The Software Engineering Institute

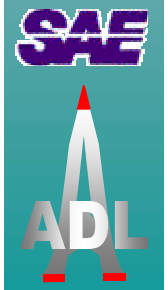
Pattern-Based Analysis of an Embedded Real-Time System Architecture

Peter Feiler

Software Engineering Institute

phf@sei.cmu.edu

412-268-7790



The Software Engineering Institute

Outline

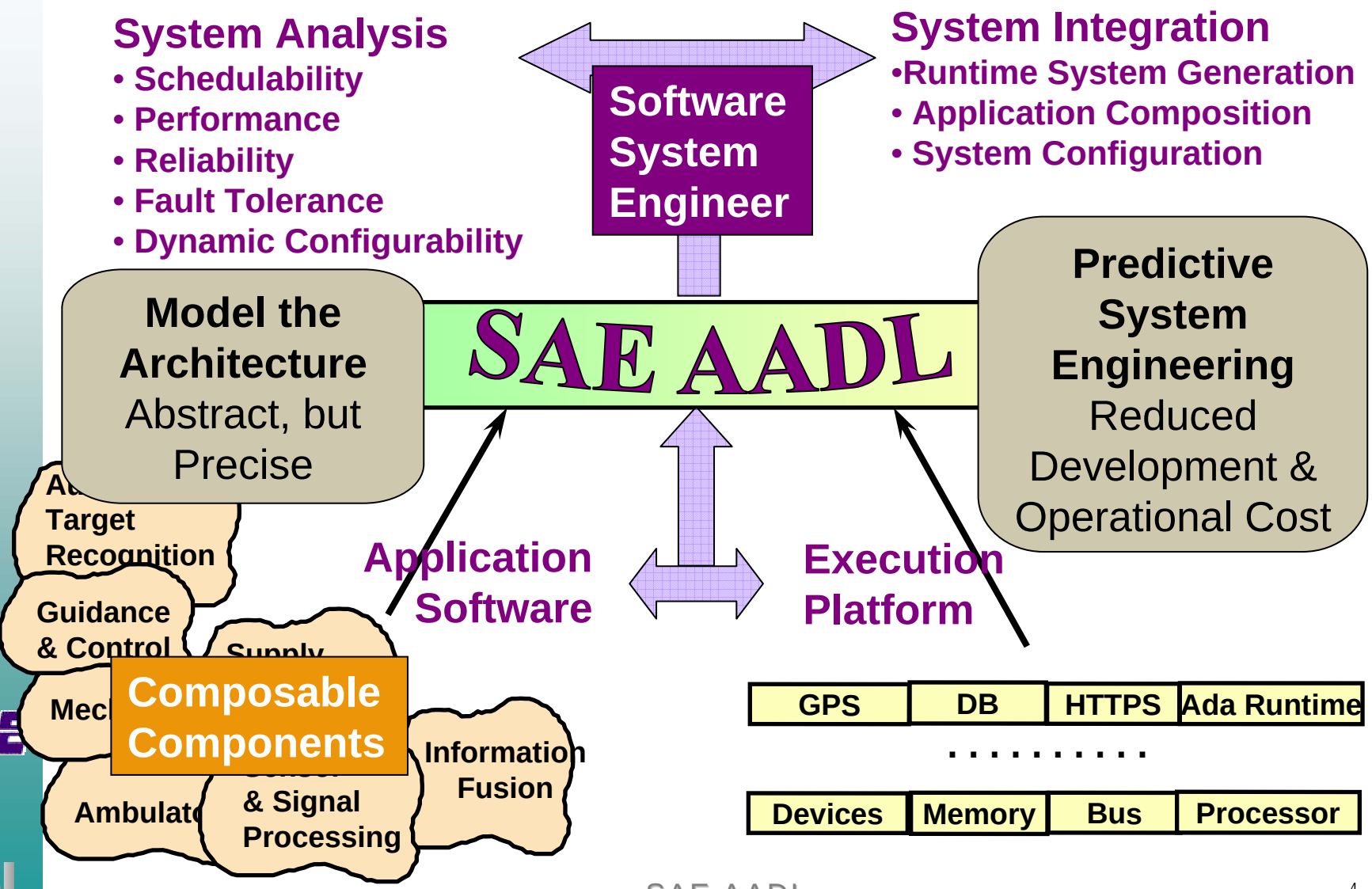
- Introduction to SAE AADL Standard
- The case study
- Towards preemptive scheduling
- Partition scheduling
- End-to-end flows
- System redundancy

The Software Engineering Institute

SAE Architecture Analysis & Design Language

- Notation for specification of task and communication architectures of Real-time, Embedded, Fault-tolerant, Secure, Safety-critical, Software-intensive systems
- Fields of application: Avionics, Automotive, Aerospace, Autonomous systems, …
- Based on 15 Years of DARPA funded technologies
- Standard approved by working group in July 2004
- www.aadl.info

AADL-Based System Engineering



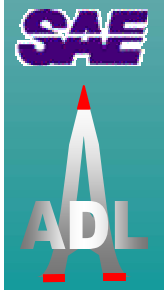
Outline

- Introduction to SAE AADL Standard
- The case study
- Towards preemptive scheduling
- Partition scheduling
- End-to-end flows
- System redundancy

The Software Engineering Institute

AADL-Based Pattern Analysis

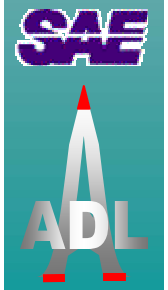
- SAE AADL employs
  - Components with precisely defined execution semantics
  - Explicit component interactions
  - Separation of concerns
- Pattern-based architecture analysis approach
  - Uses design patterns in analysis
  - Identifies systemic problems early
  - Enables the right choices with confidence
  - Provides analysis-based decisions



The Software Engineering Institute

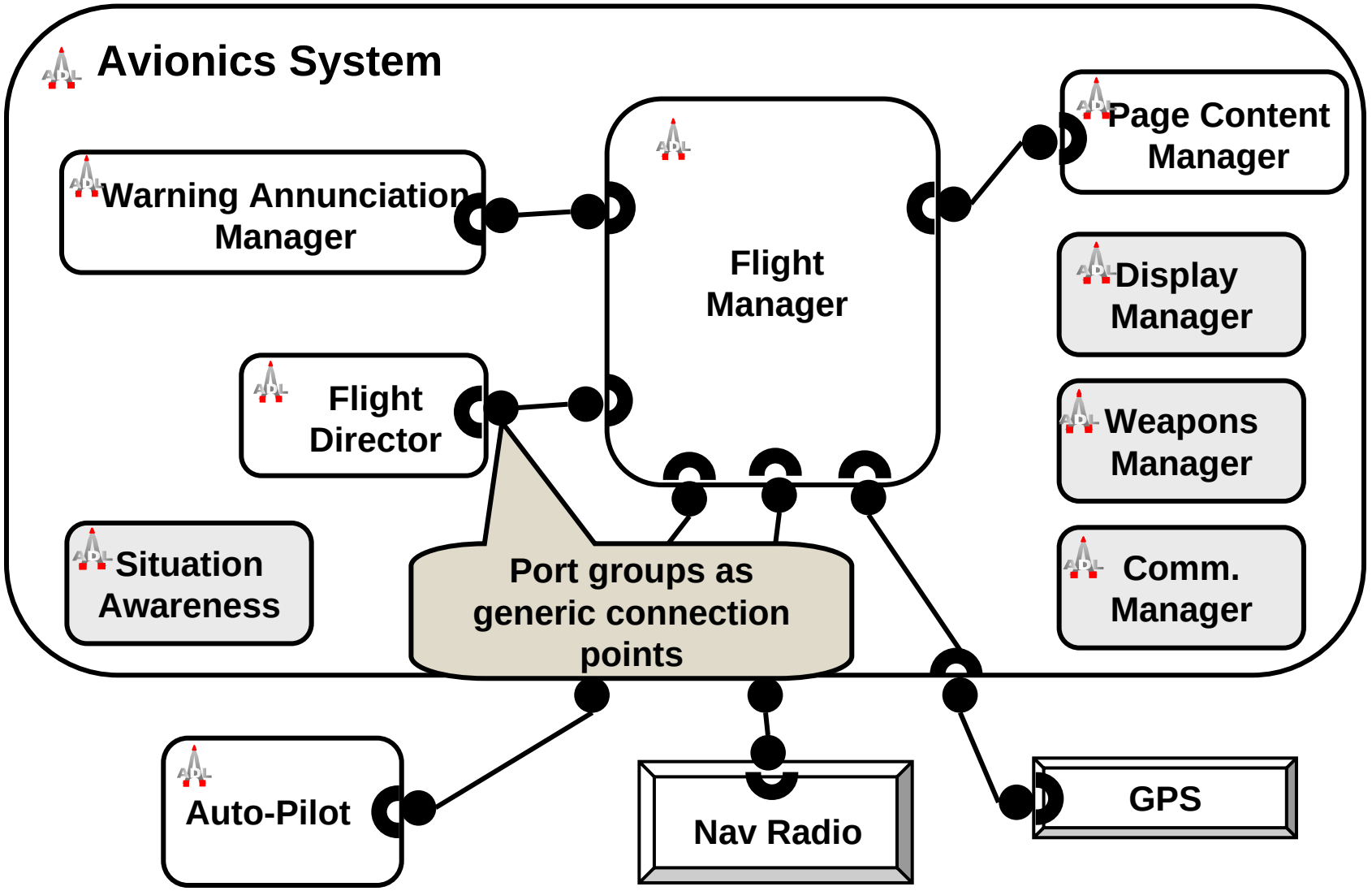
Avionics Systems

- Embedded avionics system designs are evolving to
  - From federated to integrated systems
  - From static timelines to predictable preemptive scheduling
  - Deterministic signal stream processing
  - Efficient execution and footprint
  - Fault tolerance & reconfiguration
  - Towards extensible system architectures
- There are distinct perspectives in the design
  - control and domain engineers
  - application software engineers
  - system software engineers



The Software Engineering Institute

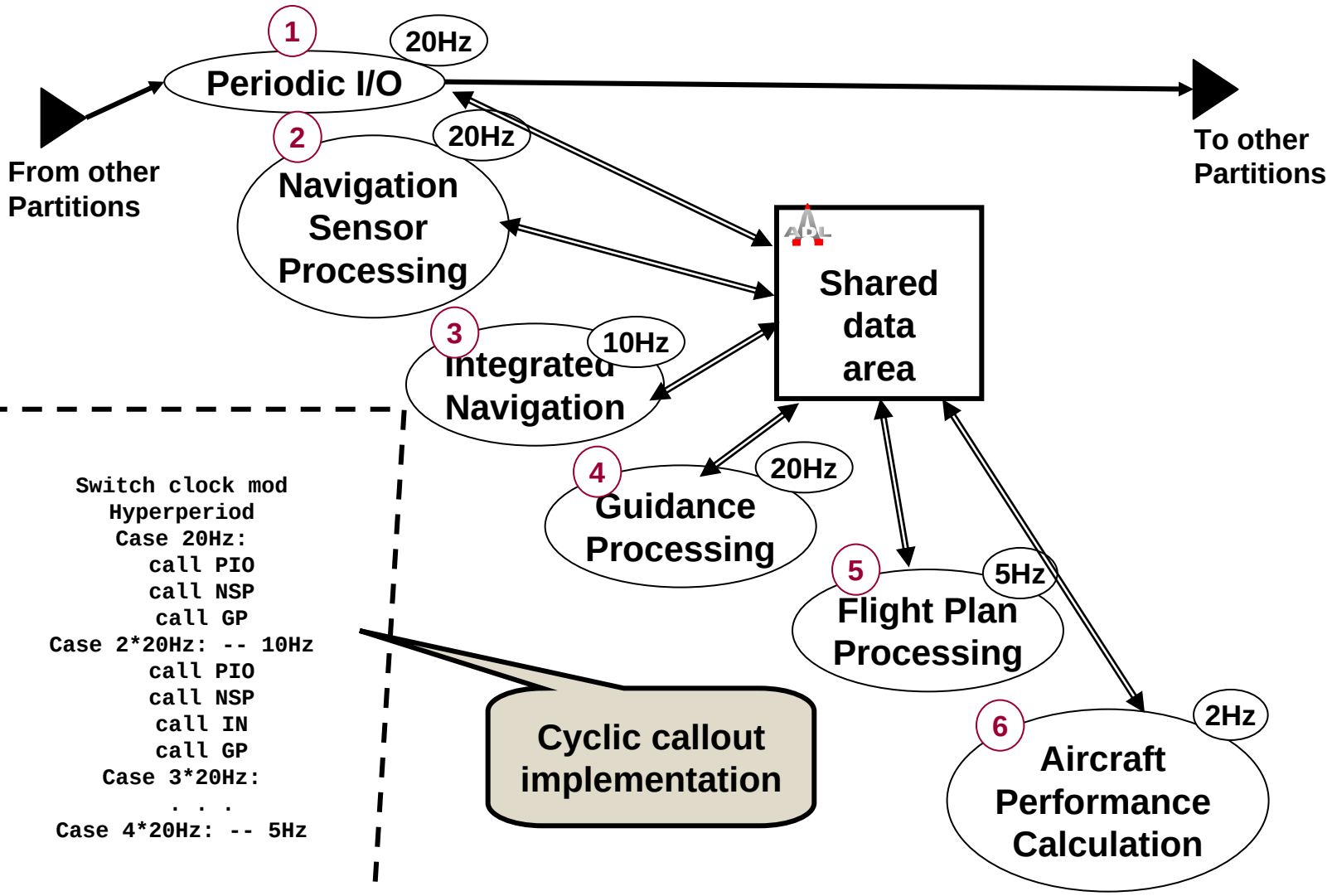
Flight Manager Context Diagram



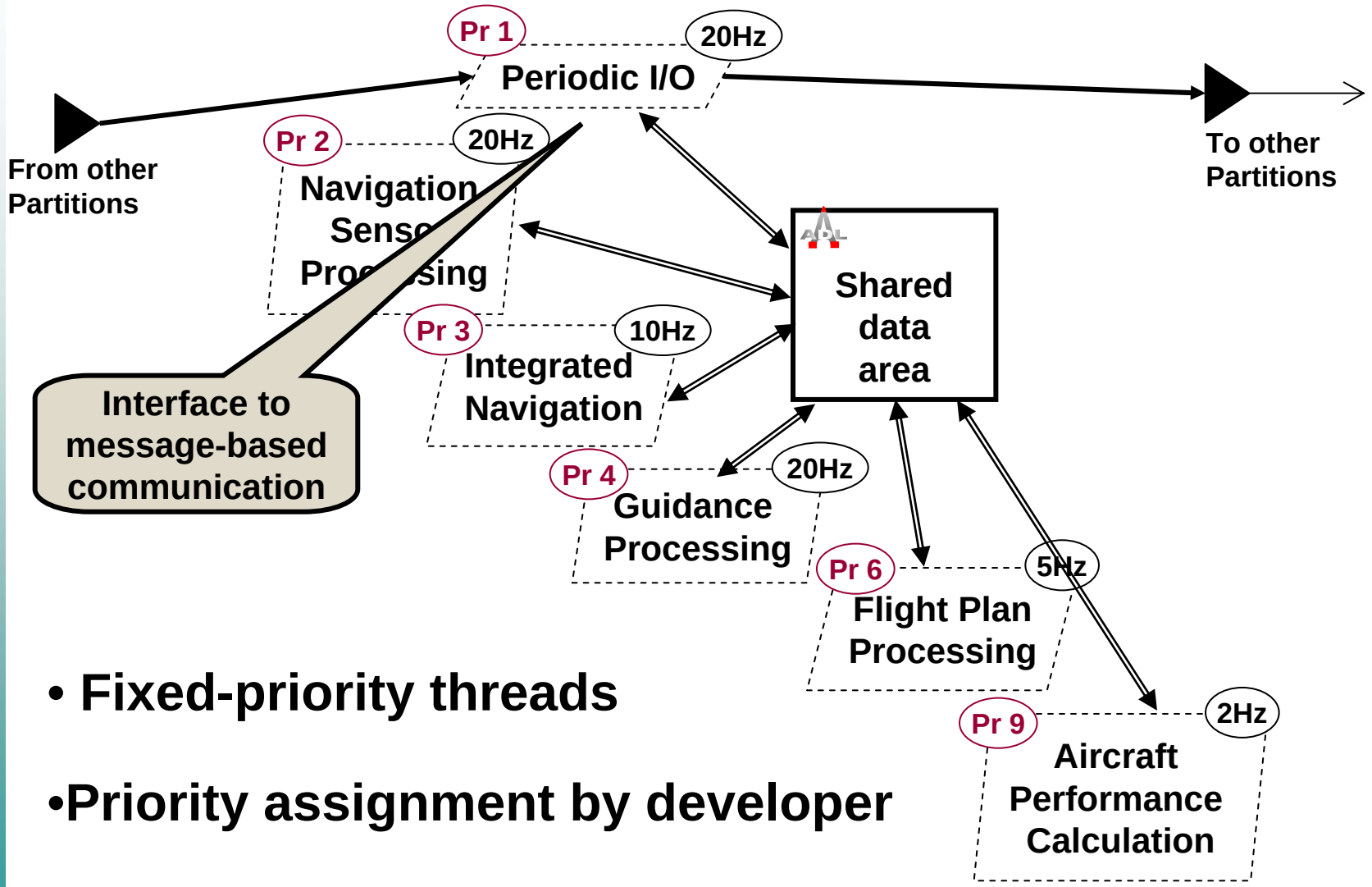
Outline

- Introduction to SAE AADL Standard
- The case study
- Towards preemptive scheduling
- Partition scheduling
- End-to-end flows
- System redundancy

A Cyclic Executive Implementation



A Naïve Thread-based Design

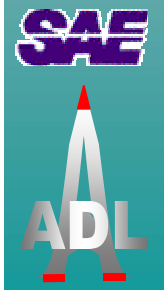


- Fixed-priority threads
- Priority assignment by developer

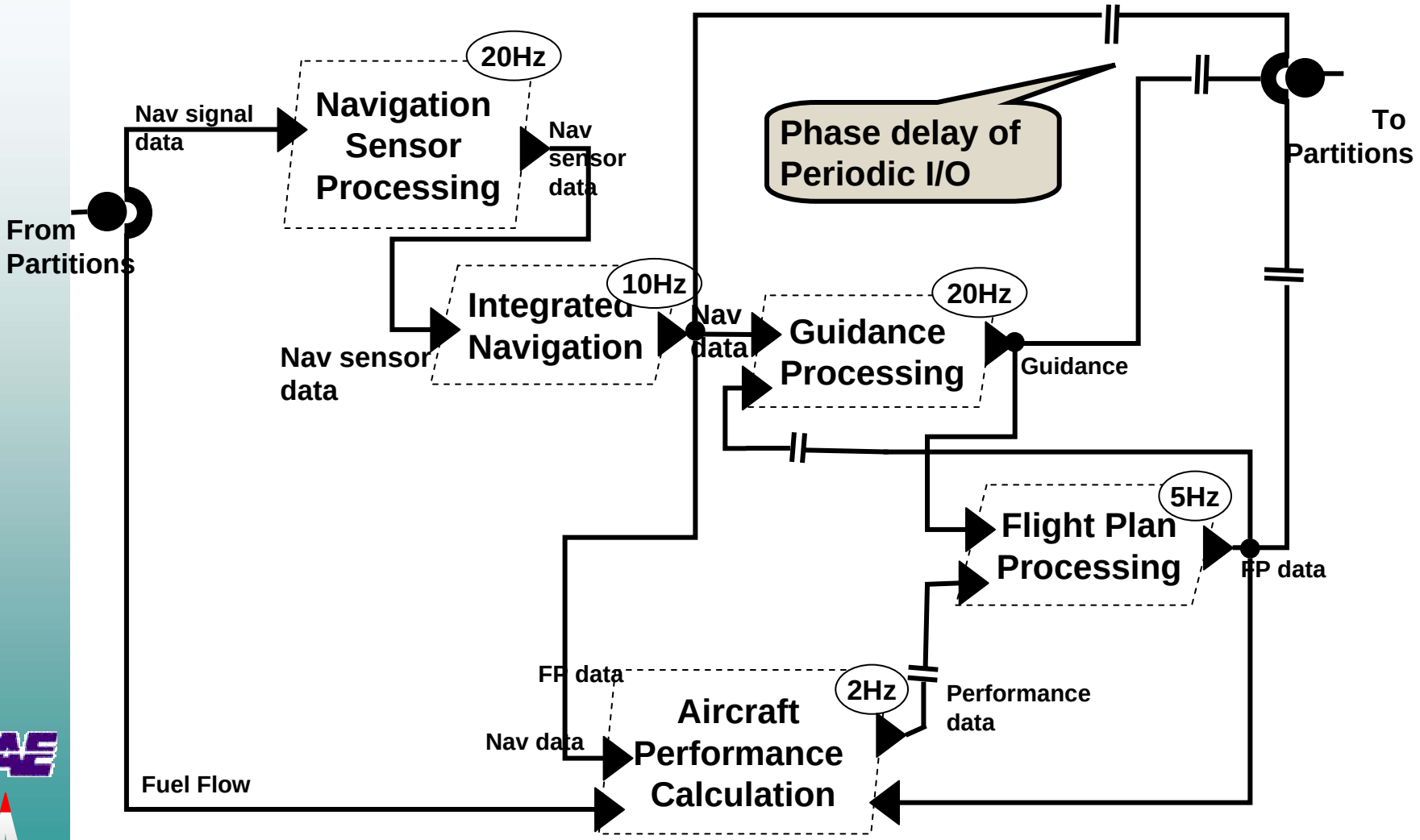
The Software Engineering Institute

Design Decisions Taken

- Shared variable communication within partition
  - Achieve efficient resource utilization
  - Accommodate legacy code
- Preemptive fixed-priority thread scheduling
  - Used Schedulability analysis (RMA) to confirm schedulability
  - Benefit of more flexible system and efficient resource usage
- Priority assignment for precedence ordering to achieve desired flow
  - Needed because of shared data communication
  - Results in potential priority inversion and non-deterministic communication



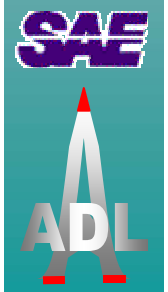
Flight Manager in AADL



The Software Engineering Institute

SAE AADL & Control Processing

- Supports mid-frame communication & single sample delay
- Shows application rates & desired phase delay explicitly
- Focus on what communication is desired, not how it is implemented
- Assures deterministic communication when desired
- Does not prescribe scheduling protocol
- Supports schedulability analysis
- Opens dialogue between control engineers and software system engineers regarding performance tradeoffs

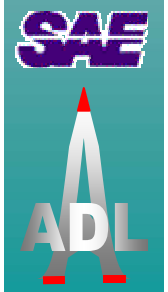


Outline

- Introduction to SAE AADL Standard
- The case study
- Towards preemptive scheduling
- Partition scheduling
- End-to-end flows
- System redundancy

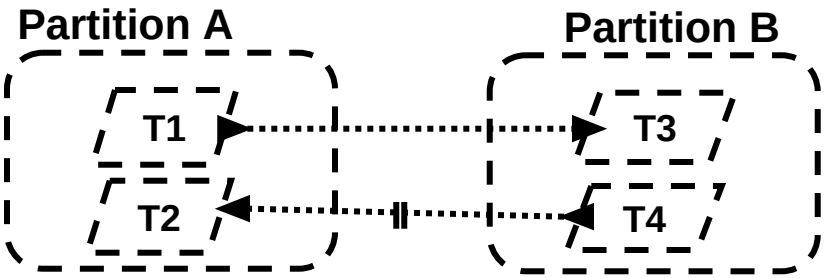
The Partition Concept

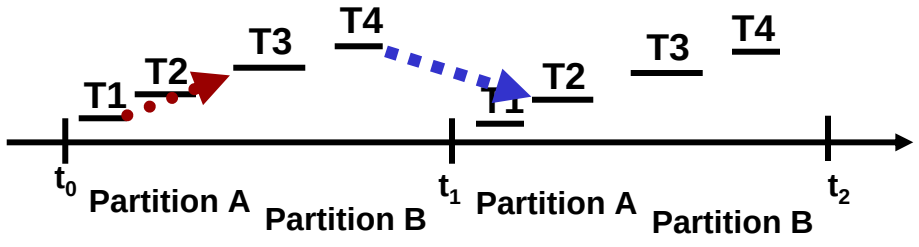
- Found in ARINC 653
- Runtime protected address space
- A virtual processor scheduled on a static timeline
- Contained threads (ARINC processes) are scheduled within the bounds of a scheduled partition
- Different partitions can use different thread scheduling protocols
- Communication of queued and unqueued data
- Inter vs. intra partition communication



Partition Order Side Effects

Partition communication via send/receive





The Software Engineering Institute

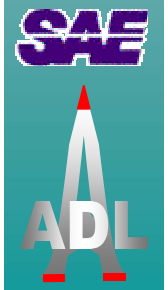
Partitioned System Design in AADL

- Partition as a core AADL extension
- Focus on partition order isolation
  - Delayed connections insensitive to partition order
  - Delayed connections insensitive to partition concurrency
  - Delayed connections contribute to latency
- Focus on latency
  - Immediate connections reduce latency
  - Immediate connections constrain partition order
  - Immediate connection cycles
    - Direct cycle: P_A.T1 -> P_B.T2 -> P_A.T3
    - Pair-wise cyclic: P_A.T1 -> P_B.T2 & P_B.T4 -> P_A.T3

Detectable by analysis

- Focus on flexibility
  - Acceptable variation in phase delay

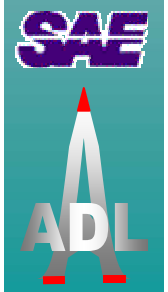
Document as property



The Software Engineering Institute

Outline

- Introduction to SAE AADL Standard
- The case study
- Towards preemptive scheduling
- Partition scheduling
- End-to-end flows
- System redundancy

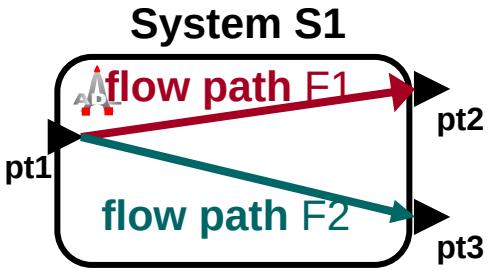


Connection Patterns

- Connection sequences
  - Pipeline, flow
- Connection tree
  - Branching flow
  - Different endpoint latencies
- Directed acyclic graph (DAG)
  - Flow with merge points
  - Phase delay difference of branches at merge point
  - Effects of phase delay oscillation in non-deterministic case
- Cyclic connections
  - Feedback control, action/observation
  - Phase delay breaks cycle

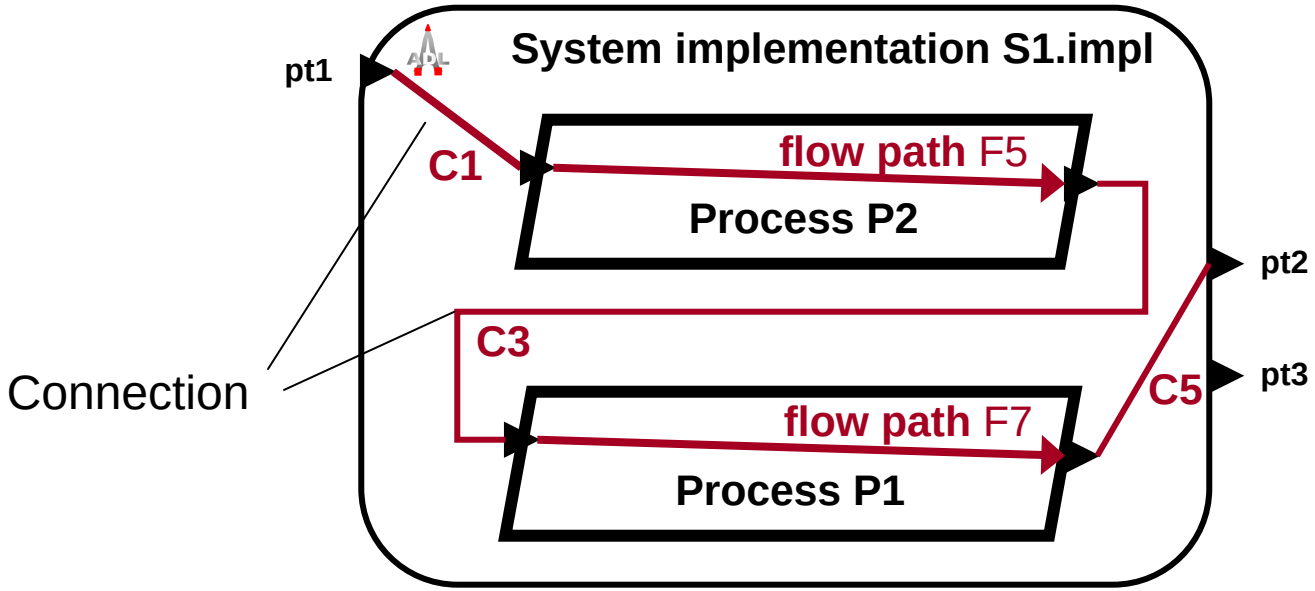
Analyzable in AADL

Flow Specification in AADL



Flow Specification

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



Flow Implementation

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

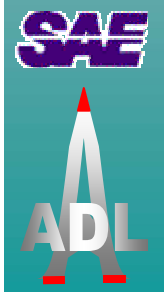
The Software Engineering Institute

Data Stream Latency Analysis

- Flow specifications in AADL
  - Properties on flows: expected & actual end-to-end latency
  - Properties on ports: expected incoming & end latency
- End-to-end latency contributors
  - Delayed connections result in sampling latency
  - Immediate periodic & aperiodic sequences result in cumulative execution time latency
- Phase delay shift & oscillation
  - Noticeable at flow merge points
  - Variation interpreted as noisy signal to controller

Potential hazard

Latency calculation & jitter accumulation



The Software Engineering Institute

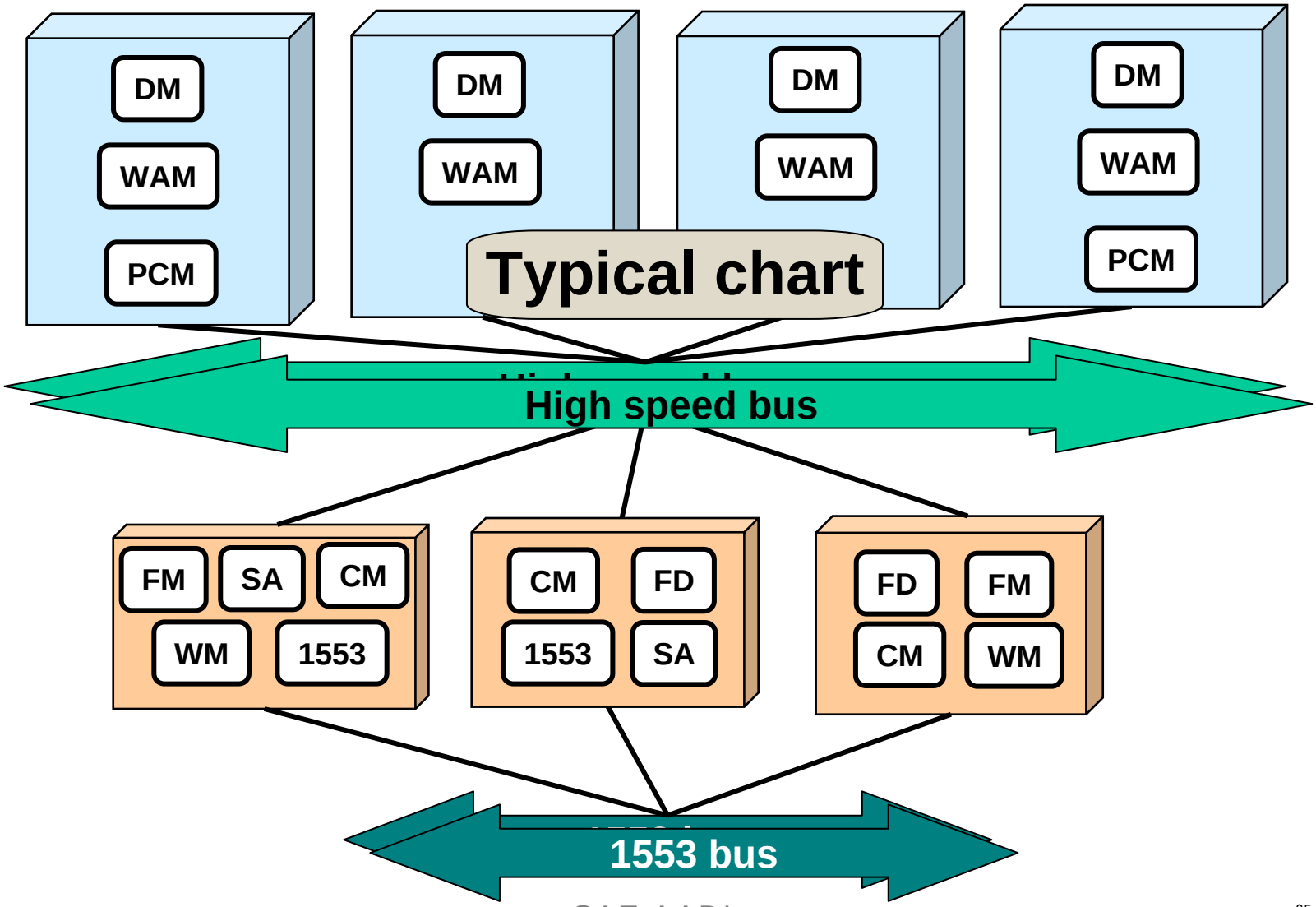
Other Flow Characteristics

- Miss rate of data stream
  - Accommodates incomplete sensor readings
  - Allows for controlled deadline misses
- State vs. state delta communication
  - Data reduction technique
  - Implies requirement for guaranteed delivery
- Data accuracy
  - Reading accuracy
  - Computational error accumulation
- Message acknowledgment semantics
  - In terms of flow steps

Outline

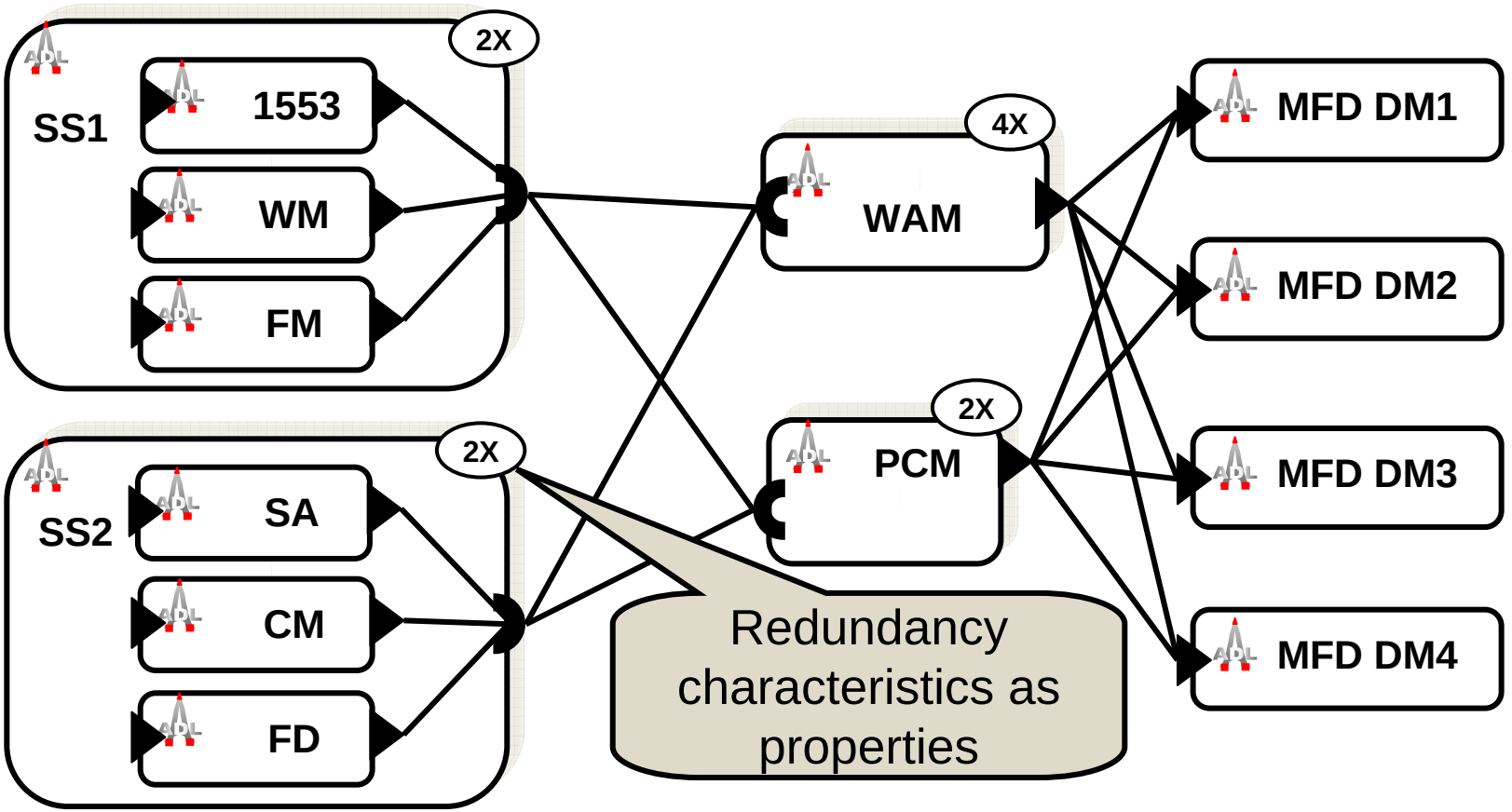
- Introduction to SAE AADL Standard
- The case study
- Towards preemptive scheduling
- Partition scheduling
- End-to-end flows
- System redundancy

System Redundancy

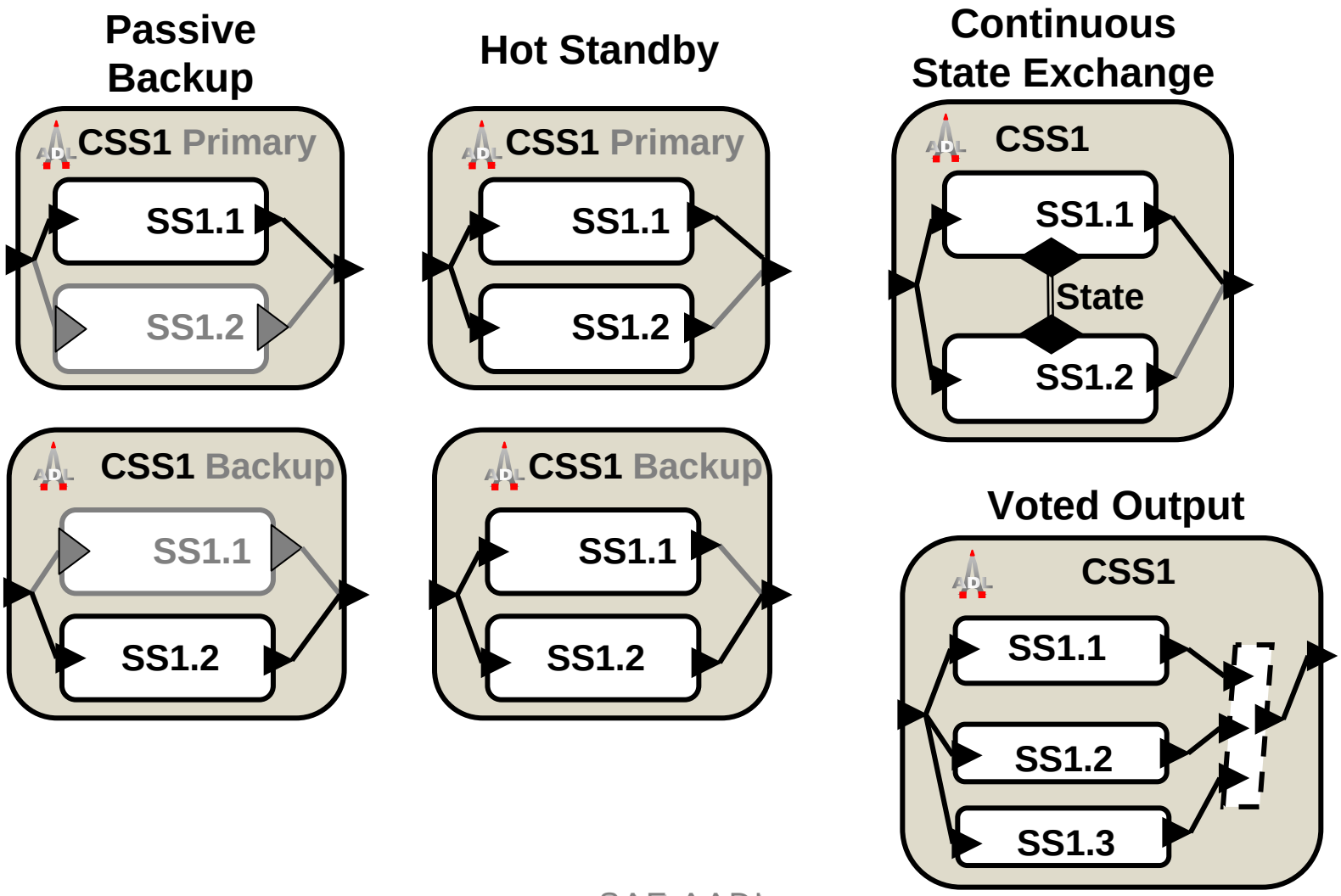


Redundancy Specification

- Redundancy abstraction
- Co-location constraints on execution platform binding

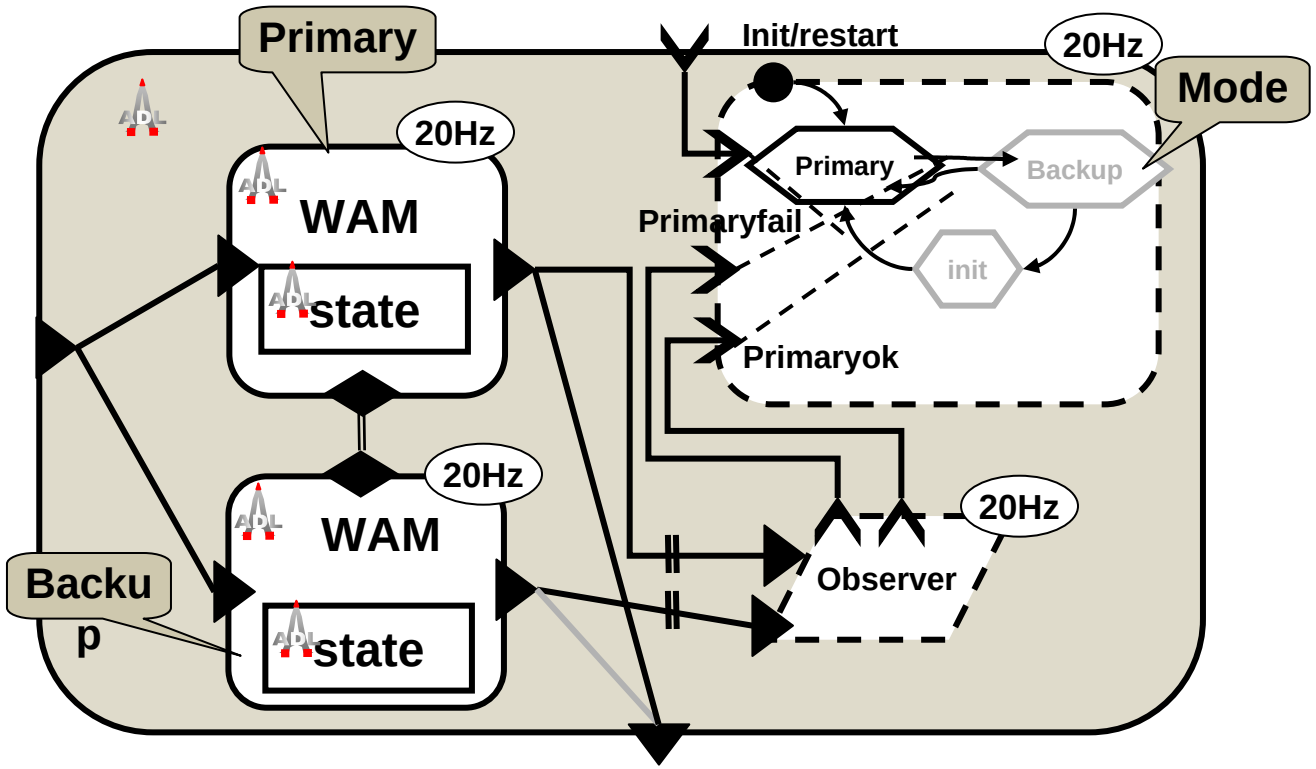


Primary/Backup Patterns



Primary Backup Synchronization

- External and internal mode control
- Errors reported as events
- Supports reasoning about Primary/Backup logic

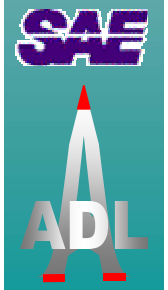


The Software Engineering Institute

Observations On System Redundancy

- Redundancy as an abstraction
  - Multiple redundant instances
  - Grouping of redundant instances
  - Redundancy protocol selection
  - Deployment constraints
- Redundancy mechanism as pattern
  - An orthogonal architecture view
  - Nominal & anomalous behavior
  - Modeling of redundancy logic

Understandable and analyzable



The Software Engineering Institute

Final Observations

We demonstrated a pattern-based analysis approach

- Use of SAE AADL as notation for capturing architecture patterns in actual systems
- Early identification of systemic issues thanks to precise execution semantics of SAE AADL

Full scale architecture modeling and analysis provides prediction and validation of non-functional properties