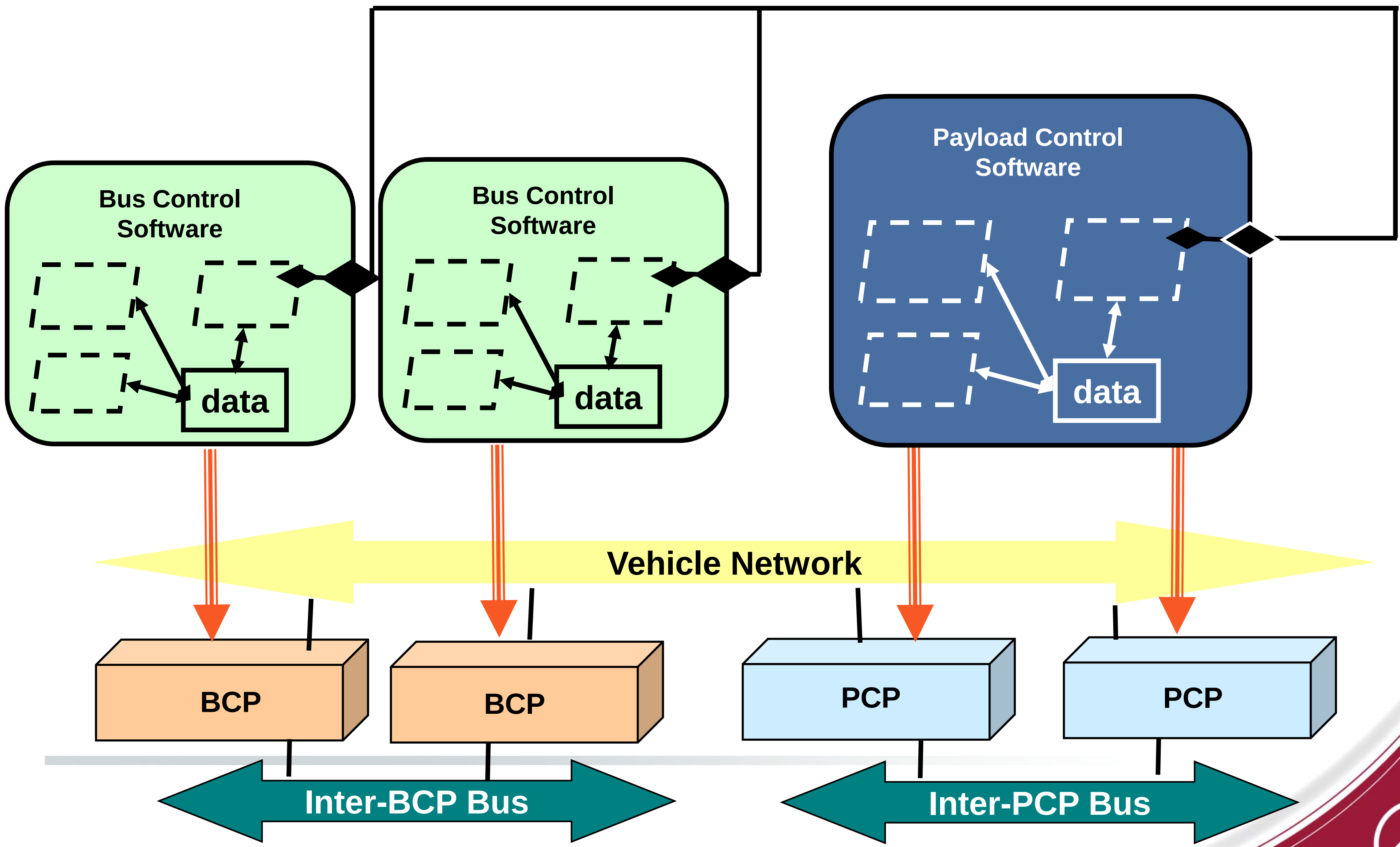


AADL Components (graphical representation)



– text and xml representations also defined

AADL Hardware/Software Architecture Representation



AADL Major Features

- Provides a standardized textual and graphical notation for describing software and hardware system architectures and their functional interfaces
- Components have interactions, flows, and subcomponents
- Component interactions: Consists of directional flow through
  - *data ports for unqueued state data*
  - *event data ports for queued message data*
  - *event ports for asynchronous events*
  - *synchronous subprogram calls*
  - *explicit access to data components*
- Different system configurations and topologies can be represented using the AADL mode concept

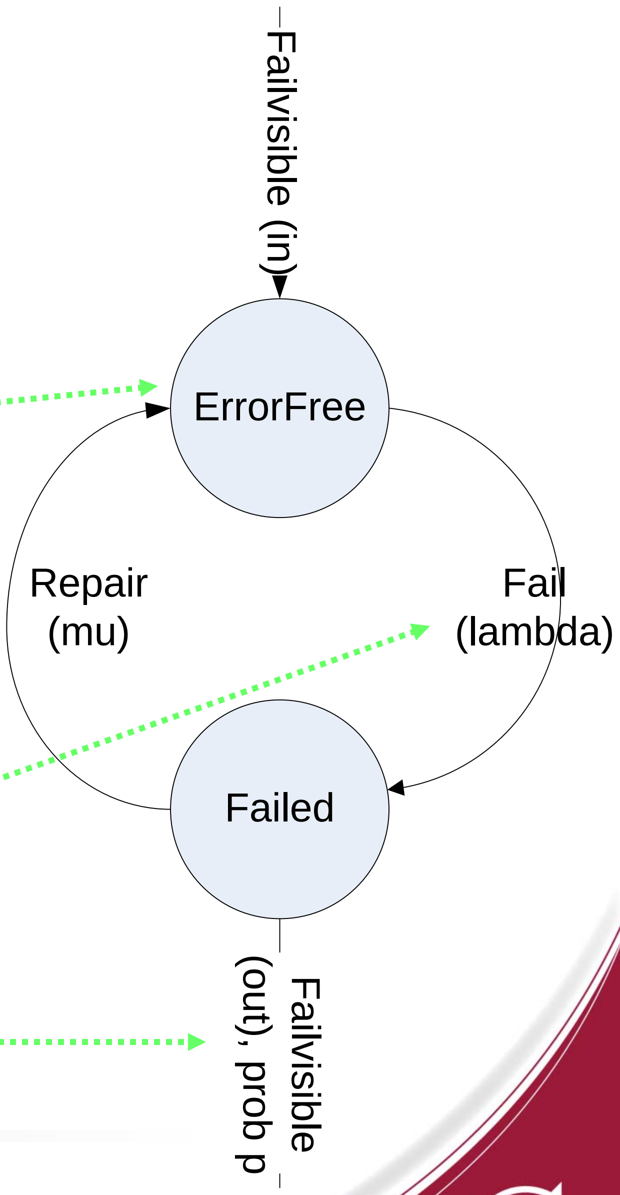
AADL Error Annex

- AADL annex that supports reliability analysis
- Defines error model
  - *State transition diagram that represents normal and failed states*
  - *Error models can be associated with hardware components, software components, connections, and "system" (composite) components*
- Error model consists of
  - *State definitions*
  - *Propagations from and to other components*
  - *Probability distribution and parameter definitions*
  - *Allowed state transitions and probabilities*

AADL Error Model Example

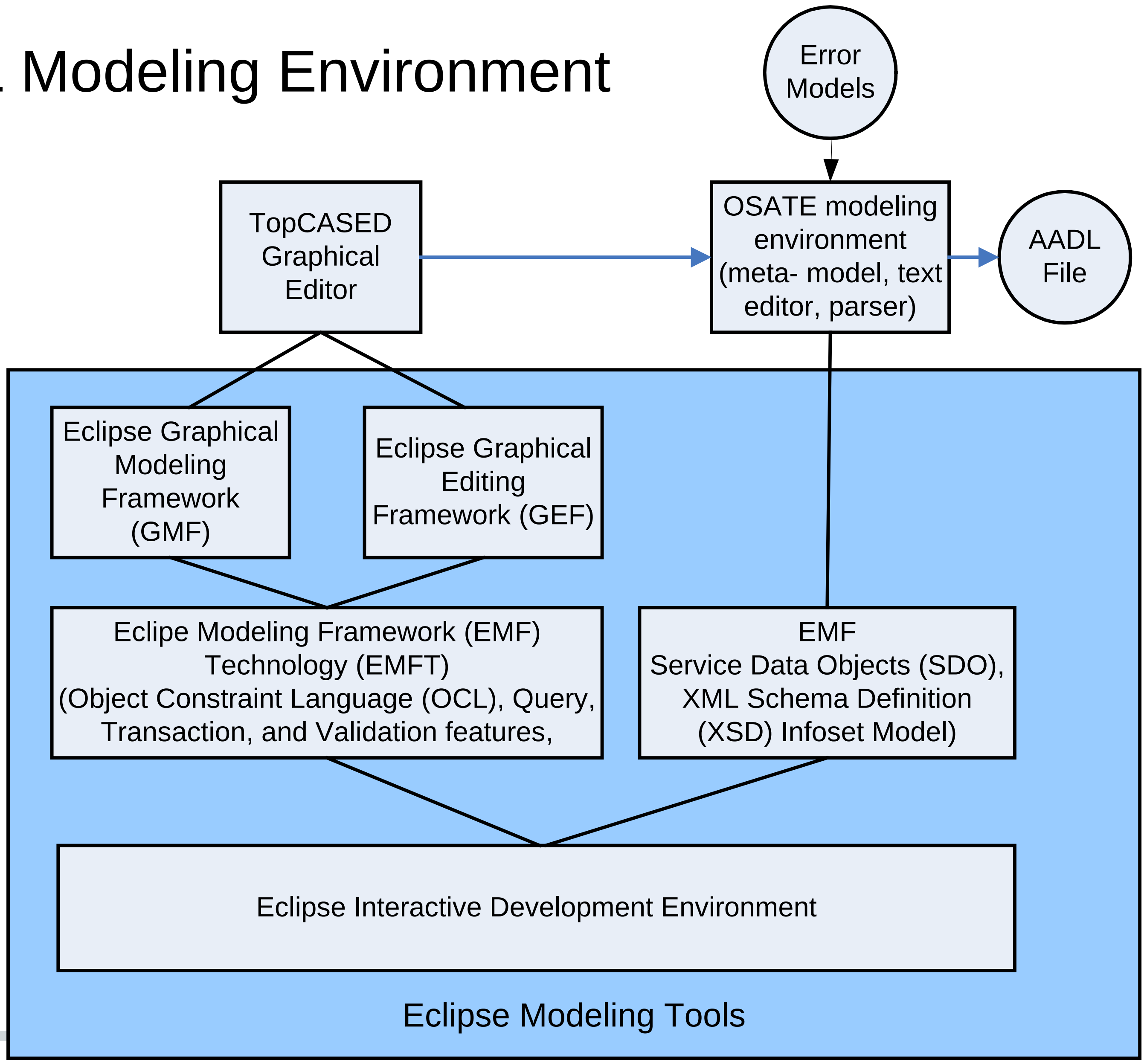
```
error model example
features
ErrorFree: initial error state;
Failed: error state;
Fail: error event {Occurrence => poisson lambda};
Repair: error event {Occurrence => poisson mu};
Failvisible: in out error propagation {Occurrence => fixed p};
end example;
```

```
error model implementation example.general
transitions
ErrorFree-[Fail]->Failed;
Failed-[Repair]->ErrorFree;
ErrorFree-[in Failvisible]->Failed;
Failed-[out Failvisible]->Failed;
end example.general;
```

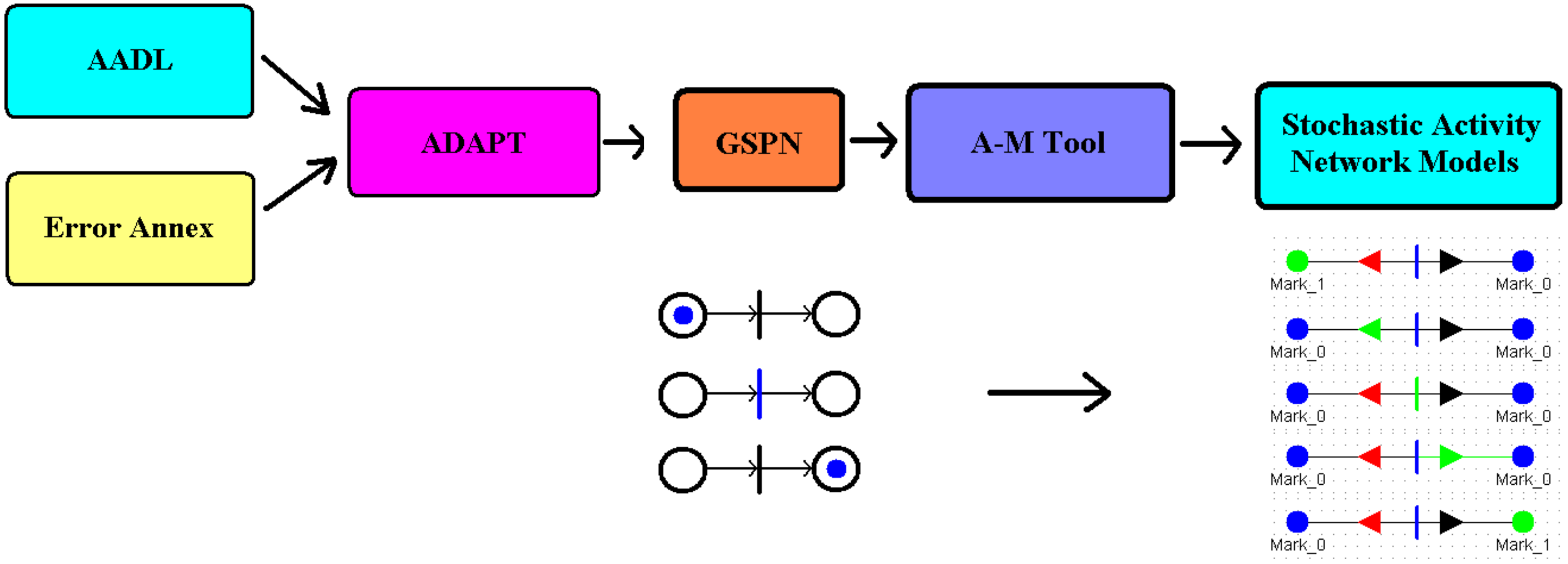


More information: Feiler (2007)

AADL Modeling Environment


Error Models
TopCASED Graphical Editor
OSATE modeling environment (meta- model, text editor, parser)
AADL File
Eclipse Graphical Modeling Framework (GMF)
Eclipse Graphical Editing Framework (GEF)
Eclipe Modeling Framework (EMF) Technology (EMFT) (Object Constraint Language (OCL), Query, Transaction, and Validation features,
EMF Service Data Objects (SDO), XML Schema Definition (XSD) Infoset Model)
Eclipse Interactive Development Environment
Eclipse Modeling Tools

AADL transformation



- ADAPT Tool (Ana Rugina, LAAS-CNRS)
  - *Packaged as an eclipse plug-in*
  - *Takes in AADL architecture and error behavior information*
  - *Converts to a general stochastic petri net*
  - *Outputs GSPN information to an XML file*
- ADAPT-MOBIUS Converter
  - *Takes in the ADAPT XML file.*
  - *Converts a GSPN to a Mobius Stochastic Activity Network*
  - *Outputs SAN information to an XML format.*

Open Source AADL Tool Environment (OSATE)

- Based on Eclipse Release 3
- Parsing & semantic checking of approved AADL
- AADL (Text) to AAXL (XML) and back
- Syntax-sensitive text editor
- Syntax-Sensitive AADL Object Editor
- AADL property viewer
- AADL to MetaH translator
- Online help
- Graphical layout editor
- Multi-file XML support
- First analysis plug-ins

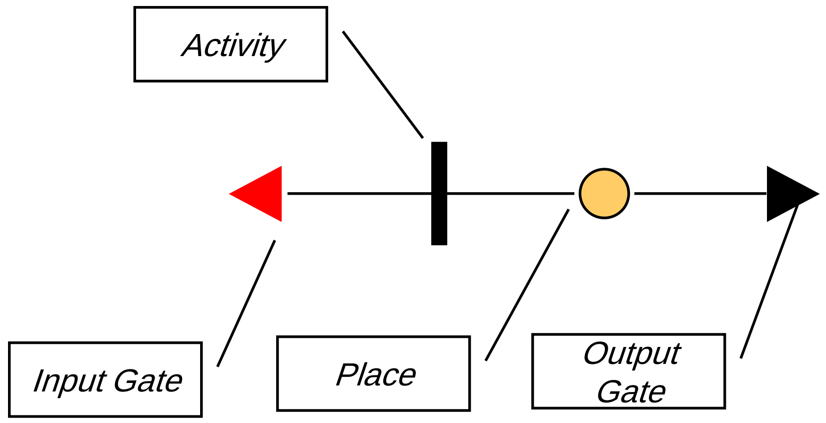
More information: www.osate.org

MOBIUS Modeling Tool

- Developed by the University of Illinois at Urbana Champaign
- System-level performance and dependability modeling
- Based on stochastic analysis network representation

More information: www.mobius.uiuc.edu

More on Stochastic Activity Networks



- *Can handle recoverable redundant systems with multiple states*
- *Adaptation of Petri Nets*
- *Evaluated using Mobius Software*
  - Developed by University of Illinois at Urbana-Champaign
- *Advantages:*
  - Simulation-based solution; does not require assumption of exponential distribution
  - Allows for higher fidelity modeling of spacecraft recovery models
- *Possible alternatives:*
  - Stochastic Petri Nets
  - Markov models

TOPCASED Graphical Editor

- Open-Source “Meta-modeling” toolset
  - A model type or a language can be described using a meta model or a meta language
  - The Open Modeling Group (OMG) defined a 4-layer model
    - *M3-Meta modeling language (ECORE – modification of MOF used by TOPCASED)*
    - *M2-Meta-model (schemas, AADL definition)*
    - *M1 – Model (AADL, finite state machine)*
    - *M0 – Real Object*
  - AADL representation in ECORE
  - TOPCASED can graphically represent ECORE

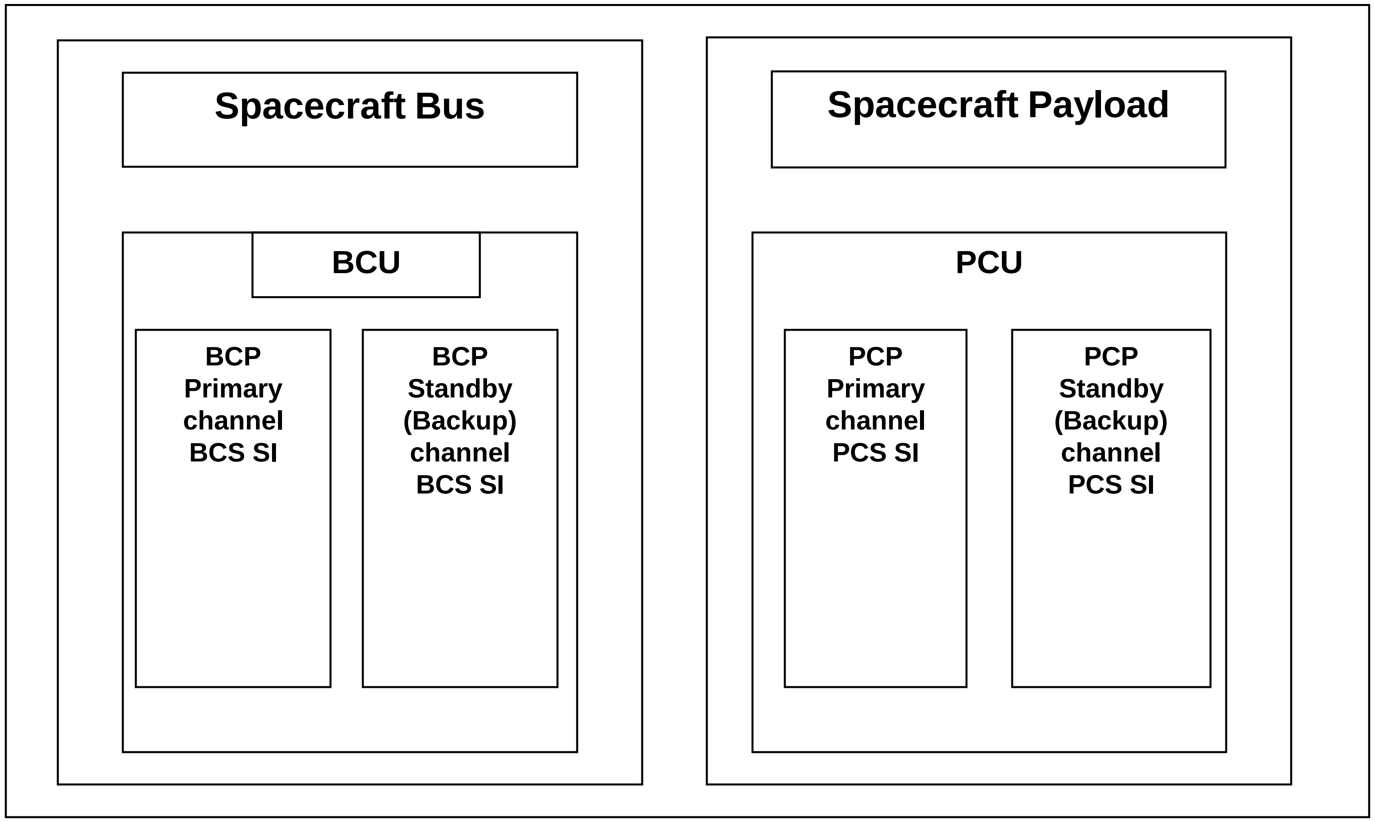
More information: www.topcased.org

Example: Low Earth Orbit Space System (LSS)

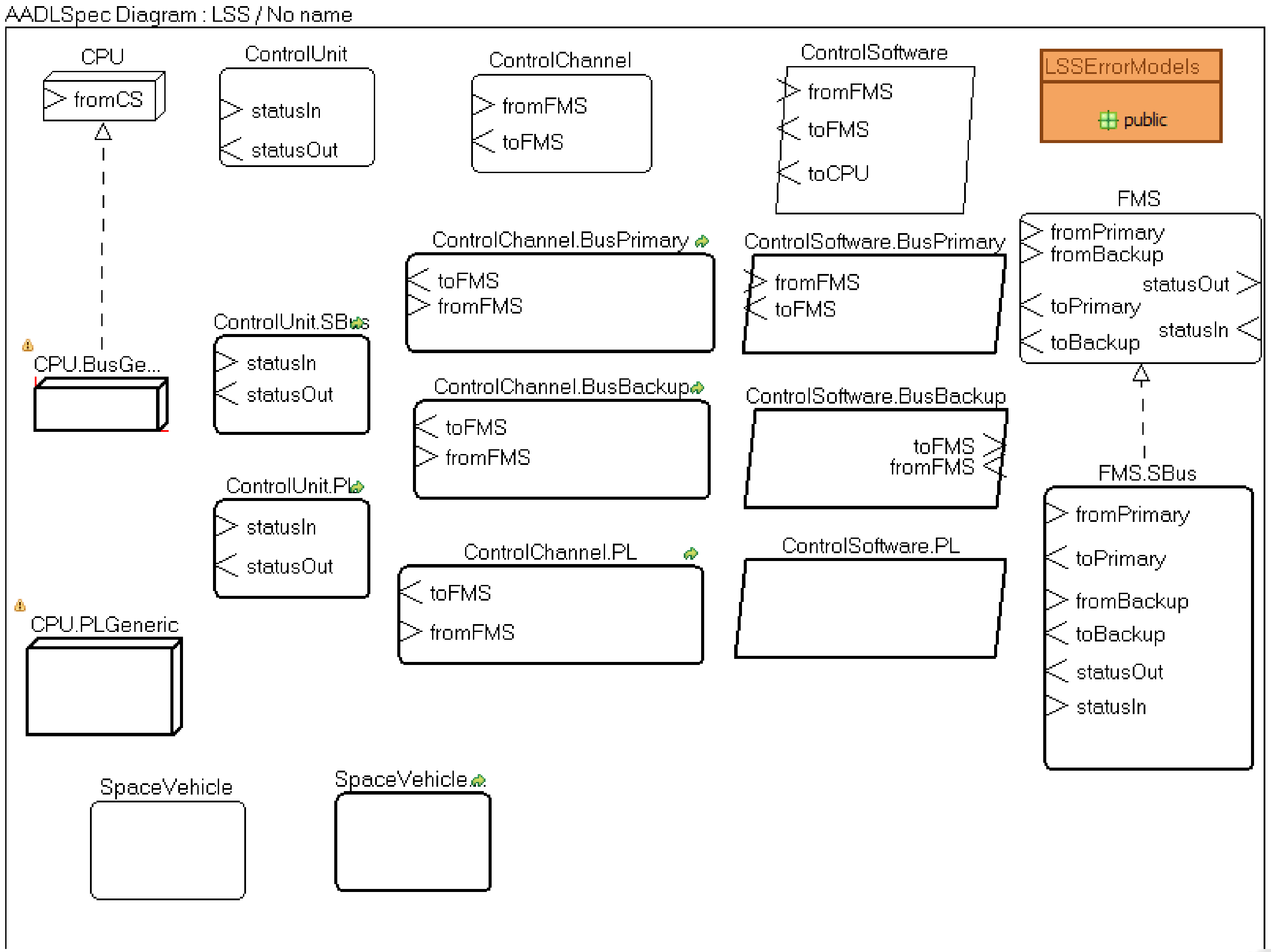
- Description of System
  - *The system contains one Bus Control Unit (BCU) and on Payload Control Unit (PCU).*
  - *Each Control Unit contains two Control Channels comprised of one piece of Control Software (BCS, PCS) and one Control Processor (BCP, PCP).*
  - *The BCU is in hot backup with imperfect switching assumed.*
  - *The PCU is in cold backup with perfect switching assumed (thus it is modeled as having only one Control Channel).*
  - *The Payload relies on the Bus, thus whenever the Bus is in Standby, the Payload goes to Standby.*

System Representation (1/2)

Static Architecture

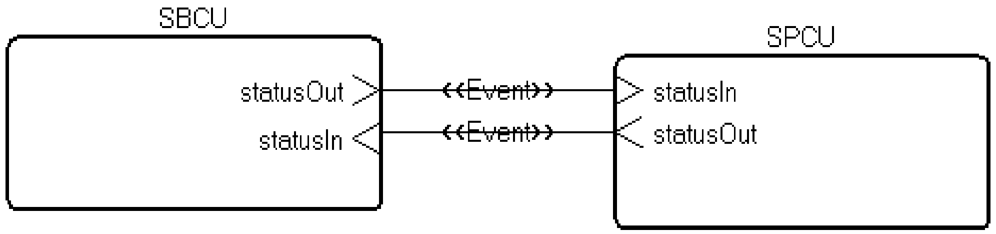

Spacecraft Bus
BCU
BCP Primary channel BCS SI
BCP Standby (Backup) channel BCS SI
Spacecraft Payload
PCU
PCP Primary channel PCS SI
PCP Standby (Backup) channel PCS SI

AADL Type Representations (using TOPCASED)

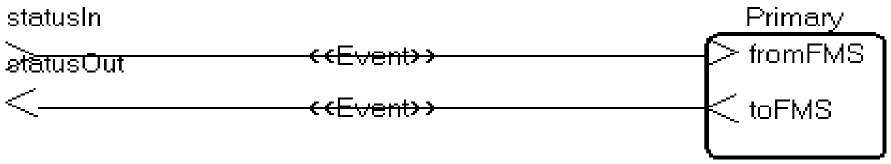


AADL Architecture Representations (using TOPCASED)

Space Vehicle Diagram

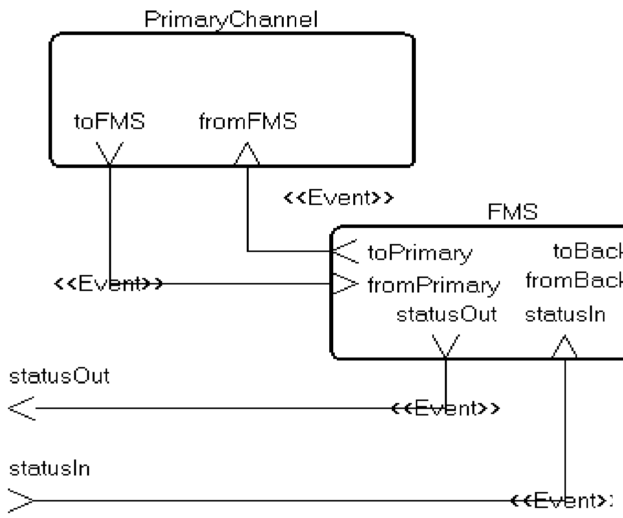


SPCU Diagram
(next hierarchical level)

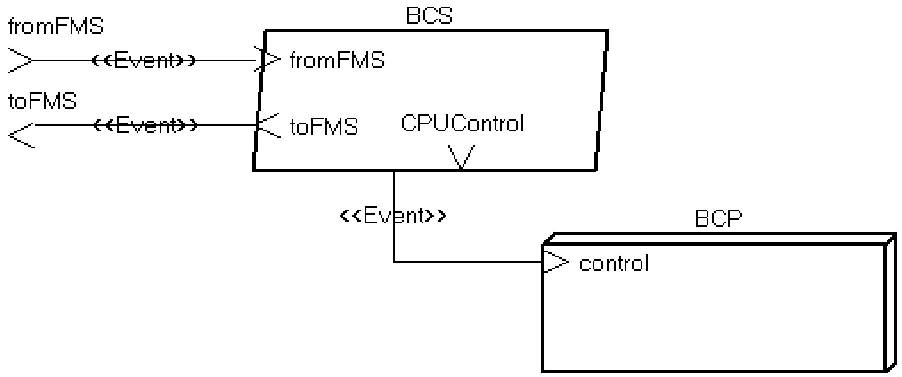


AADL Representation (using TOPCASED, continued)

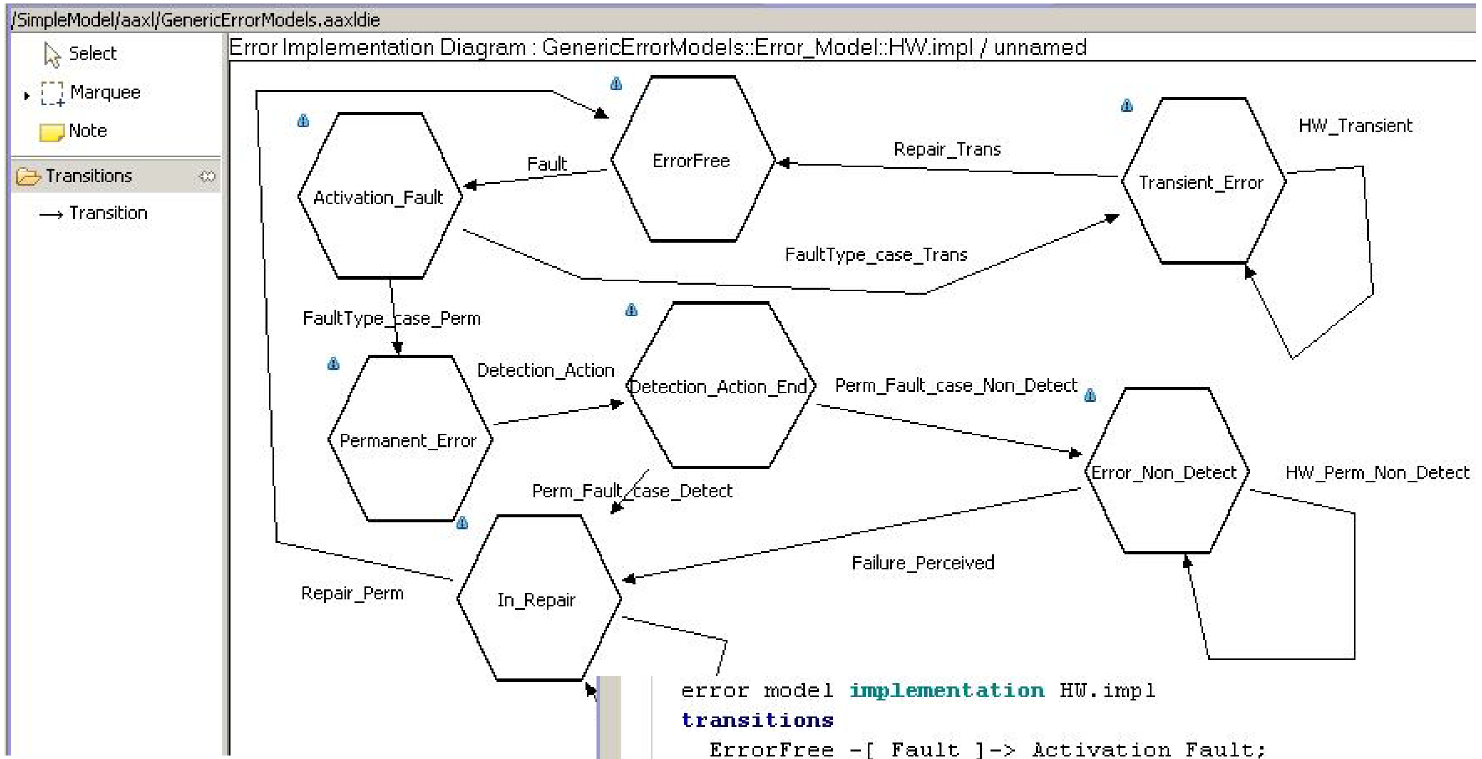
BCU Diagram

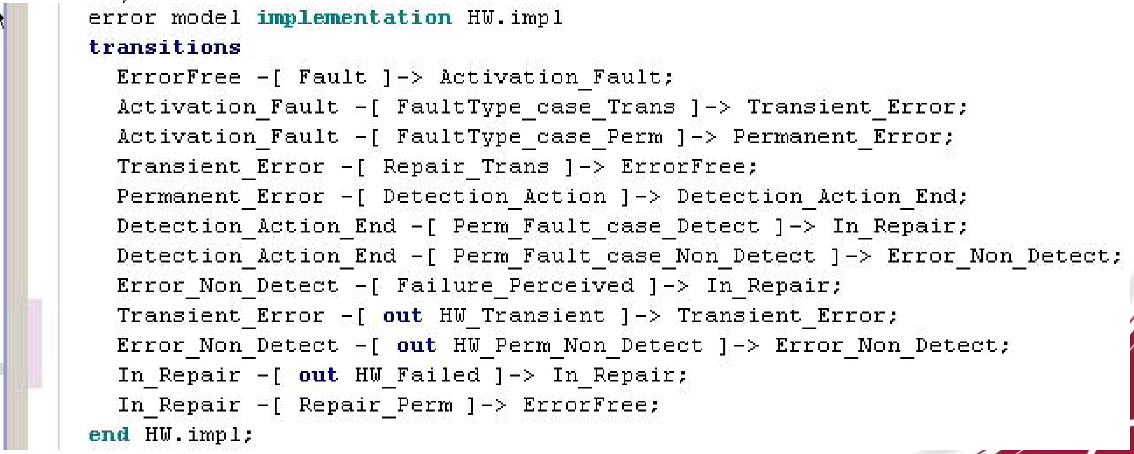


Control Channel Diagram (both Primary and Backup)



Error Model Editor



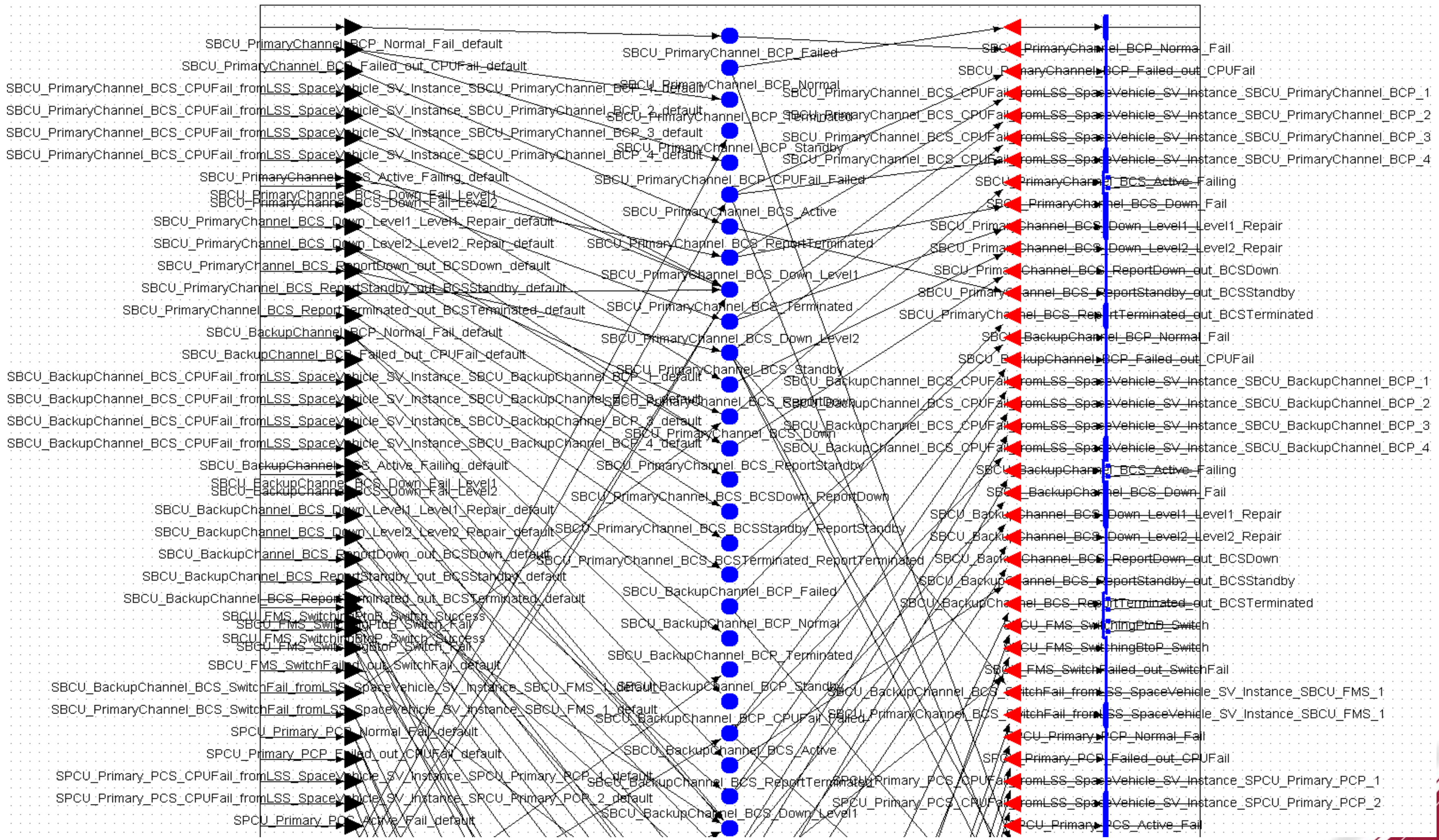


```
error model implementation HW.impl
transitions
  ErrorFree -[ Fault ]-> Activation_Fault;
  Activation_Fault -[ FaultType_case_Trans ]-> Transient_Error;
  Activation_Fault -[ FaultType_case_Perm ]-> Permanent_Error;
  Transient_Error -[ Repair_Trans ]-> ErrorFree;
  Permanent_Error -[ Detection_Action ]-> Detection_Action_End;
  Detection_Action_End -[ Perm_Fault_case_Detect ]-> In_Repair;
  Detection_Action_End -[ Perm_Fault_case_Non_Detect ]-> Error_Non_Detect;
  Error_Non_Detect -[ Failure_Perceived ]-> In_Repair;
  Transient_Error -[ out HW_Transient ]-> Transient_Error;
  Error_Non_Detect -[ out HW_Perm_Non_Detect ]-> Error_Non_Detect;
  In_Repair -[ out HW_Failed ]-> In_Repair;
  In_Repair -[ Repair_Perm ]-> ErrorFree;
end HW.impl;
```

Error Model Implementation for BCU Backup Processor

```
error model implementation BCS.impl
  transitions
    Active -[Failing]-> ReportDown;
    Active -[in Sleep]-> Standby;
    ReportDown -[out BCSFail]-> Down;
    Active -[in Terminate]-> Terminated;
    Down -[Fail_case_Minor]-> DownMinor;
    Down -[Fail_case_Major]-> DownMajor;
    DownMinor -[MinorRepair]-> ReportStandby;
    DownMajor -[MajorRepair]-> ReportStandby;
    ReportStandby -[out BCSStandby]-> Standby;
    Standby -[in Wake]-> Active;
    Standby -[ in SwitchFail ]-> Down;
    Standby -[ in Terminate]-> Terminated;
    Active -[ in CPUFail ]-> ReportTerminated;
    ReportTerminated -[out BCSTerminate]->
            Terminated;
  end BCS.impl;
```

Stochastic Analysis Representation (product of ADAPT-M conversion)



SBCU_PrimaryChannel_BCP_Normal_Fail_default
SBCU_PrimaryChannel_BCP_Failed_out_CPUFail_default
SBCU_PrimaryChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_PrimaryChannel_BCP_1_default
SBCU_PrimaryChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_PrimaryChannel_BCP_2_default
SBCU_PrimaryChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_PrimaryChannel_BCP_3_default
SBCU_PrimaryChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_PrimaryChannel_BCP_4_default
SBCU_PrimaryChannel_BCS_Active_Failing_default
SBCU_PrimaryChannel_BCS_Down_Fail_Level1
SBCU_PrimaryChannel_BCS_Down_Fail_Level2
SBCU_PrimaryChannel_BCS_Down_Level1_Level1_Repair_default
SBCU_PrimaryChannel_BCS_Down_Level2_Level2_Repair_default
SBCU_PrimaryChannel_BCS_ReportDown_out_BCSDown_default
SBCU_PrimaryChannel_BCS_ReportStandby_out_BCSStandby_default
SBCU_PrimaryChannel_BCS_ReportTerminated_out_BCSTerminated_default
SBCU_BackupChannel_BCP_Normal_Fail_default
SBCU_BackupChannel_BCP_Failed_out_CPUFail_default
SBCU_BackupChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_BackupChannel_BCP_1_default
SBCU_BackupChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_BackupChannel_BCP_2_default
SBCU_BackupChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_BackupChannel_BCP_3_default
SBCU_BackupChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_BackupChannel_BCP_4_default
SBCU_BackupChannel_BCS_Active_Failing_default
SBCU_BackupChannel_BCS_Down_Fail_Level1
SBCU_BackupChannel_BCS_Down_Fail_Level2
SBCU_BackupChannel_BCS_Down_Level1_Level1_Repair_default
SBCU_BackupChannel_BCS_Down_Level2_Level2_Repair_default
SBCU_BackupChannel_BCS_ReportDown_out_BCSDown_default
SBCU_BackupChannel_BCS_ReportStandby_out_BCSStandby_default
SBCU_BackupChannel_BCS_ReportTerminated_out_BCSTerminated_default
SBCU_FMS_SwitchingPtoB_Switch_Success
SBCU_FMS_SwitchingPtoB_Switch_Fail
SBCU_FMS_SwitchingBtoP_Switch_Success
SBCU_FMS_SwitchingBtoP_Switch_Fail
SBCU_FMS_SwitchFailed_out_SwitchFail_default
SBCU_BackupChannel_BCS_SwitchFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_FMS_1_default
SBCU_PrimaryChannel_BCS_SwitchFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_FMS_1_default
SPCU_Primary_PCP_Normal_Fail_default
SPCU_Primary_PCP_Failed_out_CPUFail_default
SPCU_Primary_PCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SPCU_Primary_PCP_1_default
SPCU_Primary_PCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SPCU_Primary_PCP_2_default
SPCU_Primary_PCS_Active_Fail_default
SBCU_PrimaryChannel_BCP_Failed
SBCU_PrimaryChannel_BCP_Normal
SBCU_PrimaryChannel_BCP_Terminated
SBCU_PrimaryChannel_BCP_Standby
SBCU_PrimaryChannel_BCP_CPUFail_Failed
SBCU_PrimaryChannel_BCS_Active
SBCU_PrimaryChannel_BCS_ReportTerminated
SBCU_PrimaryChannel_BCS_Down_Level1
SBCU_PrimaryChannel_BCS_Terminated
SBCU_PrimaryChannel_BCS_Down_Level2
SBCU_PrimaryChannel_BCS_Standby
SBCU_PrimaryChannel_BCS_ReportDown
SBCU_PrimaryChannel_BCS_Down
SBCU_PrimaryChannel_BCS_ReportStandby
SBCU_PrimaryChannel_BCS_BCSDown_ReportDown
SBCU_PrimaryChannel_BCS_BCSStandby_ReportStandby
SBCU_PrimaryChannel_BCS_BCSTerminated_ReportTerminated
SBCU_BackupChannel_BCP_Failed
SBCU_BackupChannel_BCP_Normal
SBCU_BackupChannel_BCP_Terminated
SBCU_BackupChannel_BCP_Standby
SBCU_BackupChannel_BCP_CPUFail_Failed
SBCU_BackupChannel_BCS_Active
SBCU_BackupChannel_BCS_ReportTerminated
SBCU_BackupChannel_BCS_Down_Level1
SBCU_PrimaryChannel_BCP_Normal_Fail
SBCU_PrimaryChannel_BCP_Failed_out_CPUFail
SBCU_PrimaryChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_PrimaryChannel_BCP_1
SBCU_PrimaryChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_PrimaryChannel_BCP_2
SBCU_PrimaryChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_PrimaryChannel_BCP_3
SBCU_PrimaryChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_PrimaryChannel_BCP_4
SBCU_PrimaryChannel_BCS_Active_Failing
SBCU_PrimaryChannel_BCS_Down_Fail
SBCU_PrimaryChannel_BCS_Down_Level1_Level1_Repair
SBCU_PrimaryChannel_BCS_Down_Level2_Level2_Repair
SBCU_PrimaryChannel_BCS_ReportDown_out_BCSDown
SBCU_PrimaryChannel_BCS_ReportStandby_out_BCSStandby
SBCU_PrimaryChannel_BCS_ReportTerminated_out_BCSTerminated
SBCU_BackupChannel_BCP_Normal_Fail
SBCU_BackupChannel_BCP_Failed_out_CPUFail
SBCU_BackupChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_BackupChannel_BCP_1
SBCU_BackupChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_BackupChannel_BCP_2
SBCU_BackupChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_BackupChannel_BCP_3
SBCU_BackupChannel_BCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_BackupChannel_BCP_4
SBCU_BackupChannel_BCS_Active_Failing
SBCU_BackupChannel_BCS_Down_Fail
SBCU_BackupChannel_BCS_Down_Level1_Level1_Repair
SBCU_BackupChannel_BCS_Down_Level2_Level2_Repair
SBCU_BackupChannel_BCS_ReportDown_out_BCSDown
SBCU_BackupChannel_BCS_ReportStandby_out_BCSStandby
SBCU_BackupChannel_BCS_ReportTerminated_out_BCSTerminated
SBCU_FMS_SwitchingPtoB_Switch
SBCU_FMS_SwitchingBtoP_Switch
SBCU_FMS_SwitchFailed_out_SwitchFail
SBCU_BackupChannel_BCS_SwitchFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_FMS_1
SBCU_PrimaryChannel_BCS_SwitchFail_fromLSS_SpaceVehicle_SV_Instance_SBCU_FMS_1
SPCU_Primary_PCP_Normal_Fail
SPCU_Primary_PCP_Failed_out_CPUFail
SPCU_Primary_PCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SPCU_Primary_PCP_1
SPCU_Primary_PCS_CPUFail_fromLSS_SpaceVehicle_SV_Instance_SPCU_Primary_PCP_2
SPCU_Primary_PCS_Active_Fail

LSS – Results of MoBIUS processing

Performance Variable	Simulated Mean Value
Mission Duration	**95038 hours**
Space Vehicle Online Time	**67291 hours**
Payload Online Time	**67291 hours**
Payload Down Time	**26026 hours**
Bus Online Time	**71069 hours**

Future Plans

- TOPCASED robustness improvements
- Addition of automated FMEA
- Application to more systems

Conclusions

- A tool set using a common language between system engineering and dependability engineering fori Space Systems
  - *Can enable better decision support*
  - *Enables tradeoffs and analyses during the early design phases, but can be used during other phases*
- AADL currently offers the best semantics
  - *Failure rate and failure mode definition*
  - *Inclusion of probability distributions*
  - *but progress is being made in other OMG-sponsored efforts*
- Tool set is based on public domain software
  - *Enables cooperative development*
  - *Less dependence on commercial vendor viability – challenging in a dynamic and small market place*
  - *Tradeoff: configuration control and more owner responsibility for maintenance and debugging*

Acronyms

ADAPT: AADL Architectural models to stochastic Petri nets through model Transformation,

AADL: Architecture Analysis & Design Language

BCP: Bus Control Processor

BCS: Bus Control Software

BCU: Bus Control Unit

EMF: Eclipse Modeling Framework (part of Eclipse)

GEF: Graphical Editing Framework (part of Eclipse)

GMF: Graphical Modeling Framework (part of Eclipse)

GSPN: Generalized Stochastic Petri Net

LSS: Low Orbit Space System

MOBIUS: Model-Based Environment for Validation of System Reliability, Availability, Security, and Performance

OSATE: Open Source AADL Tool Environment (Software tool integrated into Eclipse)

PCP: Payload Control Processor

PCS: Payload Control Software

PCU: Payload Control Unit

SAN: Stochastic Analysis Network

TOPCASED: Toolkit In OPen source for Critical Applications & SystEms Development

References

- Society of Automotive Engineers (SAE) Aerospace Standard AS5506 (2004)
- A. Rugina, K. Kanoun, M Kaaniche, “The ADAPT Tool: From AADL Architectural Models to Stochastic Petri Nets through Model Transformation,” *7th European Dependable Computing Conference (EDCC),* Kaunas : Lituanie (2008)
- Peter Feiler and Anna Rugina, Dependability Modeling with the Architecture Analysis & Design Language (AADL), Software Engineering Institute report CMU/SEI-2007-TN-043, July 2007, available from www.sei.cmu.edu
- D. D. Deavours, G. Clark, T. Courtney, D. Daly, S. Derisavi, J. M. Doyle, W. H. Sanders, and P. G. Webster, “The Mobius framework and its implementation,” IEEE Trans. on Soft. Eng., vol. 28, no. 10, pp. 956–969, October 2002.