Specifying Security & Safety Requirements & Solutions with the Society of Automotive Engineers' (SAE) Architectural Analysis & Design Language (AADL)

Ed Colbert
Absolute Software Co., Inc. & University of Southern California Center for Software Engineering
colbert@abssw.com or ecolbert@usc.edu

James E. L
High Integrity Solutions Ltd & University of Southern California Center for Software Engineering
land@highintegritysolutions.com

Mayurkumar Patel
University of Southern California Center for Software Engineering
mayurkup@usc.edu

Some of these materials were prepared by Peter Feiler of Software Engineering Institute, Carnegie Mellon University, Bruce Lewis of U.S. Army Avionics & Missile Command, or Ed Colbert of Absolute Software Co., Inc., and later modified with permission by Ed Colbert, James Land, and Mayur Patel.

Goal of Presentation

- ❑ Look at how Society of Automotive Engineers' (SAE) Architectural Analysis & Design Language (AADL) supports
  - Specifying Security & Safety Requirements
  - Defining secure &/or safe architecture
  - Analyzing architecture

Outline

- Safe & Secure Systems
- Background of SAE AADL
- Overview of AADL Major Features & How they support Safety & Security
- Using Error Model Annex & GSN for Safety or Security
- Example using Error Model Annex and GSN
- Further Work Required
- AADL & its Relationship to SysML
- Summary

Safety & Security in Embedded, Real–time, Systems

- Most embedded, real–time systems developed in last half of 20th century
  - Safety often an issue
    - Late result or bad result → injury or death

    e.g.
      - London ambulance project
      - Medical radiation machine
      - Factory controllers
      - Avionics
      - Computer control breaking in auto's
      - Nuclear reactor control

Safety & Security in Embedded, Real–time, Systems (cont.)

- For last half of 20th century (cont.)
  - Security often handled by physical controls
    - Isolated systems

      e.g. Vehicle control
    - Non–networked / isolated bus / shared memory

      e.g. Avionics systems
    - Private–network systems

      e.g. Air-traffic control systems, networks of automated teller machines

Safety & Security in Embedded, Real–time, Systems (cont.)

- Current & future system using internet & wireless networks to satisfy users' growing need for
  - Quick access to information to understand "real–time" situation

    e.g.
    - Just–in–time production or inventory management
    - Emergency response
    - Battle command
  - Remote control of remote systems

    e.g.
    - Surveillance
    - Hazardous material cleanup
    - Unmanned vehicles

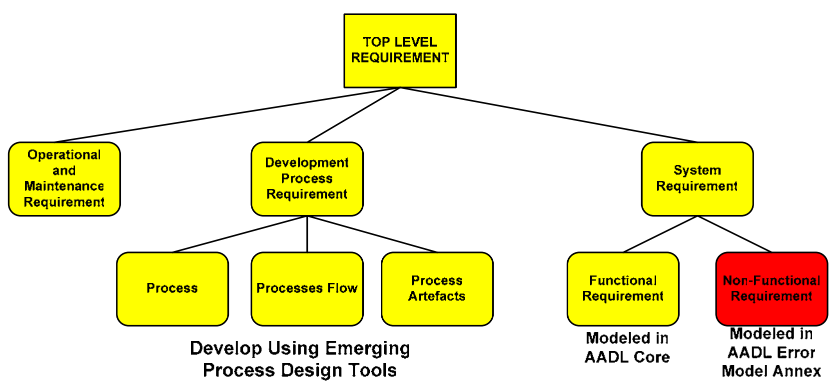
Safety & Security in Embedded & Real–time, Systems (cont.)

- Ineffective security threatens safety of
  - Embedded system as evolve into System of Systems

    e.g.
    - Penetration to gather/change secret data or algorithms
    - Man-in-middle" attack to gather or change information
    - Enemy captures networked vehicle & crew in battle
    - "Denial of Service" attack on emergency control center
  - Financial & commercial systems as use internet to support real-time sales

    e.g. Denial of Service (DoS) attacks on web-sites

Effect of High-level Safety & Security Requirements



Background of Industrial Safety Engineering

- Civil Aviation
  - Safety Engineering dictated by Law for Civil Aviation
    - Title 14, *Code of Federal Regulations* (*CFR*)
      - *FAA FAR 25.1309* for air transport
      - *Advisory Circular AC 25.1309* provides guidance
        - » Defines list of design principles
        - » Cites *RTCA DO-178B* as acceptable means to assure software compliance
  - *SAE Aerospace Recommended Practice (ARP) 4761*
    - Provides procedures for compliance with AC 25.1309

Background of Industrial Safety Engineering (cont.)

- Other Industrial Practices
  - European systems governed by *IEC 41508* for functional safety of electrical/electronic equipment
    - Establishes
      - 4 levels of Safety Integrity
      - Means to show compliance with Safety Integrity Levels
  - National laws require development of *Safety Case* to show achievement of safety levels
- Military Standards
  - US DOD Mil Std 882D for ground
  - UK MOD Def Stan 00-52 & 00-55
- *Goal Structured Notation* (*GSN*)
  - Top-down method to show satisfaction of safety goals

AC 25.1309 Safety Design Principles

- ❑ Design integrity & quality
  - – Ensure intended functions & prevent failures through design
  - – Must consider life limits
- ❑ Redundant or backup systems
- ❑ Isolation (independence)
- ❑ Proven reliability
- ❑ Failure warning & indication
- ❑ Checkable
- ❑ Designed failure effect limits
  - – Design to sustain damage
- ❑ Designed failure path
  - – Limit impact on safety
- ❑ Margins or factors of safety
- ❑ Error tolerance

(c) 2004 High Integrity Systems, Ltd. & Absolute Software Co., Inc. 11 10/13/2004

Security Engineering Background

- ❑ *Common Criteria* (*CC*) defines
  - – Sets of
    - • *Security Function Requirements* (*SFR*)
    - • *Security Assurance Requirements* (*SAR*)
      - ▪ 7 different *Evaluation Assurance Levels* (EAL)
  - – 4 major artifacts
    - • 3 *Target Of Evaluation* (*TOE*)
      - ▪ System being developed with its administrator & user documentation
    - • Protection *Profile* (*PP*)
      - ▪ Defines implementation–independent set of security requirements for TOE
    - • *Security Target* (*ST*)
      - ▪ Defines implementation–dependent design of security mechanisms for TOE
    - • 4 *package*
      - ▪ Set of components that subset of SAR's & SFR's
- ❑ *Common Evaluation Method* (*CEM*) describes how to evaluate systems using CC
- ❑ GSN can be enabling tool for security also
  - – Can use to develop top-down argument to show design satisfies SFR

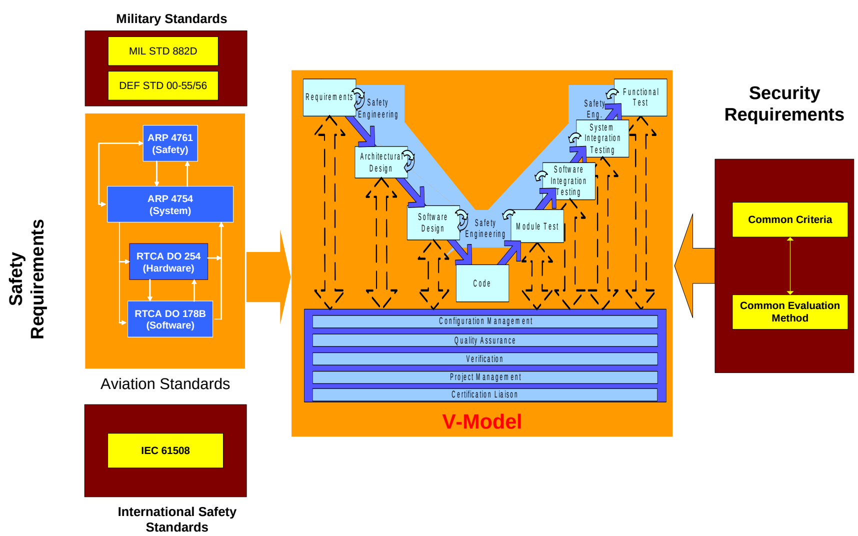
(c) 2004 High Integrity Systems, Ltd. & Absolute Software Co., Inc. 12 10/13/2004

Importance of Engineering Process for Safe & Secure Systems

- Very difficult to prove correctness of large & complex software systems
  - Industrial relies on
    - Rigorous processes
    - *Independent verification*
- Requirements engineering
  - Safety
  - Security
  - Other non-functional
- “V” design model
  - Core engineering process
  - Integrated with safety & security engineering process

 13 10/13/2004

Putting It All Together



 14 10/13/2004

Outline

- Safe & Secure Systems
- Background of SAE AADL
- Overview of AADL Major Features & How they support Safety & Security
- Using Error Model Annex & GSN for Safety or Security
- Example using Error Model Annex and GSN
- Further Work Required
- AADL & its Relationship to SysML
- Summary

 15 10/13/2004

SAE Architecture Analysis & Design Language (AADL)

- Based on 15 years of DARPA funded research
  - Language definition
  - Evaluation & demonstration projects
- Base standard approved by
  - Working group in July 2004
  - SAE board in October 2004
- Annexes in committee review
  - Graphical Language annex
  - UML profile annex
  - XML annex
  - Error model/dependability annex
- Sponsored by
  - SAE International
  - Avionics Systems Division (ASD)
  - Embedded Systems (AS2)
  - AADL Subcommittee (AS-2C)
- Contact
  - Bruce Lewis AS-2C chair, bruce.a.lewis@us.army.mil
  - http://www.aadl.info
  - For Information email to info@aadl.info

 16 10/13/2004

SAE AADL

- Specification of
  - *Predictable* systems
    - Real-time
    - Embedded
    - Fault-tolerant
    - Securely partitioned
    - Dynamically configurable
  - Software task & communication architectures
  - Software bound to distributed, multiple-processor, hardware architectures
- Fields of application
  - Avionics, Automotive, Aerospace, Autonomous systems, Medical, …

AADL Evolution from Honeywell's Meta-H

1991	DARPA DSSA Meta-H program begins
1992	Partitioned PFP target (Tartan MAR/i960MC)
1994	Multi-processor target (VME i960MC)
1995	Slack stealing scheduler
1998	Portable Ada 95 and POSIX middleware configurations
1999	Hybrid automata verification of core middleware modules
2000	SAE AADL Standard Working Group Forms
2001	(Spring) Requirements document, ARD5296, approved by SAE
2004	(July) AADL WG Approves Standard
2004	(Oct) SAE Approves Standard
2005	*(Jan?) Target for AADL WG approval of Annexes & AADL v1.1*
2007/8	*Target for approval of AADL v2.0*

1992-present Evaluation and demonstration projects include

- Missile G&C reference architecture, demos, others (AMCOM SED)
- Hybrid automata formal verification (AFOSR, Honeywell)
- Missile defense (Boeing)
- Fighter guidance SW fault tolerance (DARPA, CMU, Lockheed-Martin)
- Incremental Upgrade of Legacy Systems (AFRL, Boeing, Honeywell)
- Comanche study (AMCOM, Comanche PO, Boeing, Honeywell)
- Tactical Mobile Robotics (DARPA, Honeywell, Georgia Tech)
- Advanced Intercept Technology CWE (BMDO, MaxTech)
- Adaptive Computer Systems (DARPA, Honeywell)
- Avionics System Performance Management (AFRL, Honeywell)
- Ada Software Integrated Development/Verification (AFRL, Honeywell)
- FMS reference architecture (Honeywell)
- JSF vehicle control (Honeywell)
- IFMU reengineering (Honeywell)

Organizations Evaluating or Planning to use AADL

- Airbus — New System Engineering Approach incorporates AADL
- ESA — Modeling of Satellite Systems, Architecture Verification - ASSERT
- Rockwell Collins — Modeling of Avionics Software System
- Lockheed Martin
- Smith Industries
- Raytheon
- Boeing
- Common Missile
- System Plug and Play

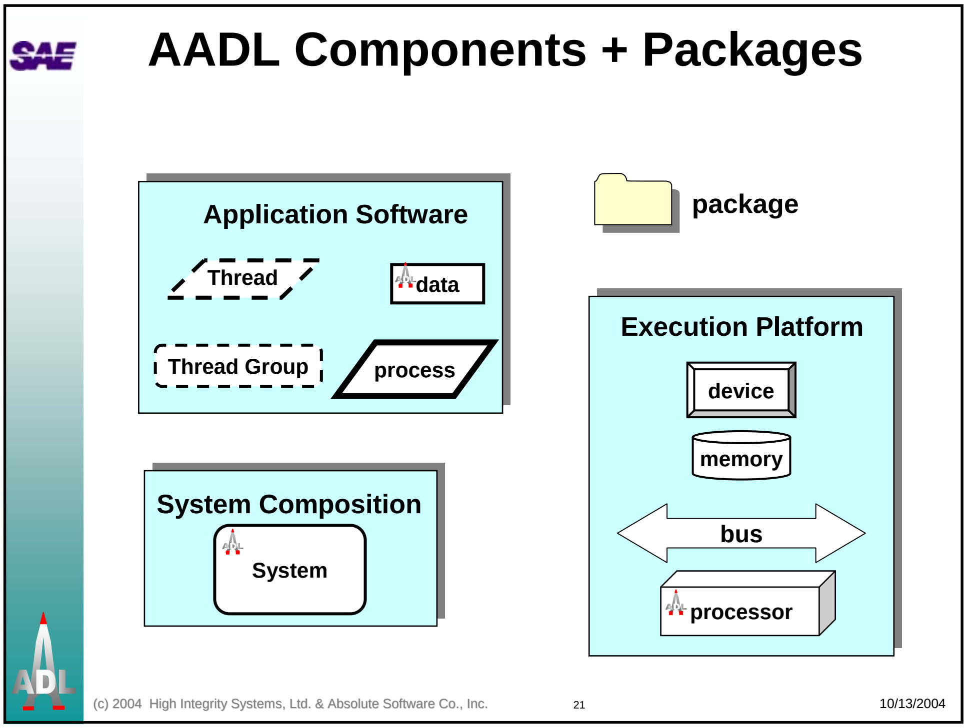
Embedded System Engineering & AADL

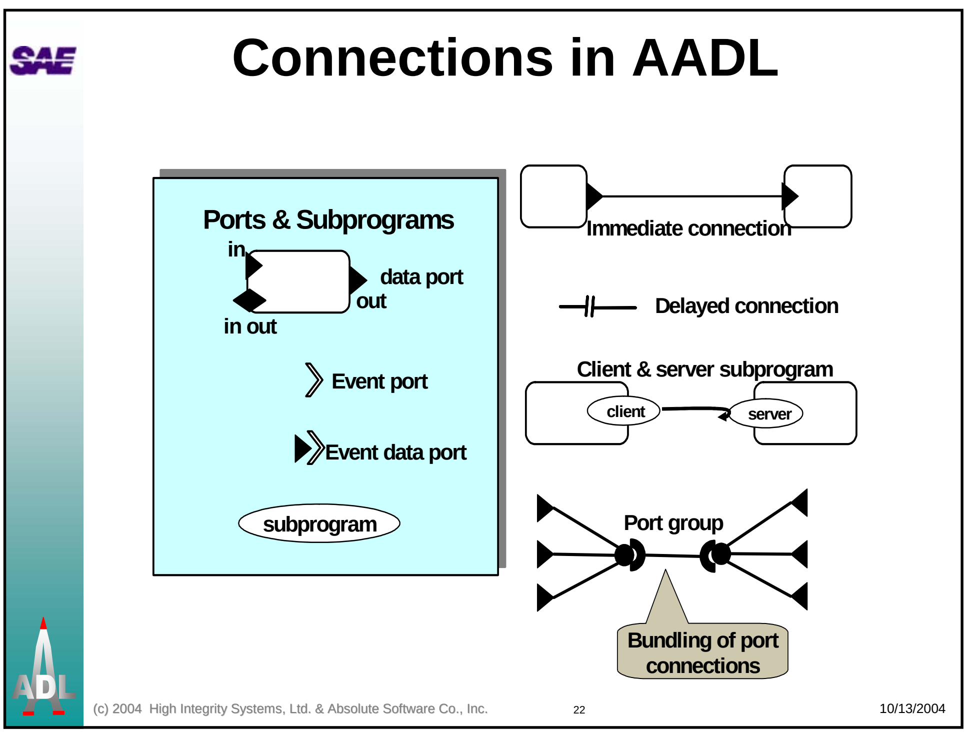
Apply AADL for systems integration modeling & analysis

NATO/SAE AS1 Weapon System Integration

Outline

- Safe & Secure Systems
- Background of SAE AADL
- Overview of AADL Major Features & How they support Safety & Security
- Using Error Model Annex & GSN for Safety or Security
- Example using Error Model Annex and GSN
- Further Work Required
- AADL & its Relationship to SysML
- Summary


AADL Components + Packages
Application Software
Thread
data
Thread Group
process
package
Execution Platform
device
memory
bus
processor
System Composition
System
(c) 2004 High Integrity Systems, Ltd. & Absolute Software Co., Inc.
21
10/13/2004

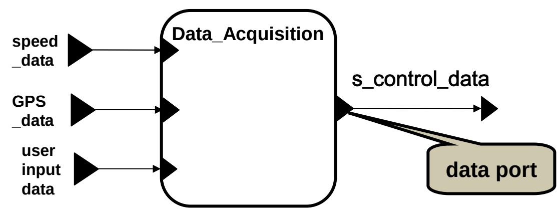

Connections in AADL
Ports & Subprograms
in
data port
out
in out
Event port
Event data port
subprogram
Immediate connection
Delayed connection
Client & server subprogram
client
server
Port group
Bundling of port connections
(c) 2004 High Integrity Systems, Ltd. & Absolute Software Co., Inc.
22
10/13/2004

Graphical & Textual Notation

```
system Data_Acquisition
provides
  speed_data: in data metric_speed;
  GPS_data: in data position_carthesian;
  user_input_data: in data user_input;
  s_control_data:out data state_control;
end Data_Acquisition;
```

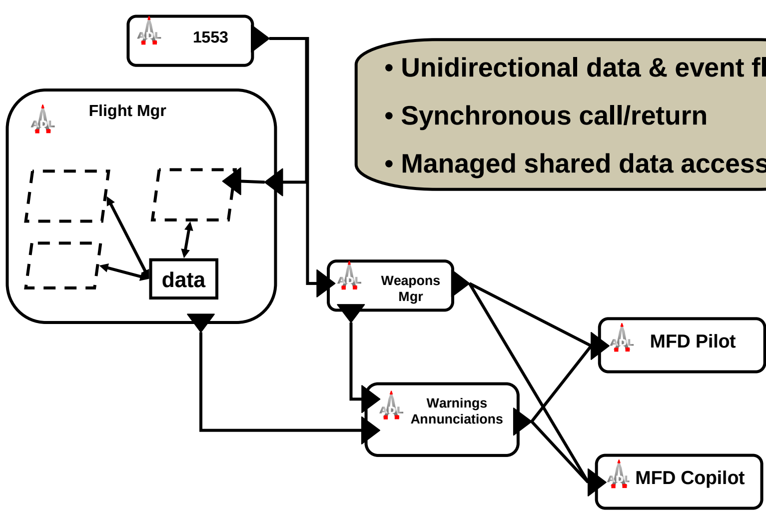
data port

data type of port

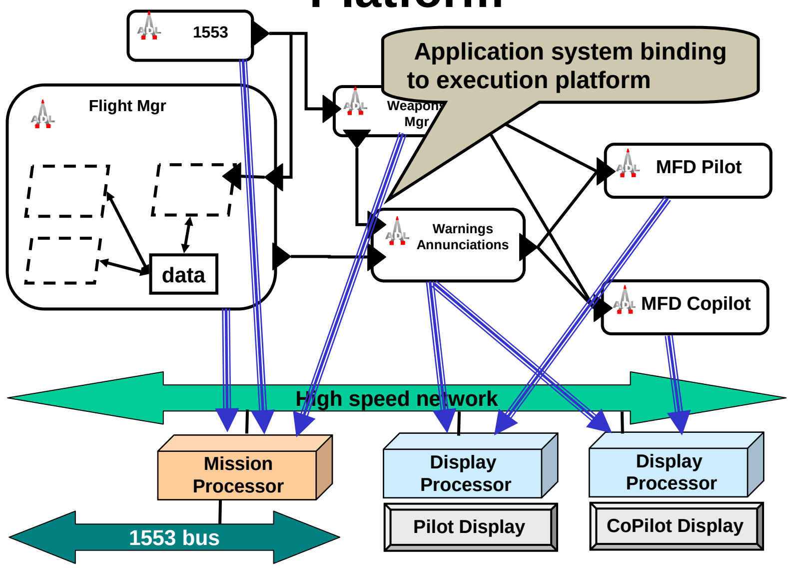


AADL Component Interaction

- Unidirectional data & event flow
- Synchronous call/return
- Managed shared data access



Application System & Execution Platform



 25 10/13/2004

AADL Features & How Supports Safety & Security Analyses

- Provides well-defined semantic foundation for architecture description
  - Facilitates
    - Specification of
      - Target of Evaluation (TOE) Description
      - TOE Security Environment
      - Security Objectives
        - e.g. Identification of Assets (i.e. Shared data & data ports)
      - IT Security Requirements
      - Rationale required for Protection Profile (PP)
    - Creation of Security Target (ST) artifacts
    - Analyzing whether requirements are met
    - Developing tools for analysis

 26 10/13/2004

AADL Error Model Annex

- Optional set of declarations & semantics

  e.g. new properties of components that support addition analysis techniques

- Specify error models
- Can use to define qualitative & quantitative analysis of non-functional requirements (NFR)

  e.g.
  - Security
  - Safety,
  - Reliability,
  - Availability, &
  - Maintainability

 27 10/13/2004

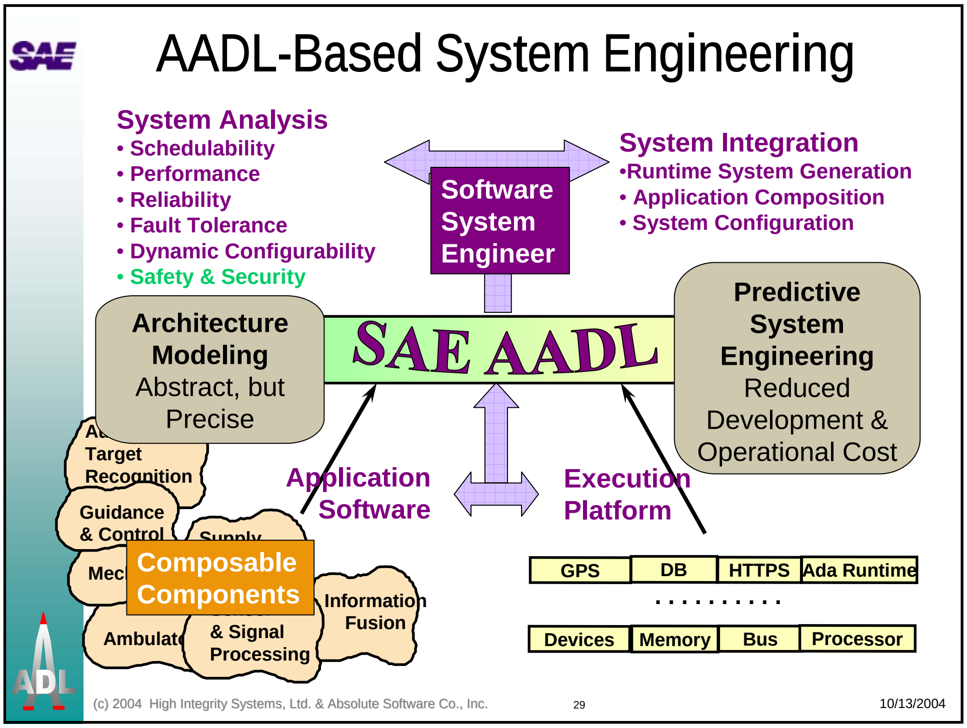
AADL Features & How Supports Safety & Security Analyses (cont.)

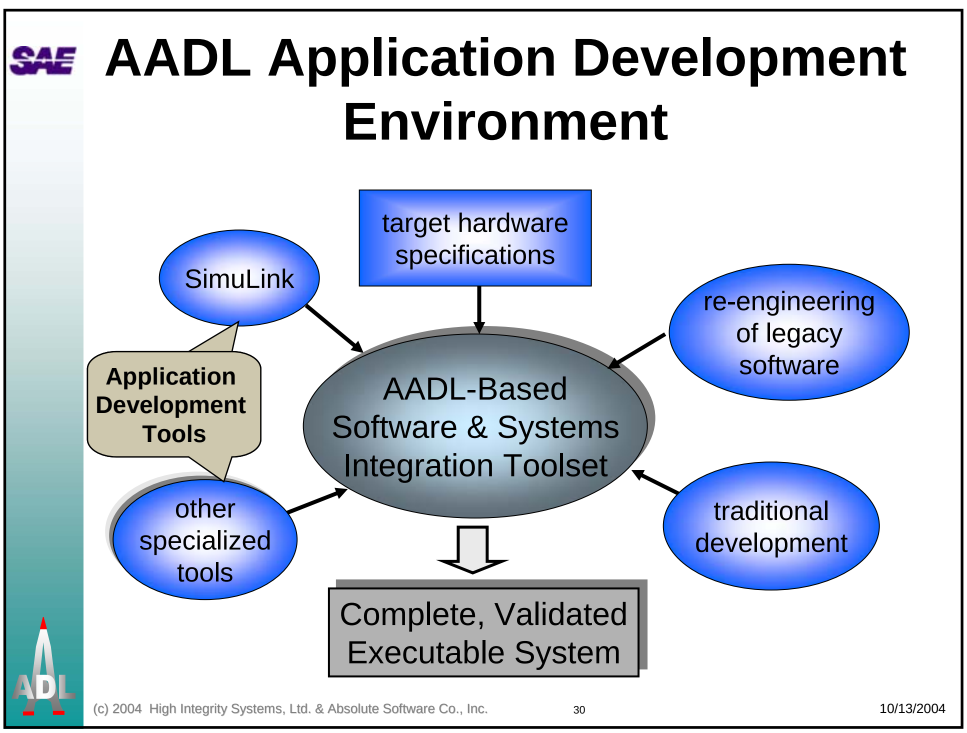
- Can extend language
  - Property sets
  - "Annexes"
    - Error model
    - Other models or languages
      - e.g. for formal constraint or specification languages
- Can define safety engineering concepts

  e.g.
  - MTBF of physical objects
  - Propagation through connections
- Can define security concepts

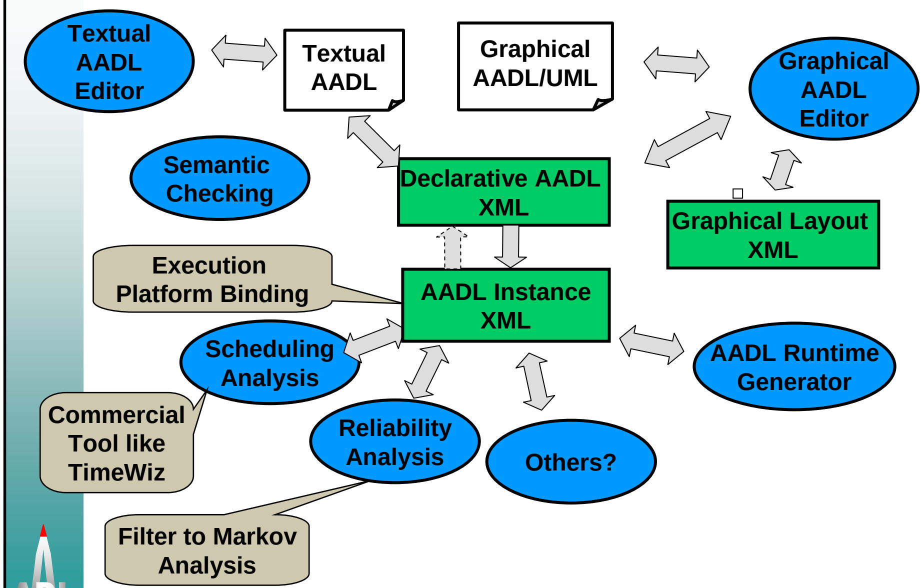
  e.g.
  - State propagation to "declassify-immediate"
  - Fail secure attributes model
  - Probability of surreptitious penetration

 28 10/13/2004


AADL-Based System Engineering
System Analysis
• Schedulability
• Performance
• Reliability
• Fault Tolerance
• Dynamic Configurability
• Safety & Security
Software System Engineer
System Integration
•Runtime System Generation
• Application Composition
• System Configuration
Architecture Modeling
Abstract, but Precise
SAE AADL
Predictive System Engineering
Reduced Development & Operational Cost
Target Recognition
Guidance & Control
Composable Components
& Signal Processing
Information Fusion
Application Software
Execution Platform
GPS
DB
HTTPS
Ada Runtime
Devices
Memory
Bus
Processor
(c) 2004 High Integrity Systems, Ltd. & Absolute Software Co., Inc.
29
10/13/2004


AADL Application Development Environment
target hardware specifications
SimuLink
re-engineering of legacy software
Application Development Tools
AADL-Based Software & Systems Integration Toolset
other specialized tools
traditional development
Complete, Validated Executable System
(c) 2004 High Integrity Systems, Ltd. & Absolute Software Co., Inc.
30
10/13/2004

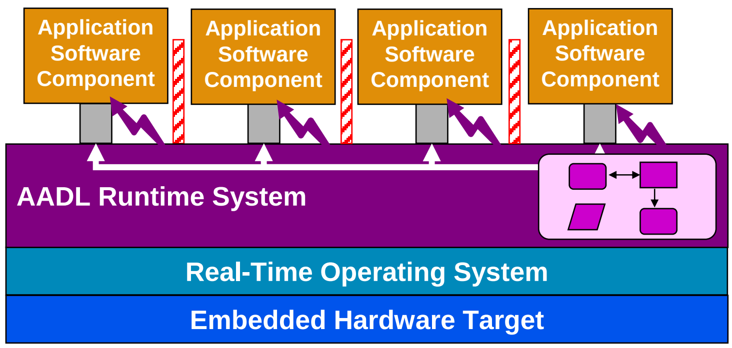
XML-Based AADL Tool Strategy



How AADL Development Environment Supports Safety & Security Analyses

- ❑ Aids verification & certification processes
  - – Tools can verify that application code in execution image was specified in architecture
  - – Reliability analysis tool(s) can verify that code is reliable enough to meet security & safety requirements
  - – System generator produces
    - Execution image
    - Execution-platform specific to
      - Run-time environment
      - Application-required facilities
        - » i.e. can eliminate or block access to unused environment services
    - “Glue” code to support communication between components
  - – Can “easily” add tool(s) can analyze specific security & safety requirements

Application Plug–in Integration



Strong Partitioning
- Timing Protection
- OS Call Restrictions
- Memory Protection

Interoperability/Portability
- Tailored Runtime Executive
- Standard RTOS API
- Application Components

 33 10/13/2004

How AADL Execution Environment Supports Safety & Security Analyses

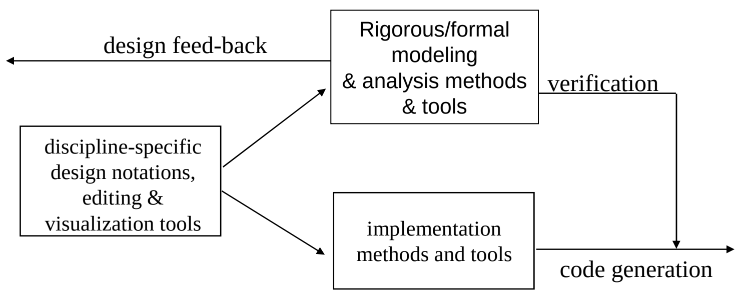
- ❑ Aids verification & certification processes
  - Formally specified using Finite Automata Model
  - Environment enforces time & space partition specified in architecture description
  - Certifying AADL environment will reduce cost of certifying systems developed for same hardware

 34 10/13/2004

AADL Engineering Paradigm

- ❑ Specify SW & HW architecture using formal semantics
  - Interfaces
  - Properties
- ❑ Analyze system concerns early
  - e.g. Safety &/or security
- ❑ Eliminate errors thru automatic generation & integration
- ❑ Verify compliance of source code, middleware behavior, environment, platform
- ❑ Model & analyze throughout product life cycle

  e.g. when refine models & components



 35 10/13/2004

How AADL Engineering Paradigm Supports Safety & Security Analyses

- ❑ Easier to
  - Keep architecture & implementation consistent
    - Auto generation & integration
    - Verification of consistent
  - Re-analyze changes to architecture
  - Add new analysis techniques
- ❑ Because easier to maintain architecture & consistency with implementation
  - More likely to be done

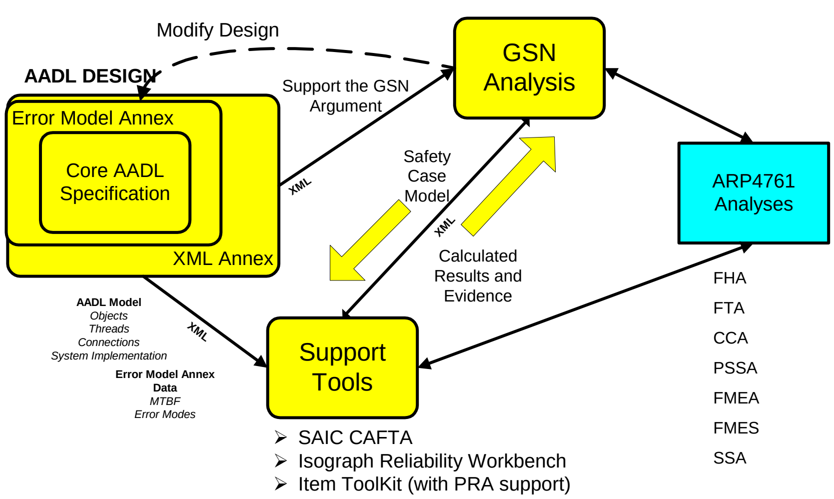
 36 10/13/2004

Outline

- ❑ Safe & Secure Systems
- ❑ Background of SAE AADL
- ❑ Overview of AADL Major Features & How they support Safety & Security
- ➡ Using Error Model Annex & GSN for Safety or Security
- ❑ Example using Error Model Annex & GSN
- ❑ Further Work Required
- ❑ AADL & its Relationship to SysML
- ❑ Summary

 37 10/13/2004

Integration of AADL with Development of Safe &/or Secure System



 38 10/13/2004

Outline

- Safe & Secure Systems
- Background of SAE AADL
- Overview of AADL Major Features & How they support Safety & Security
- Using Error Model Annex & GSN for Safety or Security
- Example using Error Model Annex and GSN
- Further Work Required
- AADL & its Relationship to SysML
- Summary

 39 10/13/2004

Tram Example

- Model Problem
  - Automated Tram System running on tram way
- Next Slides show
  - Partial model of tram system using basic AADL
    - AADL Model of Tram Navigation System
      - has 3 redundant sensors, a bus for communication & health monitor.
  - Use of Error Model annex to augment base model with non-functional safety properties
  - The use of the AADL Error Model Annex
    - Errors propagating
      - Sensor Failures
      - Detecting fault which causes stop of tram

 40 10/13/2004

AADL Model (Sensors)

```
device type GPSSensor
features
    controller_sensor: requires bus access CommunicationBus;
    VoteOut : out event port;
    VoteIn : in event port;
end GPSSensor;

device implementation GPSSensor.type1
end GPSSensor.type1;

device implementation GPSSEnsor.type2
end GPSSensor.type2;

device implementation GPSSEnsor.type3
end GPSSensor.type3;
```


AADL Model (bus + monitor)

```
bus CommunicationBus
end CommunicationBus;

bus implementation CommunicationBus.Type1
end CommunicationBus.Type1;

processor type health_monitor
features
    controller_sensor: requires bus access CommunicationBus;
    VoteOut : out event port;
    VoteIn : in event port;
    StopNow: out event port;
end health_monitor;

processor implementation health_monitor.type1
end health_monitor.type1;
```


AADL Model (Safety)

```
system type navigation.type1
subcomponents
    sens1: device GPSSensor.type1;
    sens2: device GPSSensor.type2;
    sens3: device GPSSensor.type3;
    bus1: bus CommunicationBus.type1;
    monitor: processor health_monitor.type1;
connections
    conn1: bus access bus1 -> sens1.controller_sensor in modes (ALL_Working,
    sens2_Failed, sens3_Failed);
    conn2: bus access bus1 -> sens2.controller_sensor in modes (ALL_Working,
    sens1_Failed, sens3_Failed);
    conn3: bus access bus1 -> sens3.controller_sensor in modes (ALL_Working,
    sens1_Failed, sens2_Failed);
    conn4: bus access bus1 -> monitor.controller_sensor in modes (ALL_Working,
    sens1_Failed, sens2_Failed, sens3_Failed);
```


AADL Model (Safety)

```
modes
    All_Working: initial mode;
    sens1_Failed: mode;
    sens2_Failed: mode;
    sens3_Failed: mode;
    bus1_Failed: mode;
    Fail_Stop: mode;
    All_Working -[sens1.VoteOut]-> sens1_Failed;
    All_Working -[sens2.VoteOut]-> sens2_Failed;
    All_Working -[sens3.VoteOut]-> sens3_Failed;
    sens1_Failed -[monitor.StopNow]-> Fail_Stop;
    sens2_Failed -[monitor.StopNow]-> Fail_Stop;
    sens3_Failed -[monitor.StopNow]-> Fail_Stop;
end navigation.type1;
```


AADL Model (Safety) – Error Annex

```
{**
error model BasicErrors
features
    All_Working, initial error state;
    sens1_Failed in out error state;
    sens2_Failed in out error state;
    sens3_Failed in out error state;
    bus1_Failed in out error state;
    Fail_Stop in out error state;
end BasicErrors;
```


AADL Model (Safety) – Error Annex

```
error model implementation BasicErrors.Implementation1
features
    WrongDataReading:  fault event;
    Byzantine: fault event;
    MonitorStopNow: fault event;
modes
    All_Working -[WrongDataReading, in sens1_Failed]-> sens1_Failed;
    All_Working -[WrongDataReading, in sens2_Failed]-> sens2_Failed;
    All_Working -[WrongDataReading, in sens3_Failed]-> sens3_Failed;
    All_Working -[WrongDataReading, in bus1_Failed]-> bus1_Failed;
```

AADL Model (Safety) – Error Annex

```
    All_Working -[Byzantine, in sens1_Failed]-> sens1_Failed;
    All_Working -[Byzantine, in sens2_Failed]-> sens2_Failed;
    All_Working -[Byzantine, in sens3_Failed]-> sens3_Failed;
    All_Working -[Byzantine, in bus1_Failed]-> bus1_Failed;

    sens1_Failed -[MonitorStopNow]-> Fail_Stop;
    sens2_Failed -[MonitorStopNow]-> Fail_Stop;
    sens3_Failed -[MonitorStopNow]-> Fail_Stop;
    bus1_Failed -[MonitorStopNow]-> Fail_Stop;
properties
    WrongDataReading.Probability => poisson 10E-5;
    Byzantine.Probability => poisson 10E-10;
end BasicErrors.Implementation1;
```

 47 10/13/2004

AADL Model (Safety) Use of Error Annex

```
system type navigation.type1
subcomponents
    sens1: device GPSSensor.type1;
    sens2: device GPSSensor.type2;
    sens3: device GPSSensor.type3;
    bus1: bus CommunicationBus.type1;
    monitor: processor health_monitor.type1;
```

 48 10/13/2004

AADL Model (Safety) Use of Error Annex (cont.)

connections

conn1: bus access bus1 -> sens1.controller_sensor in modes (All_Working, sens2_Failed, sens3_Failed);

conn2: bus access bus1 -> sens2.controller_sensor in modes (All_Working, sens1_Failed, sens3_Failed);

conn3: bus access bus1 -> sens3.controller_sensor in modes (All_Working, sens1_Failed, sens2_Failed);

conn4: bus access bus1 -> monitor.controller_sensor in modes (All_Working, sens1_Failed, sens2_Failed, sens3_Failed);

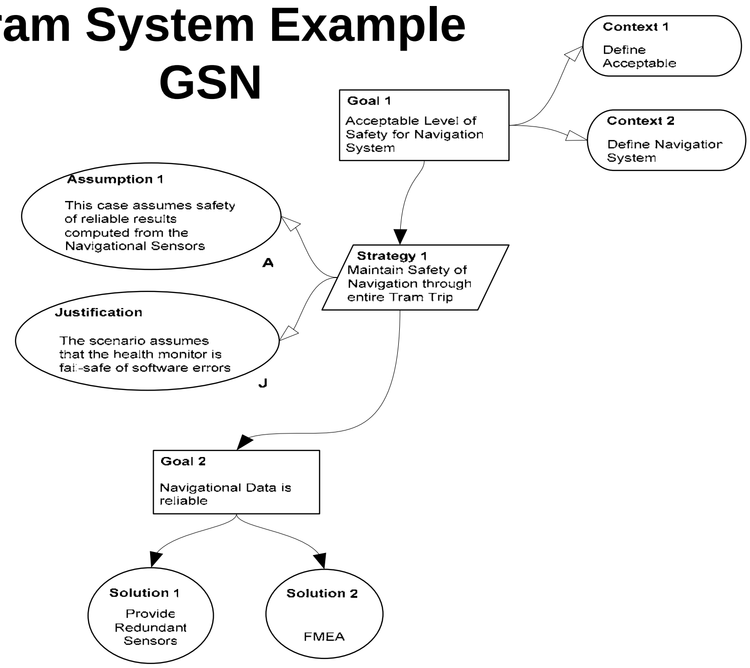
annex error {**

Model => BasicError.Implementation1;

**}

end navigation.type1;

Tram System Example GSN



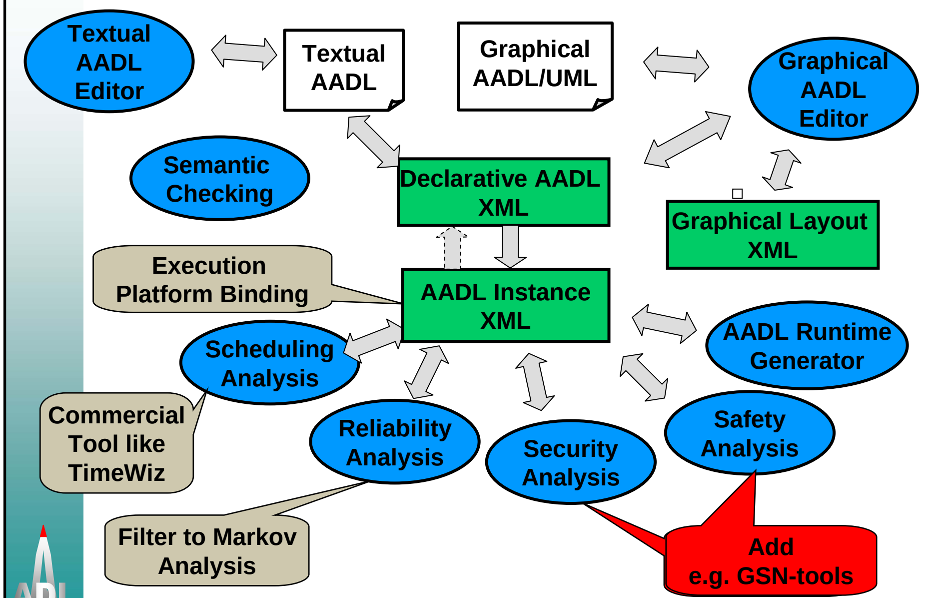
Outline

- ❑ Safe & Secure Systems
- ❑ Background of SAE AADL
- ❑ Overview of AADL Major Features & How they support Safety & Security
- ❑ Using Error Model Annex & GSN for Safety or Security
- ❑ Example using Error Model Annex and GSN
- ➡ Further Work Required
- ❑ AADL & its Relationship to SysML
- ❑ Summary

Work Needed for AADL to Fully Support Safety & Security

- ❑ Use proposed AADL Error Annex to fully define properties, features, & models for
  - Safety
  - Security
- ❑ Integrate safety & security tools in to AADL tool environment
  - Read/Write AADL-XML schema
- ❑ Augment existing AADL tools to display & edit security & safety characteristics?

XML-Based AADL Tool Strategy



(c) 2004 High Integrity Systems, Ltd. & Absolute Software Co., Inc. 53 10/13/2004

Outline

- ❑ Safe & Secure Systems
- ❑ Background of SAE AADL
- ❑ Overview of AADL Major Features & How they support Safety & Security
- ❑ Using Error Model Annex & GSN for Safety or Security
- ❑ Example using Error Model Annex and GSN
- ❑ Further Work Required
- ❑ AADL & its Relationship to SysML
- ❑ Summary

(c) 2004 High Integrity Systems, Ltd. & Absolute Software Co., Inc. 58 10/13/2004

SAE

AADL & SysML

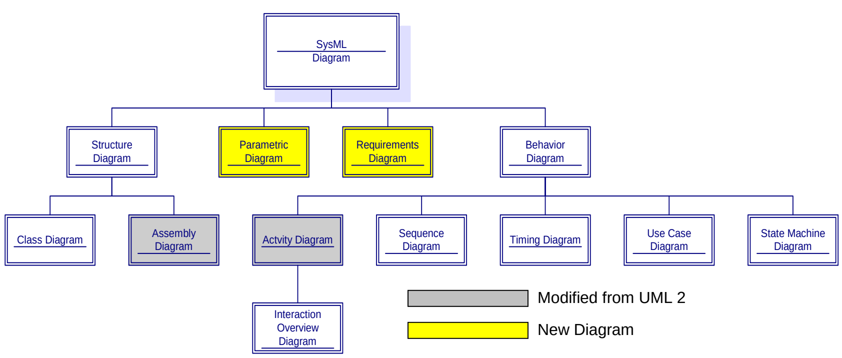
- SAE Architecture, Analysis & Design Language (AADL)
  - UML profile under review
  - Developed from top down to support system-level designs
  - Specification, architecture design, analysis
    - task & communication
  - Focused on
    - Predictable systems
    - Dynamically reconfigurable multi–processor system
  - Standard approved by SAE
- OMG Systems Modeling Language (SML) is
  - Extension of UML 2.0
  - Supports Systems Engineering
  - Extensions include:
    - Assembly Diagram
    - Activity Diagram
    - Allocations
    - Requirements Diagram
    - Parametric Diagram
    - Other Extensions
  - Draft Standard in OMG review

AADL

(c) 2004 High Integrity Systems, Ltd. & Absolute Software Co., Inc. 59 10/13/2004

SAE

SysML Diagrams



AADL

(c) 2004 High Integrity Systems, Ltd. & Absolute Software Co., Inc. 60 10/13/2004

Outline

- ❑ Safe & Secure Systems
- ❑ Background of SAE AADL
- ❑ Overview of AADL Major Features & How they support Safety & Security
- ❑ Using Error Model Annex & GSN for Safety or Security
- ❑ Example using Error Model Annex and GSN
- ❑ Further Work Required
- ❑ AADL & its Relationship to SysML
- ❑ Summary

SAE AADL Summary

- ❑ AADL is Architecture Description Language & tools for predictable systems
  - – e.g. time, reliability, fault-tolerance
  - – Based on 15 years of DARPA research
- ❑ AADL provides means to:
  - – Specify software & hardware architecture
    - • Incrementally develop from prototype to specification
  - – Analyze architecture rigorously
  - – Implement final system
    - • Integrating components with hardware & automatically generated system executive & glue code
  - – Evolve system rapidly
    - • Within development
    - • Across lifecycle

Value of AADL-Based Development

- Early prediction & verification (tool-supported)
  - Currently
    - Performance
    - Reliability
    - Fault-tolerance
- Component compliance verification (tool-supported)
  - Currently
    - Functional interface
    - Resource requirements
- System integration & verification (tool-supported)
  - Workstation testing
  - System performance

How AADL Supports Safety & Security Analyses

- Provides well-defined semantic foundation for architecture description
  - Facilitates
    - Specification of
      - Target of Evaluation (TOE) Description
      - TOE Security Environment
      - Security Objectives
        - » e.g. Identification of Assets (i.e. Shared data & data ports)
      - IT Security Requirements
      - Rationale required for Protection Profile (PP)
    - Creation of Security Target (ST) artifacts
    - Analyzing whether requirements are met
    - Developing tools for analysis

How AADL Supports Safety & Security Analyses (cont.)

- Proposed execution environment & toolset will aid verification & certification processes
  - Execution semantics formally specialized using Finite Automata Model
  - Environment enforces time & space partition specified in architecture description
  - System generator produces
    - Execution image
    - Execution-platform specific to
      - Run-time environment
      - Application-required facilities
        - » i.e. can eliminate or block access to unused environment services
    - "Glue" code to support communication between components

How AADL Supports Safety & Security Analyses (cont.)

- Proposed execution environment & toolset will aid verification & certification processes (cont.)
  - Certifying AADL environment will reduce cost of certifying systems developed for same hardware
  - Tools can verify that application code in execution image was specified in architecture
  - Reliability analysis tool(s) can verify that code is reliable enough to meet security & safety requirements

How AADL Supports Safety & Security Analyses (cont.)

- If features are need to support security or safety analysis
  - Can extend language
    - Properties
    - “Annexes”
      - e.g. for formal constraint or specification languages
  - Can define new tools

Conclusion

- Model-based system engineering benefits
  - Predictable runtime characteristics addressed early & throughout life cycle
  - Greatly reduces integration & maintenance effort
- Benefits of AADL as SAE standard
  - AADL as standard provides confidence in
    - Language stability,
    - Broad adoption,
    - Common Definitions,
    - Strong tool support
    - Extensible for safety & security

