Investigation of System Timing Concerns in Embedded Systems: Tool-based Analysis of AADL Models

Peter Feiler  
Software Engineering Institute  
phf@sei.cmu.edu  
412-268-7790
Outline

- Case study background
- Preemptive scheduling & communication timing
- Low fidelity flow latency analysis
- Cross-partition communication timing
- Network load modeling & analysis
- Modal system analysis
- AADL-based engineering environments
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
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
Outline

• Case study background
• Preemptive scheduling & communication timing
• Low fidelity flow latency analysis
• Cross-partition communication timing
• Network load modeling & analysis
• Modal system analysis
• AADL-based engineering environments
Flight Manager: Principal Functionality

- Navigation Sensor Processing
- Integrated Navigation
- Guidance Processing
- Flight Plan Processing
- Aircraft Performance Calculation
- Flight Manager State
- Periodic I/O
- Nav Radio
- Auxiliary service

Processing functions

Subsystem \( \equiv \) CSCI \( \equiv \) Partition

From other Partitions

20Hz

To other Partitions

20Hz

10Hz

5Hz

2Hz
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

Simple mapping to a cyclic callout implementation

From other Partitions
Periodic I/O
20Hz
Navigation Sensor Processing
20Hz
Integrated Navigation
10Hz
Shared data area
20Hz
Guidance Processing
5Hz
Flight Plan Processing
2Hz
Aircraft Performance Calculation

To other Partitions

1 2 3 4 5 6
A Naïve Thread-based Design

From other Partitions

Pr 1
Periodic I/O

20Hz

Pr 2
Navigation Sensor Processing

20Hz

Pr 3
Integrated Navigation

10Hz

Pr 4
Guidance Processing

20Hz

Pr 6
Flight Plan Processing

5Hz

Pr 9
Aircraft Performance Calculation

2Hz

To other Partitions

Shared data area

Potential priority inversion due to priority assignment

Tasks must complete within frame => cyclic executive behavior

Potential non-deterministic communication due to preemption

From other Partitions

 Decreasing Priority

© 2005 by Carnegie Mellon University

www.aadl.info
Priority Inversion Checker

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

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

Potential priority inversion identifiable by analysis tool

Two pages of Java code
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

From other Partitions

Periodic I/O

Navigation Sensor Processing

Integrated Navigation

Shared data area

Guidance Processing

Flight Plan Processing

Aircraft Performance Calculation

Implemented via shared data
Achieved via precedence ordering
Flight Manager in AADL

AADL connections have precise timing semantics

From Partitions

Navigation Sensor Processing

20Hz

Integrated Navigation

10Hz

Guidance Processing

20Hz

Flight Plan Processing

5Hz

Aircraft Performance Calculation

2Hz

Fuel Flow

Nav signal data

Nav data

Nav sensor data

Nav sensor data

FP data

Performance data

5Hz
Data Stream Characteristics

- **Expected stream characteristics**
  - Property values of incoming ports
  - Stream element frame delay
    - Exact or range
  - Non-timing characteristics
    - Stream element miss rate
    - Value & value delta constraints

- **Provided stream characteristics**
  - Property values of outgoing ports
  - Communication protocol properties
  - Execution platform (bus) properties

Utilize property set extension mechanism of AADL to document assumptions.

Analysis plug-ins to identify communication non-determinism & to check data stream consistency.
Outline

• Case study background
• Preemptive scheduling & communication timing
• Low fidelity flow latency analysis
• Cross-partition communication timing
• Network load modeling & analysis
• Modal system analysis
• AADL-based engineering environments
Flow Analysis of Polled Event Streams

- Event stream examples
  - Multi-Function Display interaction
  - Physical dial & switch processing
- Periodic polling at 20 Hz
  - Maintain predictability of periodic processing model
  - Reduce sampling latency
- Issue
  - Realistic scheduling analysis for 100’s of polling tasks
- Towards event-based system model
  - Realistic bounded arrival rate
  - Leverage system modality
  - Analytic models for stochastic processing
Flight Director Command Flow

- Cockpit Display
- Display Manager
- Page Content Manager
- Flight Manager
- Flight Director

Request for new page
New page content
Subsystem assumed to be a partition
Command Flow Timing

Sampling of input

Display Manager

Page Content Manager

Flight Manager

Flight Director

Cross-partition communication assumed to be a frame boundary
Response Time Analysis

• Worst-case scenario
  – Period delay per partition communication
    • DM sampling latency (max. = partition period)
    • Six periods of cross-partition communication latency
    • DM execution latency (max. = partition period)
  – Display hardware latency
A Flow Latency Analysis Plug-in

- Leverages timing semantics of partition concept
- Utilizes introduction of new property
- Written as OSATE analysis plug-in
- One page of Java code
- <demo>

Adapt plug-in to assess impact of partitioning due to safety criticality levels on critical flows
Outline

- Case study background
- Preemptive scheduling & communication timing
- Low fidelity flow latency analysis
- Cross-partition communication timing
- Network load modeling & analysis
- Modal system analysis
- AADL-based engineering environments
ARINC 653 Partitions & AADL

- ARINC Partition
- Intra-partition communication
  - Buffer
  - Blackboard
  - Semaphore
  - Event
- Inter-partition communication
  - Sampling port
  - Queuing port
- AADL process & processor abstraction
- AADL thread interaction
  - Event data port connection
  - Shared data component
  - Concurrency protocol property
  - Event port connection
- AADL process interaction
  - Data port connection
  - Event data port connection

Predeclared properties for queue processing characteristics

Project-specific properties can be added
Cross-Partition Communication Timing

- ARINC 653 cross-partition communication specifies frame-delayed communication
- Timing semantics of implementations
  - Application level send/receive calls
  - Periodic I/O to handle cross-partition communication
  - Preemptive I/O communication partition
    - Device I/O with minimized latency
    - Cross-partition communication

Modeling with immediate and delayed connection timing semantics
Partition Order & Timing Semantics

Partition communication via send/receive
Transmission initiated by send call

Partition A before Partition B

Timing semantics are sensitive to partition order
Cross-Partition Communication With Periodic I/O

Periodic I/O adds phase delay
Periodic I/O is sensitive to partition order

Timing semantics are STILL sensitive to partition order
Response Time Analysis Revisited

- **Period delay per partition communication**
  - DM sampling latency (max. = partition period)
  - Six periods of cross-partition communication latency
  - DM execution latency (max. = partition period)

- **Single processor static timeline**
  - Account for mid-frame cross-partition communication
  - Reduces cross-partition communication latency to three periods

- **Multiple processor synchronous system**
  - Account for mid-frame cross-processor communication
  - Take into consideration bus/network latency

- **Multiple processor asynchronous system**
  - Asynchronous sampling with max. sampling latency = period
Partition Communication in AADL

• AADL runtime system manages
  – Partition execution
  – Frame-delayed inter-partition communication
  – Assures delayed communication for synchronous systems

Periodic system bus as in Time-Triggered Architecture (TTA) provides a common reference time

• Communication via ports and connections
• Delayed connections insulate from partition order
• Latency variation as flow property
Communication in Asynchronous Systems

• Sampling of data stream
  – Task scheduling based on independent clocks
  – Steve’s aging formula provides upper bound on latency
  – Worst-case latency variability: sum of partition rates

• Flow-driven processing
  – Task dispatch on arrival of data (independent of local clock)
  – No sampling latency
  – Task deadlines provide upper bound on latency
  – Worst-case latency variability: sum of deadline-BCET

Globally time-stamped data allows for latency variability control
Outline

• Case study background
• Preemptive scheduling & communication timing
• Low fidelity flow latency analysis
• Cross-partition communication timing
• Network load modeling & analysis
• Modal system analysis
• AADL-based engineering environments
Analyzeable and Reconfigurable AADL Specifications for IMA System Integration

David Statezni
Advanced Technology Center
Rockwell Collins, Inc.
Proof of Concept Example

Generic Display System with Rockwell Collin’s Switched Ethernet LAN

- Only LAN-related entities modeled
- Model generated from Input/Output & Thread data stored in Database

Model Size

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

21000 lines of textual AADL
CDU Subsystem Architecture

- Not modeled for this AADL example
Graphical Software (Logical) View
Workload-based Performance Analysis

- Workload determined by
  - Connectivity in application architecture
  - Binding to particular execution platform configuration

- Modeling options
  - Explicit modeling of bound configuration as devices
  - Utilization of bus abstraction & binding properties

- Input to In-house network performance analysis tool
  - Via AADL XML representation
  - Device-based model
    - Workload data explicitly recorded
  - Bus-based model
    - Workload computed from bindings

Flexibility for what-if analysis

Models of different fidelity
Multiple Analysis Concerns

• Architecture annotated with AADL properties
  – Predeclared properties
  – Project-specific properties

• Demonstration examples
  – Processor binding & scheduling analysis
    • Based on periods, deadlines
  – Flow latency analysis
    • Based on task characteristics

Reliability & Fault tree analysis through error model annotations
Outline

• Case study background
• Preemptive scheduling & communication timing
• Low fidelity flow latency analysis
• Cross-partition communication timing
• Network load modeling & analysis
• Modal system analysis
• AADL-based engineering environments
Modal System Analysis

Operational modes of systems
Hybrid control systems

• Modeling of system modes
  – Different task & interaction configurations
  – Different property values
  – Less conservative analysis results
  – Mode-specific analysis results as modal property values

• Tool-based analysis of modal AADL models
  – Modal system instance models vs. model per mode
  – Transparent modal model traversal vs. modal analysis tools

AADL meta model defines modal instance models

© 2005 by Carnegie Mellon University
Outline

• Case study background
• Preemptive scheduling & communication timing
• Low fidelity flow latency analysis
• Cross-partition communication timing
• Network load modeling & analysis
• Modal system analysis
• AADL-based engineering environments
Two-Tier Tool Strategy

• Open Source AADL Tool Environment (OSATE)
  – Developed by SEI
  – Low entry cost solution (no cost CPL)
  – Multi-platform support based on Eclipse
  – Vehicle for in-house prototyping of project specific architecture analysis
  – Vehicle for architecture research with access to industrial models & industry exposure to research results

• Commercial Tool Support
  – UML tool environment extension based on UML profile
  – Extension to existing modeling environment with AADL export/import
  – Analysis tools interfacing via XML or XML to native filter
  – Runtime system generation tools

Artisan, Rational, …
TNI Stood
Tool Interoperability

AADL Front-end

- Textual AADL
- Graphical AADL
- Graphical View
- Parser

Co-resident tool

Declarative AADL Model

- Name Resolution
- Semantic Checking

AADL Instance Model

Persistent XML document

Tool-specific XML Representation

Convert

Tool-specific representation
OSATE Capabilities

- OSATE Release 0.4.0 based on Eclipse Release 3
- Online AADL help
- Text to XML & XML to text
- Syntax-sensitive text editor
- Parsing & semantic checking of full AADL
- AADL property viewer
- Syntax-Sensitive AADL Object Editor
- Model versioning & team support
- Model instantiation
- Model consistency checking
- AADL to MetaH translator
- Plug-in development

- Next release Feb 2005
  Graphical editor
  Multi-file support

- Over 250 downloads internationally
- Processed 21000 line AADL model in 6 sec
An Extensible Engineering Environment

- AADL Extensions
  - Error model
  - Concurrency behavior

- Embry-Riddle Reliability Analysis

- Architecture Import & Extraction

- Architecture Export
  - MetaH, TTA

- System Verification Manager (CMU)
  - Simulink/Matlab, Dymola models

- Architecture Consistency
  - Plug-ins

- OMNET++
  - Network simulation

- Object Model Interface
  - Network model

- Model Export
  - Filters
  - Timing model

- MetaH Toolset (Honeywell)
  - Scheduling analysis
  - Reliability analysis
  - Isolation analysis
  - Runtime system generation

- TimeWeaver (CMU)
  - Distributed resource allocation
  - Multi-platform runtime system generation

- TimeWiz Commercial Tool
  - Scheduling analysis
  - Execution trace analysis

© 2005 by Carnegie Mellon University

www.aadl.info
Architecture Consistency Examples

Systemic issues discovered in architecture analysis
Detection codified as analysis plug-ins

- Expected component connectivity
- Data stream characteristics
  - Miss rate of data stream
  - State vs. state delta communication
  - Data value constraints
- Safety criticality
- Security levels & information flow
- Impact dependency
- Spec-based flow analysis

Use in models of different fidelity
Benefits

• Model-based system engineering benefits
  – Prediction early and throughout lifecycle
  – Model evolution & tool-based processing
  – Multiple analyses based on same architecture model
  – Predictable runtime characteristics at different modeling fidelity
  – Reduced integration & maintenance effort

• Benefits of AADL as SAE standard
  – Common modeling notation across organizations
  – Interchange & integration of architecture models
  – Single architecture model augmented with analysis properties
  – Extensibility accommodates special analysis needs
  – Tool interoperability & extensible engineering environments
  – Aligned with UML-based engineering practices