Pyrrhus Software
Enduring Solutions

The SAE Architecture Analysis and Description Language (AADL) Standard: A Basis for Architecture-Driven Embedded Systems Engineering

DSN 2006 Workshop on Architecting Dependable Systems
(WADS)
27 June 2006
Philadelphia, PA

Joyce L Tokar, PhD
Pyrrhus Software
tokar@pyrrhusoft.com

Objectives

- *Introduce* architecture-based development concepts and benefits.
- *Provide* a summary of the SAE AADL Standard
- *Provide* a summary of AADL's capabilities.
- *Demonstrate* the benefits of AADL in real-time systems design.
- *Provide* an overview of the AADL development environment.

The SAE AADL Standard

- **Sponsored by the Society of Automotive Engineers (SAE)**
  - **Avionics Systems Division (ASD)**
    - **Embedded Systems (AS2)**
      - **Avionics Architecture Description Language Subcommittee (AS2C)**

- **Status**
  - Requirements document SAE ARD 5296 – balloted & approved in 2000.
  - Standard document SAE AS 5506 – balloted & approved in 2004.
  - Annex documents – balloted & approved in 2005.
    - Graphical Annex
    - XML Annex
    - Programming Language Annex
    - Error Annex
  - UML Annex to be balloted in 2006.

- **Coordination with**
  - NATO Aviation, NATO Plug and Play, French Government COTRE, ASSERT, SAE AS-1 Weapons Plug and Play, OMG UML

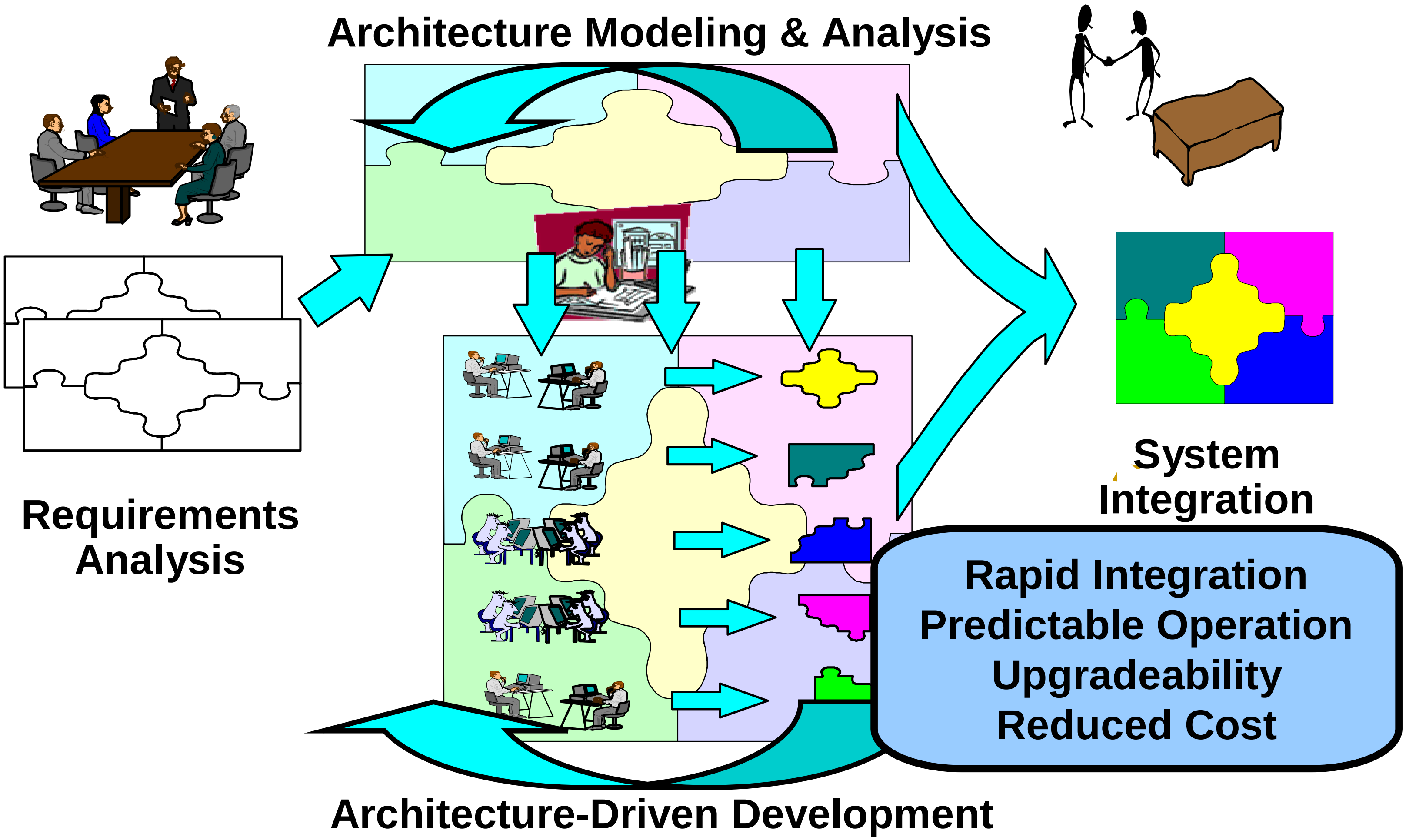
http://www.aadl.info email: info@aadl.info

SAE AS-2C AADL Subcommittee

- **Key Players:**
  - Bruce Lewis (AMCOM): Chair, technology user
  - Steve Vestal (Honeywell): MetaH originator, co-author
  - Peter Feiler (SEI):Technical lead, author, co-editor, technology user
  - Ed Colbert (USC): AADL & UML Mapping
  - Joyce Tokar (Pyrrhus Software): Programming Language Annex, co-editor
- **Members:**
  - Boeing, Rockwell, Honeywell, Lockheed Martin, Raytheon, Smith Industries, Airbus, Axlog, Dassault, EADS , High Integrity Solutions
  - NAVAir, Open Systems JTF, British MOD, US Army
  - European Space Agency
- **Coordination with:**
  - NATO Aviation, NATO Plug and Play, ESA, French Government CÔTRE, OMG-UML&SysML, SAE AS-1 Weapons Plug-n-Play

Model-Based System Engineering

Predictive Analysis Early In & Throughout Life Cycle



What is Architecture?

- Architecture is the fundamental organization of a system as embodied in
  - its components,
  - their relationships to each other and the environment,
  - the principles governing its design and evolution.

- The architecture of a program or computing system is
  - the structure or structural arrangements of its composite elements, both hardware and software,
  - the externally visible properties of those elements,
  - the relationships among them.

Architecture is the foundation of good software & systems engineering

What is an Architecture Description Language (ADL)?

- The *architecture* of a system defines its high-level structure and exposes its gross organization as a collection of interacting components.
- An *Architecture Description Language (ADL)* focuses on the high-level structure of the overall application rather than on the implementation details of any specific component.
- ADLs and their accompanying toolsets support architecture-based development, formal modeling, and analysis of architectural specifications.
- The *AADL* is an architecture description language that includes support for the inclusion of both the software components and the execution platform components in the system architectural specification.

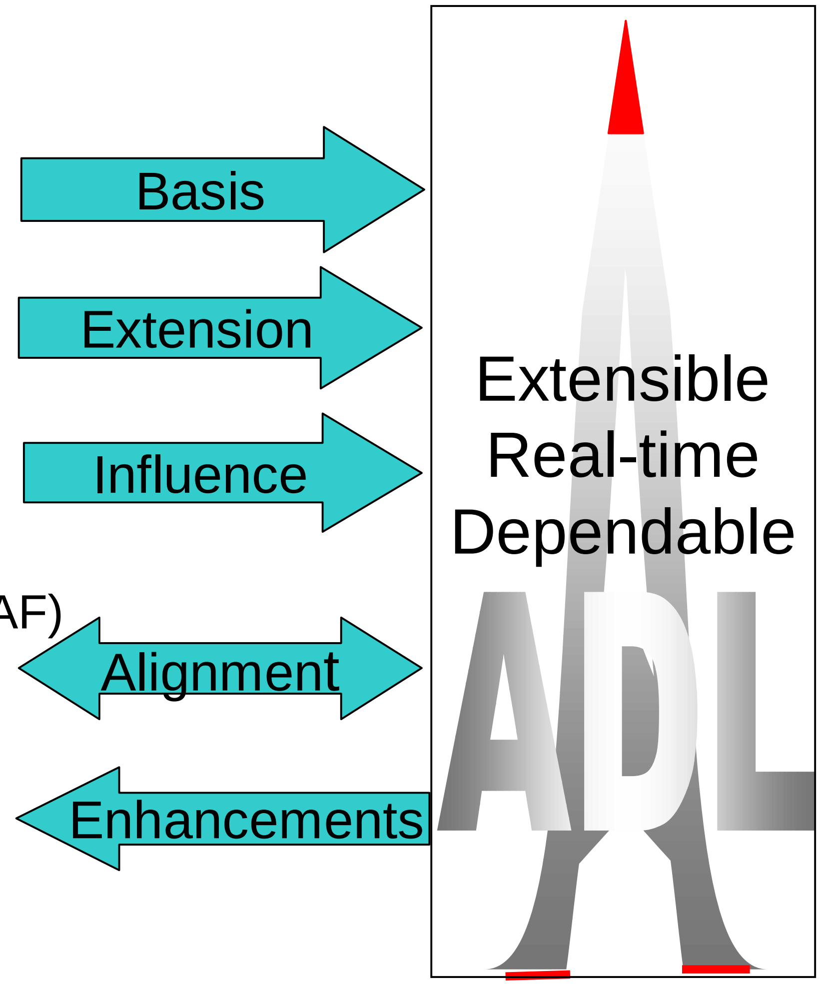
Architecture Description Languages

Research ADLs

- MetaH
  - Real-time, modal, system family
  - Analysis & generation
  - RMA based scheduling
- Rapide, Wright, ..
  - Behavioral validation
- ADL Interchange
  - Acme, xADL
  - ADML (MCC/Open Group, TOGAF)

Industrial Strength

- HOOD/Stood
- SDL
- UML 2.0, UML-RT



The SAE Architecture Analysis and Design Language (AADL)

- A language for
  - abstract and precise description of real time, performance critical architectures including both hardware and software components.
  - incrementally integrating multiple dimensions of analysis (time, safety, dependability, schedulability, utilization, fault tolerance etc) through component properties for system engineering analysis.
  - taking a specification of the architecture and using it to auto-integrate a compliant system from compliant components.

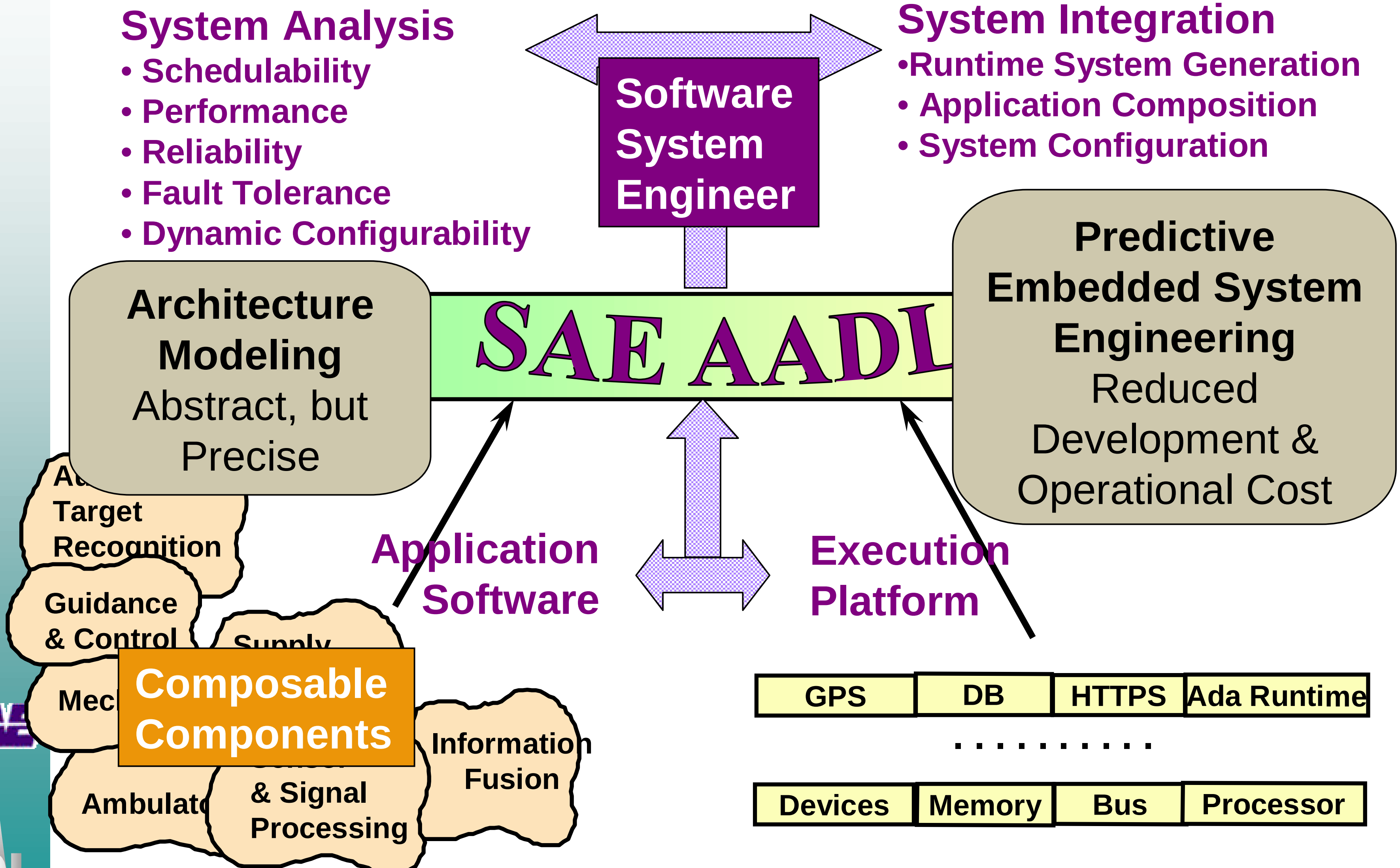
SAE Architecture & Analysis Description Language (AADL)

- Specification of
  - Real-time
  - Embedded
  - Fault-tolerant
  - Securely partitioned
  - Modal & dynamically configurable
- Software task and communication architectures
- Bound to
  - Distributed multiple processor hardware architectures
- Fields of application
  - Avionics, Aerospace, Automotive, Autonomous systems, …

AADL-Based System Engineering

System Analysis
- Schedulability
- Performance
- Reliability
- Fault Tolerance
- Dynamic Configurability

System Integration
- Runtime System Generation
- Application Composition
- System Configuration



AADL: The Language

- The AADL defines standard ***categories*** of ***components:***
  - Software: data, subprogram, thread, thread group, process
  - Execution platform: device, memory, bus, processor
  - Composite: System
- A ***connection*** between component ***ports*** declares a ***flow*** of control and/or data between components.
  - Ports: data, event, event data
  - Connections: port-to-port, subprogram calls.
- The relationship between software and execution platform components is represented through the use of ***bindings***.

AADL: The Language

- ***Property associations*** are used to constrain the model, for example, the legal and required bindings, but bindings need not be completely and explicitly declared by the developer.
- A component may have an ***implementation***, an internal sub-architecture declared as a set of connected sub-components.
- A ***package*** provides a way to organize components and port group types into a related sets of declarations.
- ***Modes*** may be used to model transition between statically known states & configurations.

AADL: The Language

- ***Component Type*** -- specifies the interface to the component.

- ***Component Implementation*** -- zero or more specifications of the component's internal representation.

AADL: The Language

Components with precise semantics

- Thread, thread group, process, system, processor, device, memory, bus, data, subprogram

Completely defined interfaces & interactions

- Data & event flow, synchronous call/return, shared data access
- End-to-end flow specification

Real-time Task Scheduling

- Supports different scheduling protocols including GRMA, EDF
- Defines scheduling properties and execution semantics

Modal, reconfigurable systems

- Mode to mode transition between statically known states & configurations

Component evolution & large scale development support

- Inheritance for types and implementations
- Component packages provide subcontractor support

Language extensibility

- Standard typing sublanguage for user defined types
- User/vendor/industry/standard Annex sublanguages

Multiple Viewpoints of SAE AADL

- ***Component View***
  - Model of system composition & hierarchy.
  - Well-defined component interfaces.
- ***Concurrency & Interaction View***
  - Time ordering of data, messages, and events.
  - Dynamic operational behavior.
  - Explicit interaction paths & protocols.
- ***Execution View***
  - Execution platform as resources.
  - Specification & analysis of runtime properties
    - timeliness, throughput, reliability, graceful degradation, ...
  - Binding of application software.
- ***User-defined View***
  - Analysis-oriented.
- ***Logical View***
  - Specification of relationships between software and execution platform components.

Primary target was the concepts and viewpoints associated with an operational system.

Pyrrhus Software
Enduring Solutions

The AADL in a Nutshell

REUSABLE
Very portable.
Function/non-functional requirements.
Ideal isolation from hardware.

EXTENSIBLE/ SCALABLE
Multi-processor/multi-process, easily add/change and see effects. User defined domain specific functions..

FLEXIBLE
System spec used to change implementation.
Interface with any standard or application

GENERIC
Modular, scalable, system "block diagram" with semantics

OBJECT-ORIENTED
Clearly defined object, messaging, properties, decomposition

System Architecture

Performance-Critical

Application
Thread, Process, System
Execution Platform
Execution engine
Memory, Bus
Device

Layering & Composition

Components
Specifications
Variant implementations
Ports
Connections
Domain data objects
Behaviors

Implementation

OPEN
Gov usage rights.
Industry AADL standard.

REAL-TIME
User specifies timing requirements, analyzers available, concurrency handled automatically!

USABLE AND AVAILABLE
Approach/formalism is *SIMPLE/UNIFORM, PRACTICAL, and EASY TO USE, LEARN, AND INTERFACE WITH OTHER APPROACHES*!

RELIABILITY, SAFETY, SECURITY SUPPORT
User specifies requirements; analyzers available

HARDWARE
MODELING AND BINDINGS FULLY SUPPORTED BY AADL(auxiliary to the SW API)

HARDWARE INDEPENDENT
No implementation specified in SW API.

VERIFIABLE
Strong support for predictable real-time architectures exhibiting high-reliability

FORMAL, RICH SEMANTICS
Models can span high-level system to detailed interfaces

The SAE AADL Standard

- Provides a standard & precise way to describe the architecture of embedded computer systems.
- Provides a standard way to describe components, assemblies of components, and interfaces to components.
- Describes how components are composed together to form complete system architectures.
- Describes the runtime semantics and thread scheduling protocols.
- Describes the mechanisms to exchange control and data between components.
- Describes dynamic run-time configurations.

AADL: The Language

System Scheduling

- Supports different scheduling protocols including Rate Monotonic Analysis (RMA), Earliest Deadline First (EDF), user-defined
- Defines scheduling properties and execution semantics
- Hardware and Software binding constraints support system optimization, product-lines, safety

Scaleable

- From software subprogram
- To hardware and software System of Systems

Component evolution & large scale development support

- Inheritance for types and implementations
- Component packages provide subcontractor support

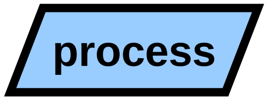
AADL language extensibility

- Standard typing sublanguage for user defined types
- User/vendor/industry/standard Annex sublanguages

Application Components

- System: hierarchical organization of components

- Process: protected virtual address space



- Thread group: organization of threads in processes

- Thread: a schedulable unit of concurrent execution

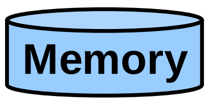
- Data: potentially sharable data

- Subprogram: Callable unit of sequential code

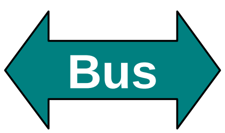
Execution Platform Components

- Processor – Provides thread scheduling and execution services

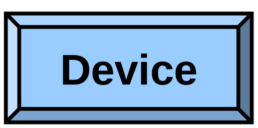
- Memory – provides storage for data and source code



- Bus – provides physical connectivity between execution platform components

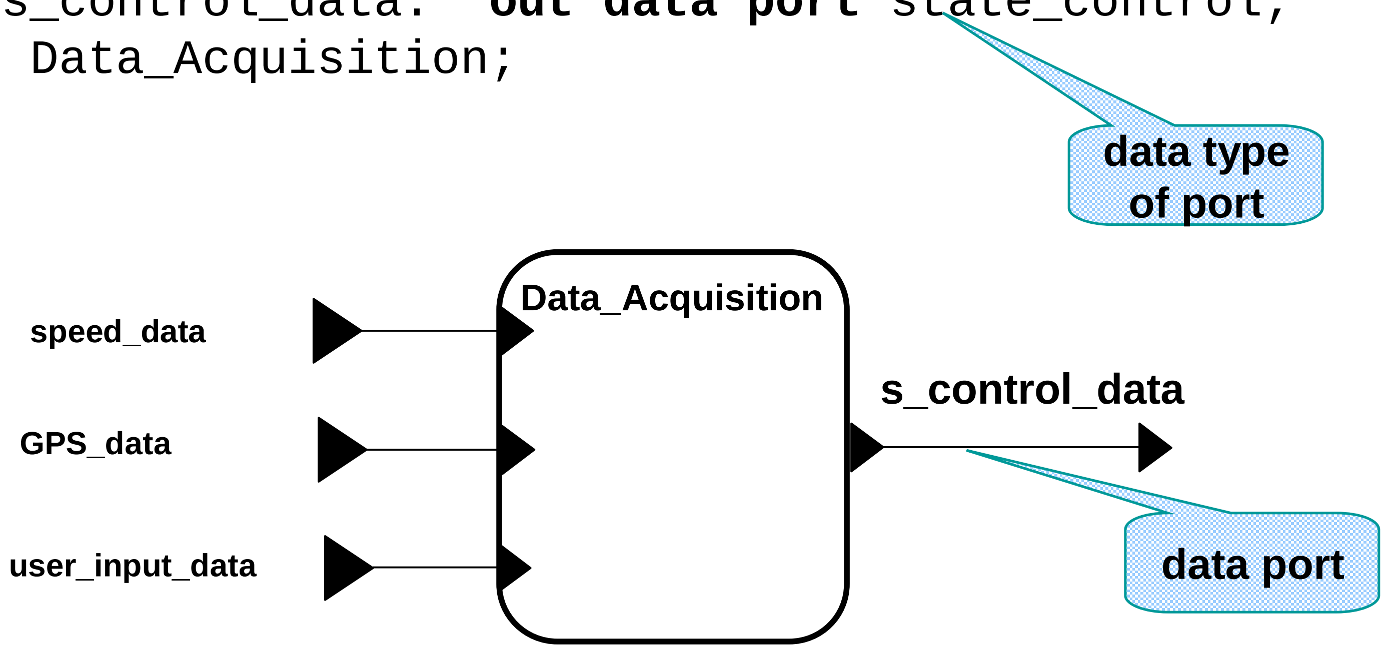


- Device – interface to external environment

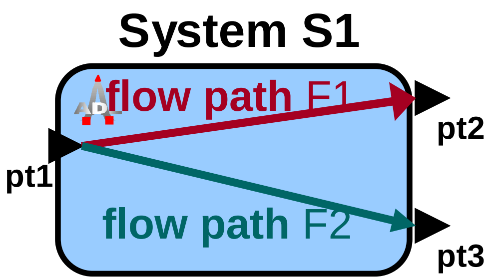


Graphical & Textual Notation

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



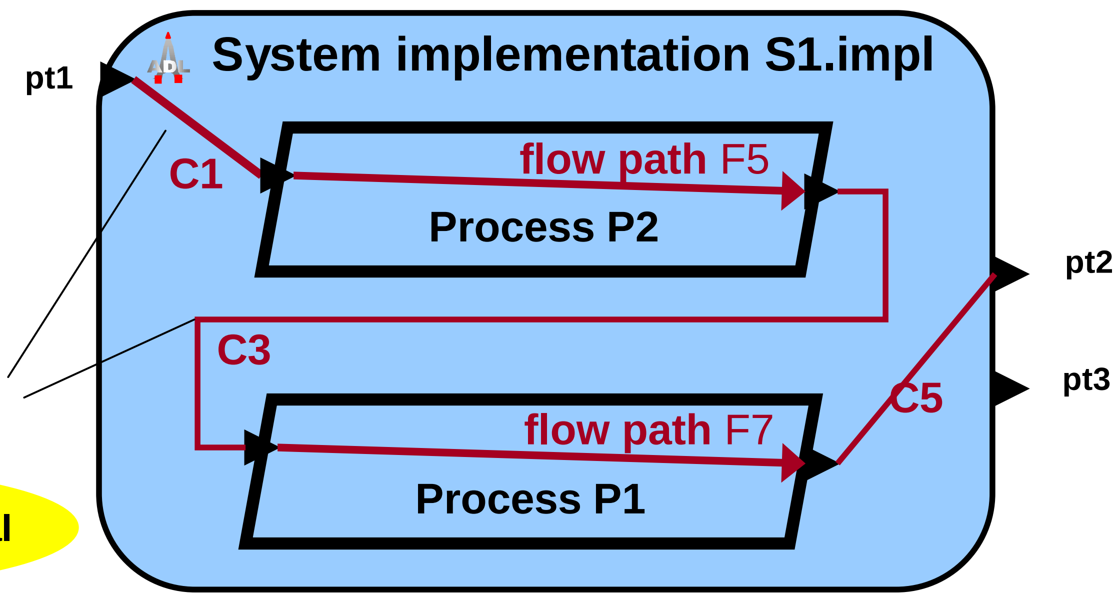
Flow Specification in AADL



Flow Specification

flow path F1: pt1 -> pt2
flow path F2: pt1 -> pt3

Flows are logical



Connection

Connections are physical

Flow Implementation

flow path F1: pt1 -> C1 -> P2.F5 -> C3 -> P1.F7 -> C5 -> pt2

Faults and Modes

- AADL provides a fault handling framework with precisely defined actions.
- AADL supports runtime changes to task & communication configurations.
- AADL defines timing semantics for task coordination on mode switching.
- AADL supports specification of mode transition actions.
- System initialization & termination are explicitly modeled.
- Error Annex provides support for error models and analysis.

An Avionics System Case Study

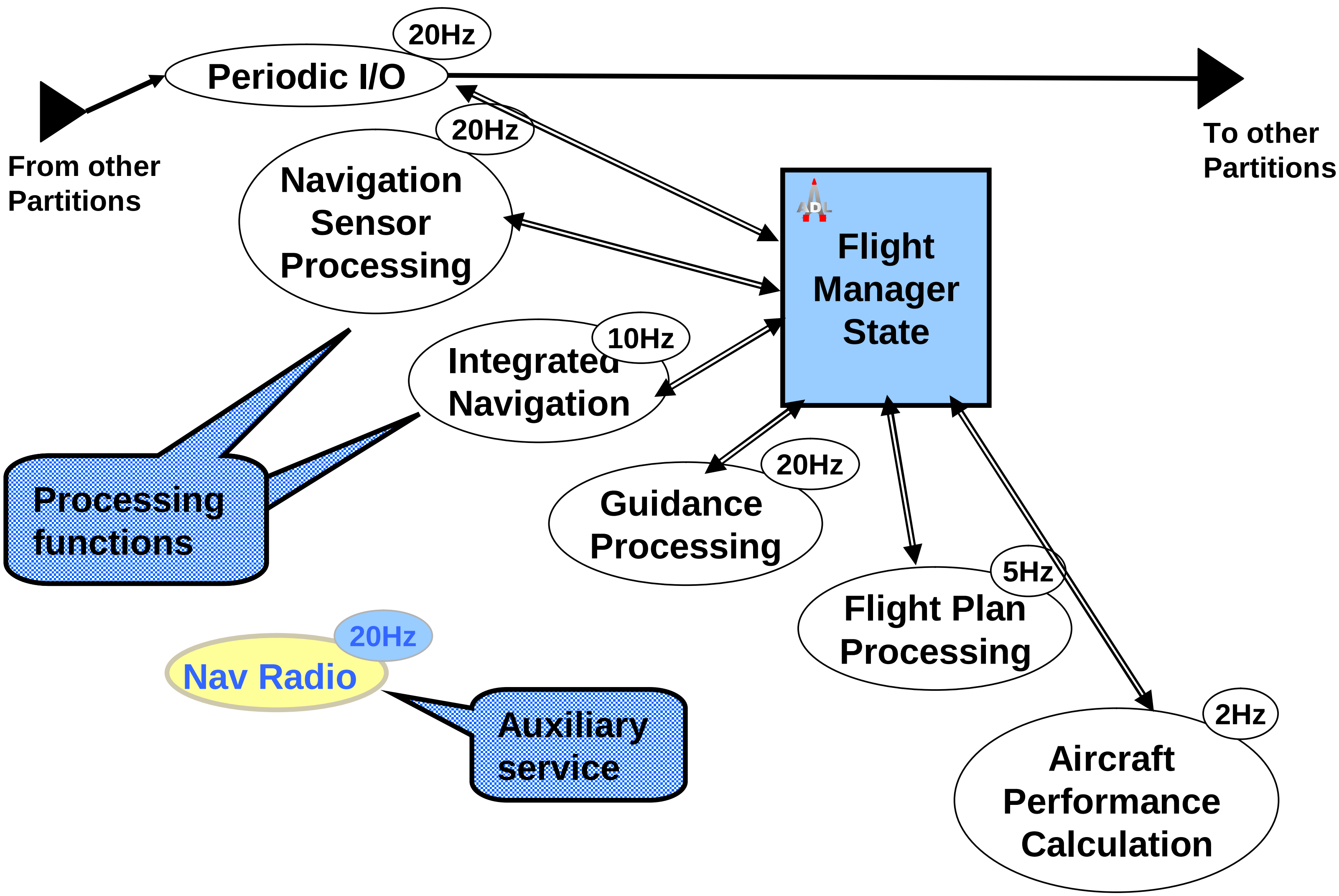
- Migration from static timeline to preemptive scheduling
- Towards distributed partitioned system
- Software & hardware redundancy
- Access to detailed design & performance data

- Pattern-based analysis of architecture
  - Abstract, but precise architecture models
  - Identify potentially systemic issues
- High-fidelity analysis of network workload
  - Model generated from design data
  - Tool-based analysis of full-scale model

System Timing Concerns

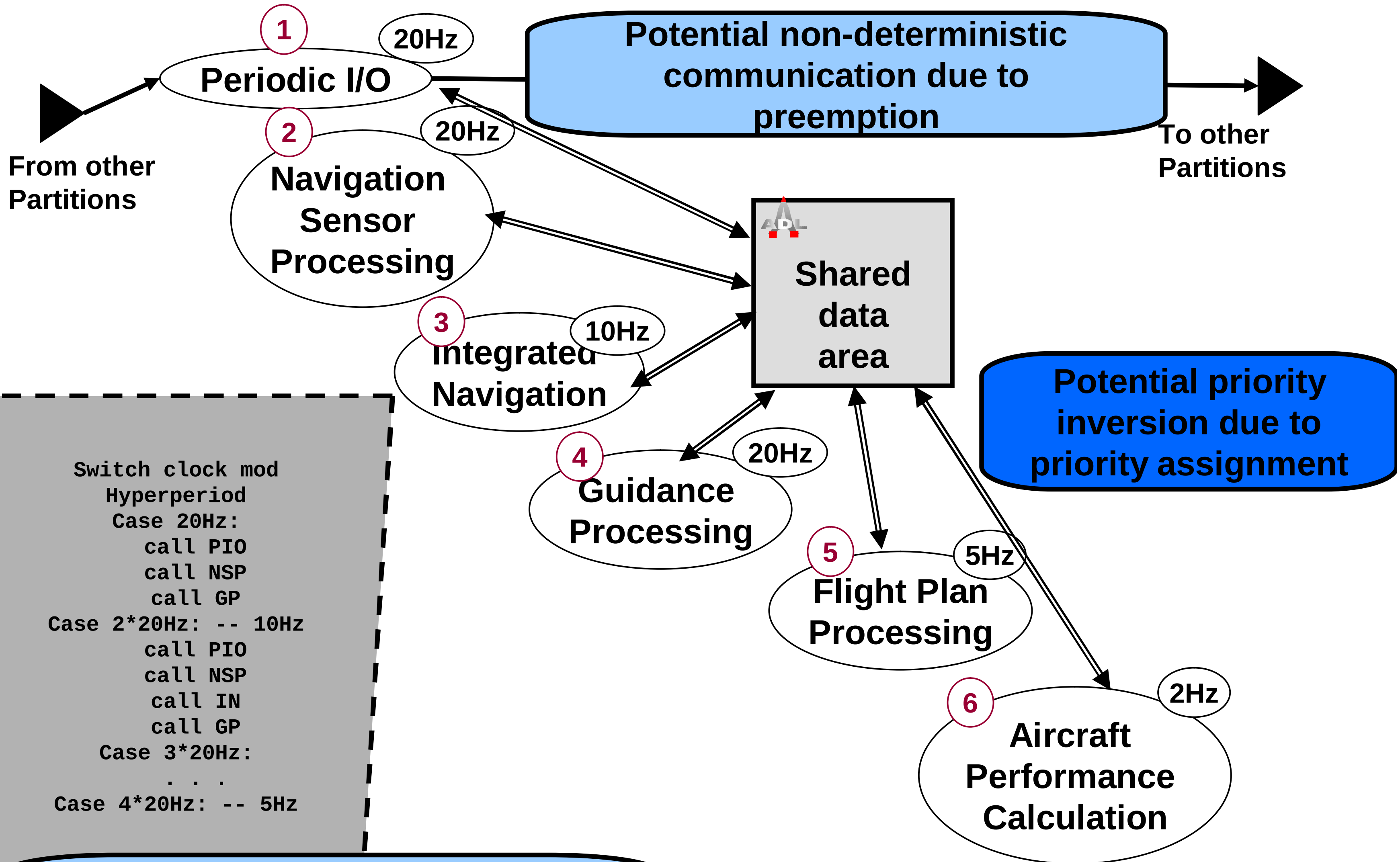
- Critical flows: application perspective
  - Unqueued data streams, event streams, queued message streams
  - Sampling of stream, throttling of processing
  - End-to-end latency, throughput
  - Variability & upper bounds
  - Hybrid control systems & modal operation
- Critical flows: embedded software perspective
  - Periodic & aperiodic threads
  - Efficient communication mechanisms
  - Time & space partitioning
  - Schedulability of processor & buses/networks
  - Hybrid & modal task architectures

Flight Manager: Principal Functionality



Pyrrhus Software
Enduring Solutions

A Cyclic Executive Implementation




```
Switch clock mod
Hyperperiod
Case 20Hz:
    call PIO
    call NSP
    call GP
Case 2*20Hz: -- 10Hz
    call PIO
    call NSP
    call IN
    call GP
Case 3*20Hz:
    . . .
Case 4*20Hz: -- 5Hz
```

Tasks must complete within frame => cyclic executive behavior

Priority Inversion Checker

- Analysis of AADL models
  - User assigned priorities
    - Modeled as new property
  - Potential red flag
    - Recording & reporting of analysis results

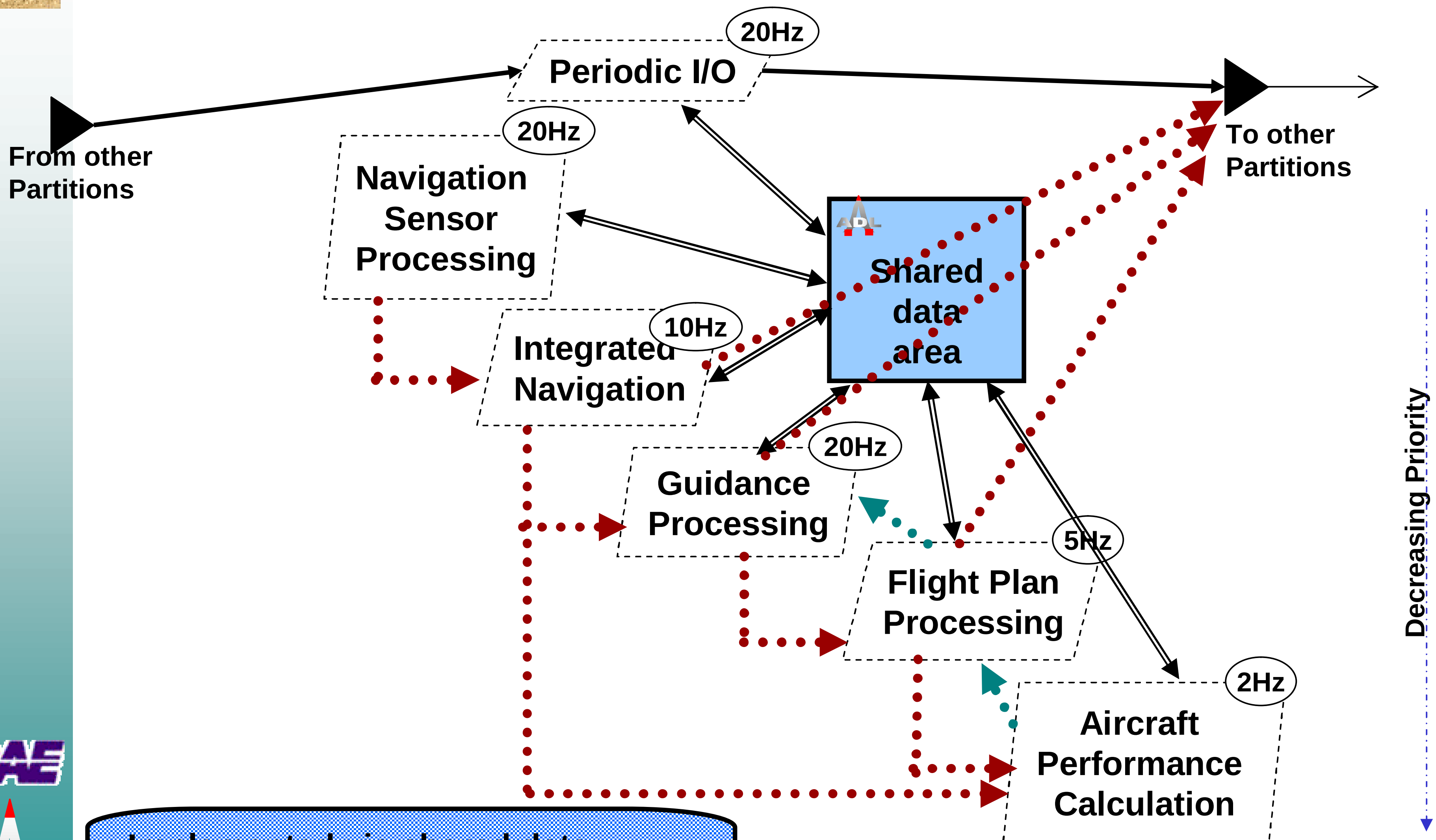
Potential priority inversion identifiable by analysis tool

- Tool support
  - Checker operates on system instance bound to execution platform
  - External tool processes XML document
  - Plug-in to Open Source AADL Tool Environment

Non-deterministic Phase Delay

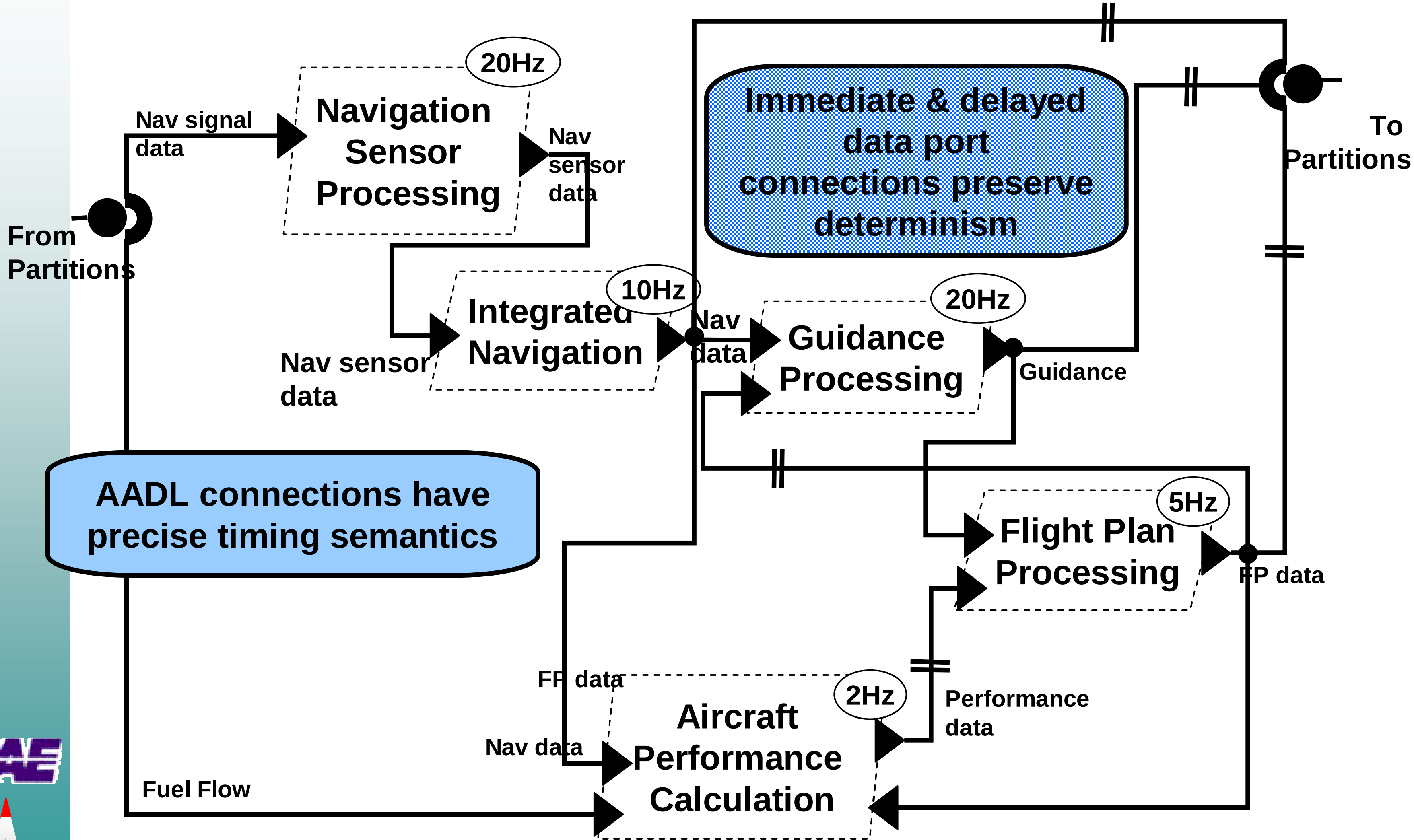
- Variable phase delay of data elements
  - Variable timing of user-level send/receive calls
  - Variable send/receive ordering due to preemption
  - Results in variable frame delay of data element
- Does it matter?
  - Data stream as controller input
    - Latency jitter viewed as noise in data stream
    - Software induced jitter engineered away
  - Data stream as display output
    - Phase delay oscillation results in blurred display
  - Time stamping of data elements
    - Time synchronization of data streams

Intended Data Flow



Implemented via shared data
Achieved via precedence ordering

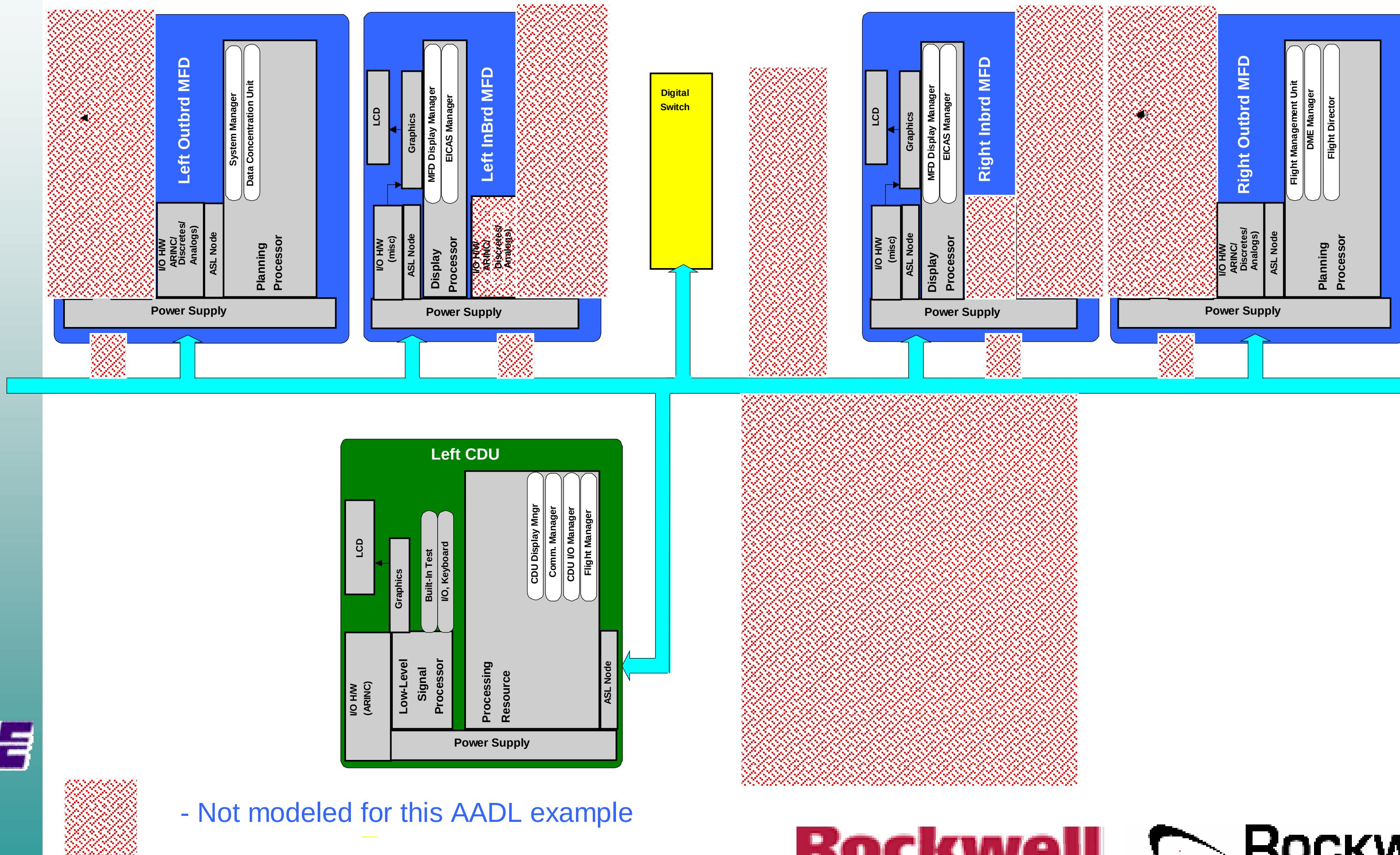
Flight Manager in AADL



Pyrrhus Software

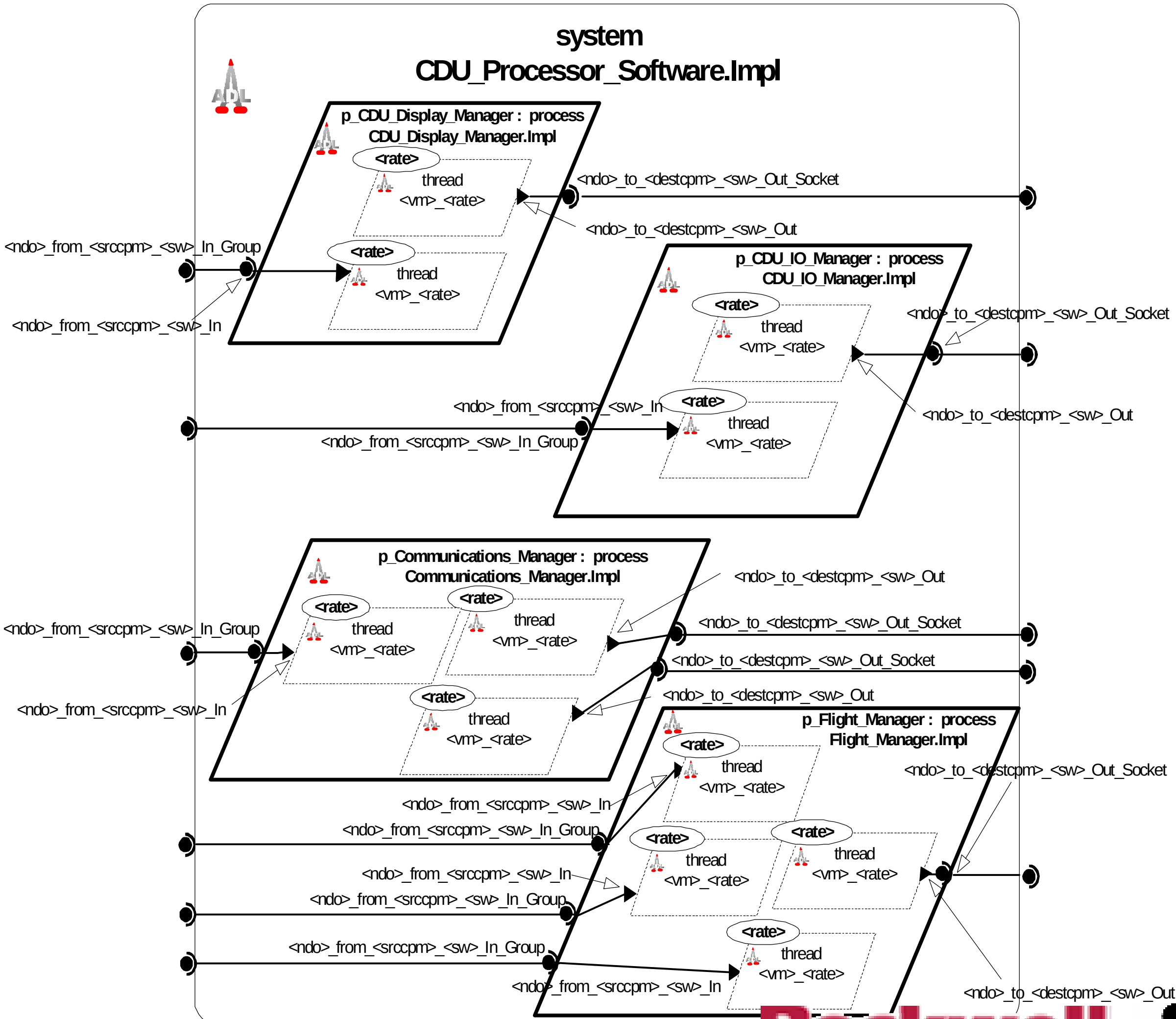
Enduring Solutions

Analyzable and Reconfigurable AADL Specifications for IMA System Integration



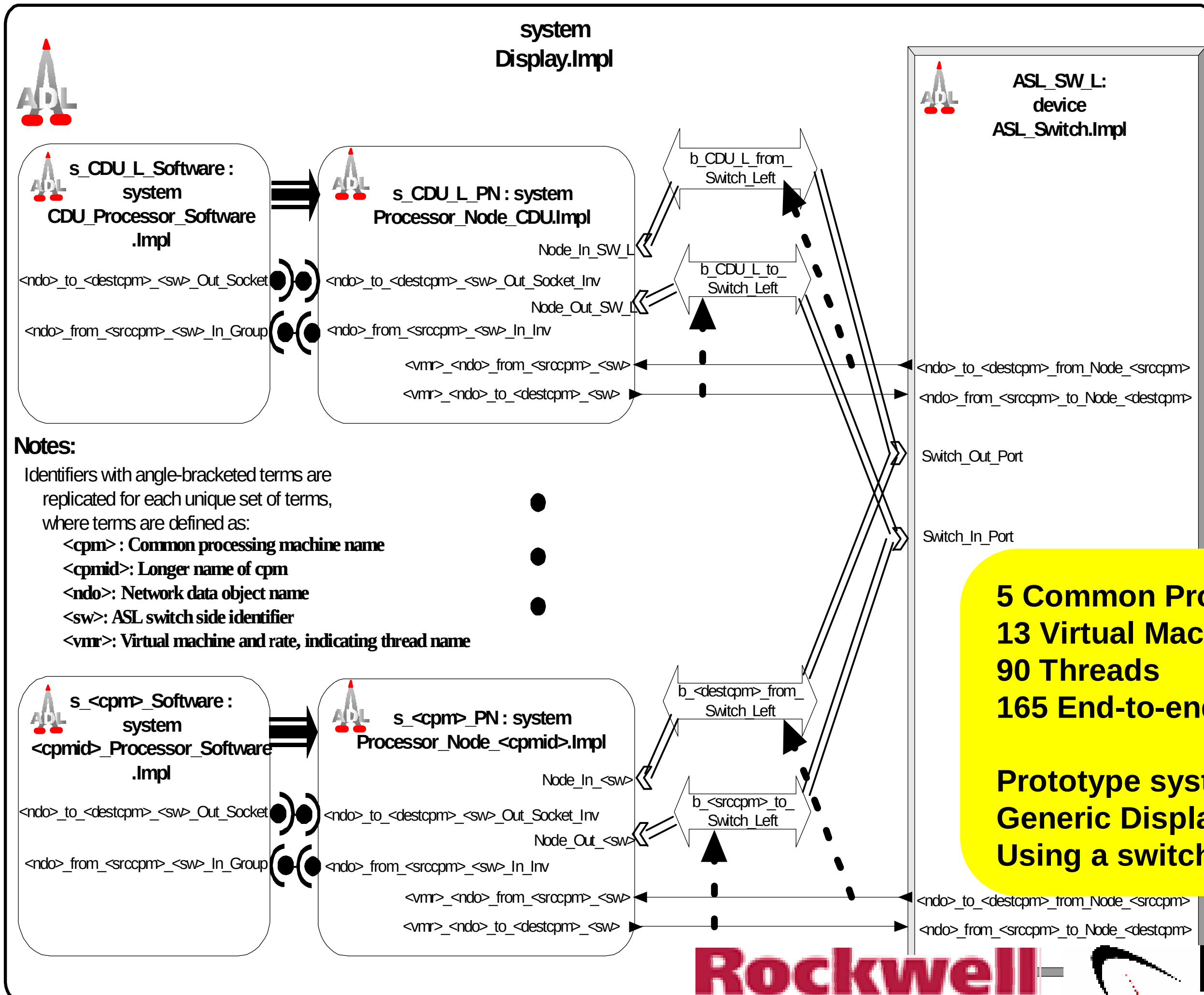
- Not modeled for this AADL example

Graphical Software (Logical) View



Pyrrhus Software
Enduring Solutions

Overall System Integration



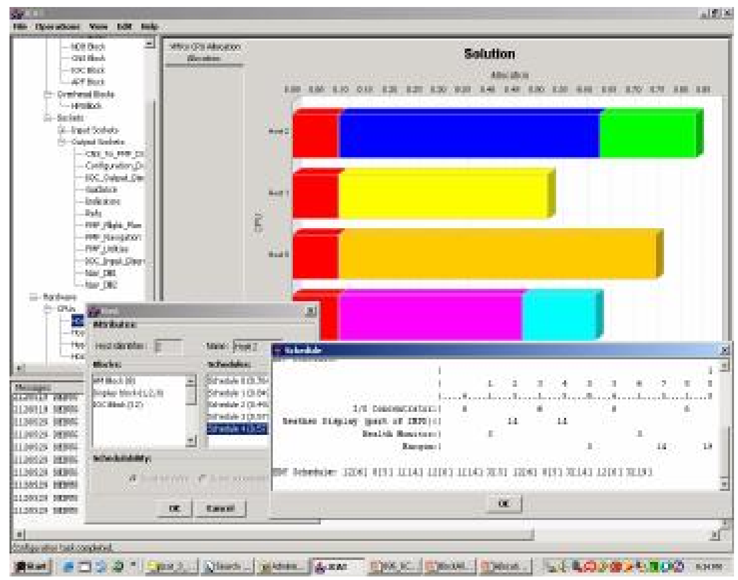
5 Common Processing Modules
13 Virtual Machines
90 Threads
165 End-to-end Data Flows

Prototype system of a
Generic Display System
Using a switched ethernet LAN.

Rockwell Collins — Rockwell Scientific

Analysis and Reconfiguration Tool

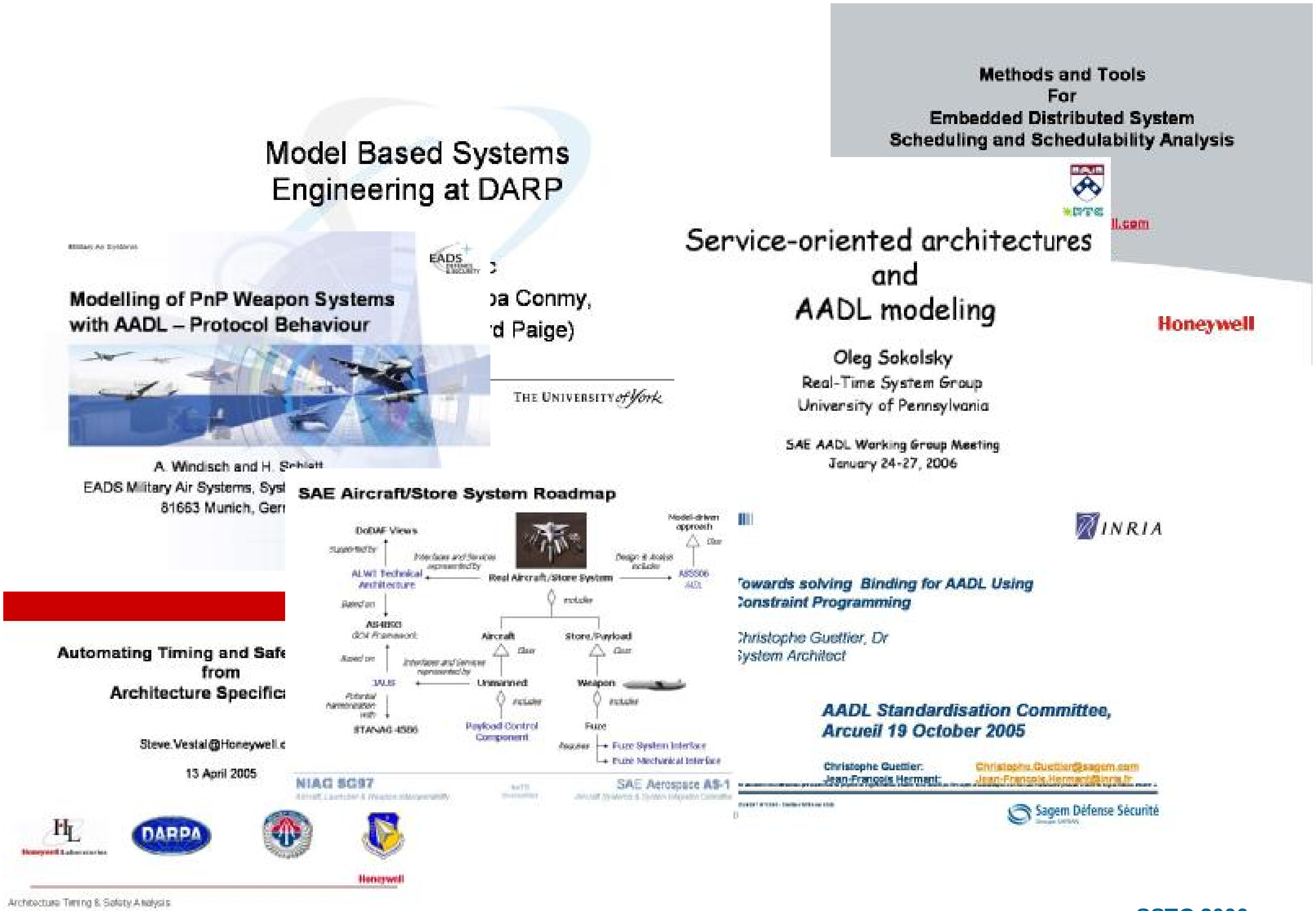
- System generation from Translated XML AADL
  - § Automatic schedule generation
  - § Allocation of VMs to hosts
- System analysis
  - § Schedulability, rate-monotonic analysis
  - § Network analysis
- Editing and visualization
  - § Direct manipulation, tree view
  - § Graphs, tables, trade studies



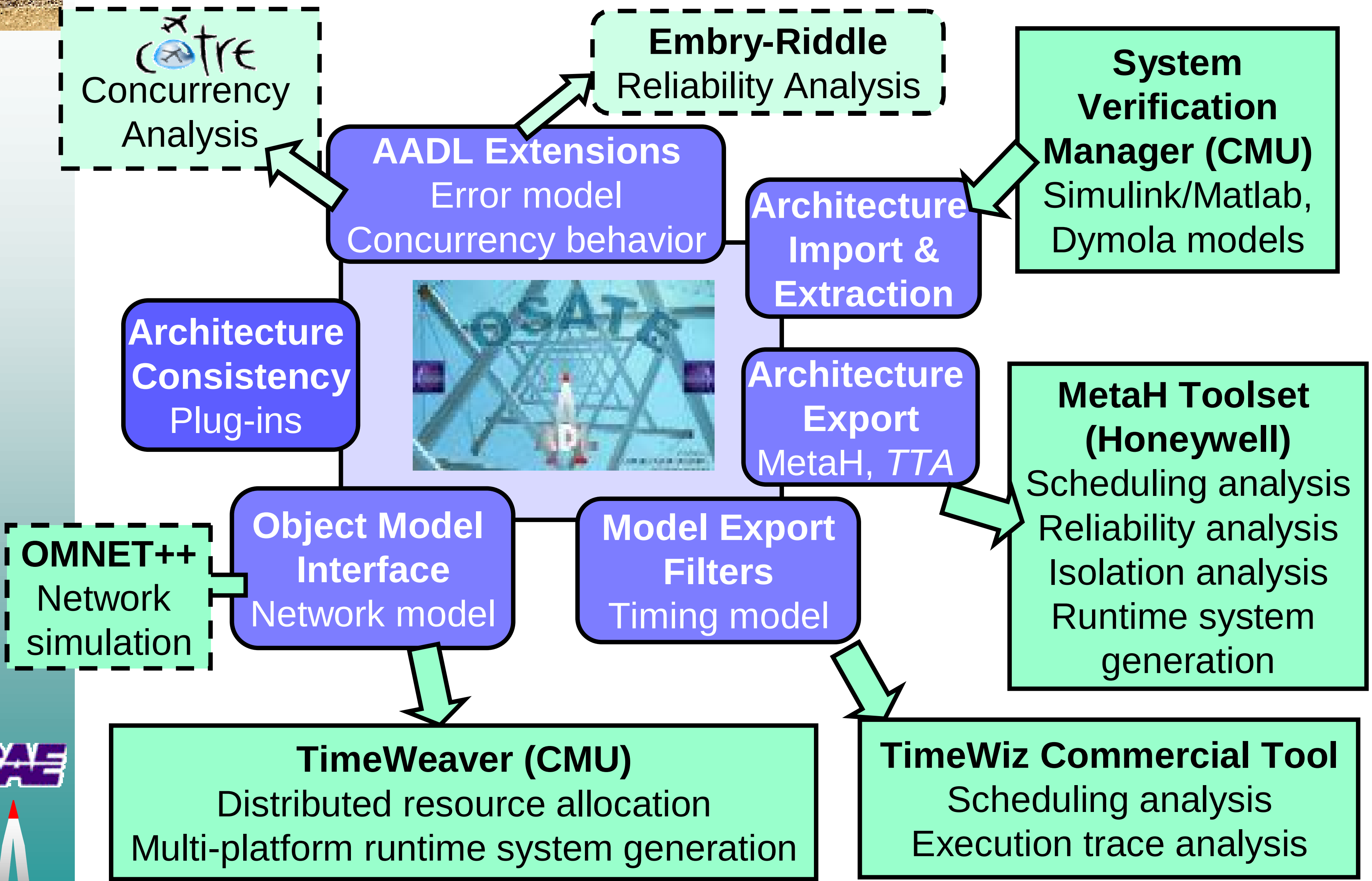
Rockwell Collins

Rockwell Scientific

Some Other AADL Users



An Extensible Engineering Environment



Benefits of Model-Driven Development

The **SAE AADL**
– as an industry standard –
provides a stable common framework for contractors to cooperatively evolve large-scale systems and for tool vendors to provide tools for a common architecture representation.

Reduction in errors in the final system through early analysis and automatic system generation.

Acronyms

- AADL – Architecture Analysis and Description Language
- ADL – Architecture Description Language
- ADML – Architecture Description Markup Language
- ASD – Avionics Systems Division
- AS2 – ASD Embedded Systems Subcommittee
- AS2C – ASD AS2 Avionics Architecture Description Language Subcommittee
- CMU – Carnegie Mellon University
- EDF – Earliest D
- HOOD – H
- IMA – Inte
- LAN – Loc
- MCC/Ope
  technolog
- OSATE – O
- RMA – Rate Monotonic Analysis
- SAE – Society of Automotive Engineers
- SDL – Specification and Description Language
- SEI – Software Engineering Institute
- STOOD -- S Object Oriented Design
- TOGAF – The Open Group Architecture Framework
- TTA – Time Triggered Architecture
- UML – Unified Modeling Language
- xADL – Highly Extensible Architecture Description Language
- XML – Extensible Markup Language

Questions?