Model-Based Embedded System Engineering & Analysis of Performance-Critical Systems

Peter H. Feiler

Jan 2007

Open Group RT Forum
Outline

Model-based Embedded System Engineering
• Resource Consumption: Resource Budgeting
• Real-time Performance: Concurrency & Timing
• Real-time Performance: End-to-end Latency
• Security: Confidentiality Analysis
• Data Quality: Temporal Data Consistency
• Availability & Reliability: Fault Tree Analysis
• Conclusions
Software & System Engineering

System Engineering

- Operational System
- Physical System Model
- Physical Component
- Computing Platform
- Physical characteristics relevant to embedded application as properties in AADL model

Embedded Software System Engineering

- SAE AADL
- Embedded Software System
- Embedded Software

Application Domain

© 2006 by Carnegie Mellon University
A Control Engineer Perspective

Matlab

Application Code

Simulink

Component Analysis

Validate expected control behavior

Validate simulation

Tune parameters

with Text_IO; package Main is
begin
  type real is digits 14;
  type flag is boolean;
  x : real := 0.0;
  ready : flag := TRUE;
end;
with Text_IO;
package Main is
begin
  type real is digits 14;
  type flag is boolean;
  x : real := 0.0;
  ready : flag := TRUE;
Annotated Architecture Model

- Schedulability analysis
- Latency analysis
- Safety analysis
- Reliability analysis
- Fault annotations
- Timing annotations
- Alternative Hardware Bindings

Application

Platform

Examples of analyses from same model

Low incremental cost for additional analyses & simulations!!!
Impact Analysis from Models

- Security
  - Intrusion
  - Integrity
  - Confidentiality

- Real-time Performance
  - Bandwidth
  - CPU time
  - Power consumption

- Availability & Reliability
  - MTBF
  - FMEA
  - Hazard analysis

- Resource Consumption
  - Data quality
  - Data precision/accuracy
  - Temporal correctness
  - Confidence

- Data Quality
  - Execution time/Deadline
  - Deadlock/starvation
  - Latency
Outline

• Model-based Embedded System Engineering
• Resource Consumption: Resource Budgeting
• Real-time Performance: Concurrency & Timing
• Real-time Performance: End-to-end Latency
• Security: Confidentiality Analysis
• Data Quality: Temporal Data Consistency
• Availability & Reliability: Fault Tree Analysis
• Conclusions
Resource Budgeting

• Resource management throughout life cycle
• Resource budgets for processors, memory, bus/networks
  – Compute resources: MIPS, MB, bandwidth
  – Physical resources: power consumption
• Budgets for major subsystems
  – System wide & resource specific budget totals
• System decomposition & budget refinement
• Task & communication model
  – Budgets against execution & communication rates
  – Scheduling analysis
Resource Budget Properties

- Property definition: Example CPU resource

\[
\text{MIPSCapacity:aadlreal units SEI::Processor\_Speed\_Units applies to (processor, system);} \\
\text{Processor\_Speed\_Units : type units} \\
\text{(KIPS, MIPS => KIPS * 1000, GIPS => MIPS * 1000);} 
\]

- Property use

\[
\text{processor missionProcessor properties} \\
\text{SEI::MIPSCapacity => 1500 MIPS;} \\
\text{end missionProcessor;} \\
\text{System AvionicsSystem properties} \\
\text{SEI::MIPSBudget => 300 MIPS;}
\]
Basic System Architecture

- High speed bus
- 1553 bus
- Application component bound to processor
- DM
- WAM
- PCM
- FM
- SA
- CM
- WM
- CM
- FD
- 1553
- SA
- CM
- WM
- 1553
Resource Budget Analysis Demo

- Parts model of major subsystems
- Does the total system budget exceed the total capacity?

- Initial subsystem/partition assignment
- Does the budget total of assigned subsystems exceed processor & memory capacity?

- Subsystems interactions
- Does the bandwidth budget total exceed network backbone capacity?
- What are network/bus-specific workloads?
Outline

• Model-based Embedded System Engineering
• Resource Consumption: Resource Budgeting
  Real-time Performance: Concurrency & Timing
  • Real-time Performance: End-to-end Latency
• Security: Confidentiality Analysis
• Data Quality: Temporal Data Consistency
• Availability & Reliability: Fault Tree Analysis
• Conclusions
Migration of Legacy Implementation

Flight Manager
- Pr 1: 20Hz
- Pr 2: 20Hz
- Pr 3: 10Hz
- Pr 4: 20Hz
- Pr 5: 5Hz
- Pr 6: 3Hz
- Pr 9: 2Hz

Periodic I/O
- From other Partitions
- To other Partitions

Latency variation

Potential Priority Inversion

Integrated Navigation
- Pr 3: 10Hz

Navigation Sensor Processing

Guidance Processing
- Pr 4

Flight Plan Processing
- Pr 5

Shared data area

Legacy code uses shared data area
Efficient thread communication

From other Partitions

© 2006 by Carnegie Mellon University
Intended Data Flow

From other Partitions

Navigation Sensor Processing
- Pr 1: Periodic I/O 20Hz

Integrated Navigation
- Pr 2: 20Hz

Guidance Processing
- Pr 3: 10Hz

Flight Plan Processing
- Pr 4: 20Hz

Aircraft Performance Calculation
- Pr 6: 5Hz
- Pr 9: 2Hz

Shared data area

Priority assignment achieves desired data flow

Intended flow documented in design document table

Decreasing Priority

To other Partitions

© 2006 by Carnegie Mellon University
Flow-based Flight Manager Model

- Navigation Sensor Processing
  - Nav signal data
  - Nav sensor data
  - 20Hz

- Integrated Navigation
  - Nav data
  - Nav sensor data
  - 10Hz

- Guidance Processing
  - Guidance
  - Nav data
  - 20Hz

- Flight Plan Processing
  - FP data
  - Flight Plan
  - 5Hz

- Aircraft Performance Calculation
  - FP data
  - Nav data
  - Performance data
  - 2Hz

- Periodic I/O
  - To Partitions
  - Phase delay of Periodic I/O

From Partitions

© 2006 by Carnegie Mellon University
Scheduling Analysis

- Scheduling protocol determines analysis
  - Processor budget for static time line (cyclic executive)
  - Rate-monotonic Analysis (RMA) for preemptively scheduled fixed-priority tasks
  - 100% utilization for Earliest Deadline First (EDF)

- What if analysis of
  - Schedulability under different dispatch & scheduling schemes
  - Miss-estimated worst case execution times (WCET)

Commercial real-time system analysis tools provide such support
Scheduling Analysis Demo

• If a single processor system is not schedulable
• Explore these options using AADL and analysis tools
  – Leverage operational modes
  – Processor speed dependent execution time
  – Rebind to different execution platform
  – Reduce worst-case execution time
  – Identify schedulable rate from sensitivity analysis results

• Might consider
  – Repartition system
  – Use faster processor
  – Add second processor
  – Rewrite code to make it faster
  – Consider lower signal processing rate for controller
Outline

• Model-based Embedded System Engineering
• Resource Consumption: Resource Budgeting
• Real-time Performance: Concurrency & Timing

Real-time Performance: End-to-end Latency
• Security: Confidentiality Analysis
• Data Quality: Temporal Data Consistency
• Availability & Reliability: Fault Tree Analysis
• Conclusions
device brake_pedal
features
  brake_status: out data port bool_type;
flows
  Flow1: flow source brake_status;
end brake_pedal;

system cruise_control
features
  brake_status: in data port;
  throttle_setting: out data port;
flows
  brake_flow_1: flow path brake_status \rightarrow throttle_setting;
end cruise_control;

device throttle_actuator
Features
  throttle_setting: in data port float_type;
flows
  Flow1: flow sink throttle_setting;
end throttle_actuator;
High-level Flow Analysis

- Determine minimum response time, maximum command rate
- Display Manager
- Cockpit Display
- Warning Annunciation Manager
- Page Content Manager
- Flight Manager
- Flight Director
- Situation Awareness
- Weapons Manager
- Comm. Manager
- Indirect connectivity due to late addition
- 1553 Access
- Subsystems mapped to partitions
Response Time Analysis Demo

• High-level end-to-end analysis
  – Account for latency impact of partition hops
  – Account for device latency
  – Account for subsystem processing

• Detailed end-to-end latency analysis
  – AADL model from design data base
  – Task model with 165 end-to-end flows
Outline

- Model-based Embedded System Engineering
- Resource Consumption: Resource Budgeting
- Real-time Performance: Concurrency & Timing
- Real-time Performance: End-to-end Latency
- Security: Confidentiality Analysis
- Data Quality: Temporal Data Consistency
- Availability & Reliability: Fault Tree Analysis
- Conclusions
Distributed Data Processing

Freshness? Confidence? Security?

Freshness: 10s
Confidence: 70%
Security: secret
Security: Objective

- **Confidentiality** concerns that sensitive data should only be disclosed to or accessed/modified by authorized users, i.e., enforcing prevention of unauthorized disclosure of information.

- **Objective**: Model security attributes for an architecture to verify that data is properly accessed and handled by users and applications.

- **Means** to achieve confidentiality include enforcing access control, perform encryption, partitioning of system

- **Confidentiality frameworks**
  - Bell-LaPadula framework: military applications
  - Chinese wall framework: commercial applications
  - Access role/role-based access framework
Bell-LaPadula: Subjects and Objects

• In Bell–La Padula, subjects operate on objects.
  – Subjects need permission, expressed as security level, to use objects.

• The security level of a subject or object is a pair:
  – Classification
    • Drawn from a partial order of classifications (e.g., unclassified < confidential < secret < top secret).
  – Set of categories
    • Drawn from a set of labels (e.g., NATO, Nuclear, Crypto).

• \((\text{Class1, Set1})\) dominates \((\text{Class2, Set2})\) if and only if
  – \(\text{Class1} >= \text{Class2}\)
  – \(\text{Set1} \supseteq \text{Set2}\)
Example

Minimum security level required is (secret, {a,b})

Uncontrolled sanitization as (conf, {a}) is dominated by max(class_i, cat_i), i=1..3

Error: O_4 is read-only
Warning: O_1 are O_2 shared more freely than necessary.
Analyzed System: Errors

CompleteSystem_Impl

Src1: Producer1
(unclassified, {A})
output

Src2: Producer2
(unclassified, {B})
output

Comp: Computer
in1
(unclassified, {A})
in2
(Secret, {A, B})
result
c1
c2

Dest: Consumer
input
(unclassified, {B, C})
result
c3

output
(output, {B})
(output, {A})
(output, {A, B})
(output, {B, C})

Problems
Properties | AADL Property Values | Error Log | Progress | Search
---|---|---|---|---
3 errors, 0 warnings, 0 infos

- Connection c3's source security level (confidential, {B, A}) does not equal the level of its destination (confidential, {C, B})
- The security level of out port result, (confidential, {B, A}), does not equal the level, (secret, {B, A}), of its containing component
- The security level of subcomponent dest, (confidential, {C, B}), is not dominated by the security level, (secret, {B, A}), of its containing component
Outline

• Model-based Embedded System Engineering
• Resource Consumption: Resource Budgeting
• Real-time Performance: Concurrency & Timing
• Real-time Performance: End-to-end Latency
• Security: Confidentiality Analysis

Data Quality: Temporal Data Consistency
• Availability & Reliability: Next presentation
• Conclusions
Performance Improvement Gone Bad

A real customer experience

- Ground station to accommodate sensor load growth
  - Reduce load in network
  - Two subsystems communicate state change instead of state

- The impact
  - Other subsystems increase network load sporadically
  - Receiving subsystem goes down

- The cause
  - Transmission protocol without guaranteed delivery
  - Overload result in dropping of transmitted state deltas
  - Missing deltas result in inconsistent receiver state
Avoiding Future Mistakes

• Relevant characteristics as properties
  – State vs. state-change communication through ports
  – Bus protocols with or without guaranteed delivery

• Annotating the model
  – Application engineer characterizes data stream
  – Embedded system engineer characterizes hardware & protocols

• The analysis tool
  – Check that connections carrying state changes are bound to buses with guaranteed delivery
Capturing Domain Characteristics

• Safe upgrading of controllers
  – Data range limits and units of measurement for input & output
  – Documenting setpoint constraints as bounded value deltas
  – Consistency checking along connections

• Security levels & information flow
  – Components with security levels
  – Security levels & containment hierarchy
  – Security levels and connections
  – Security levels & execution platform components

• Safety criticality & control of components
  – Component have safety criticality levels
  – Impact on high criticality components
Outline

• Model-based Embedded System Engineering
• Resource Consumption: Resource Budgeting
• Real-time Performance: Concurrency & Timing
• Real-time Performance: End-to-end Latency
• Security: Confidentiality Analysis
• Data Quality: Temporal Data Consistency

Availability & Reliability: Next presentation
• Conclusions
Outline

• Model-based Embedded System Engineering
• Resource Consumption: Resource Budgeting
• Real-time Performance: Concurrency & Timing
• Real-time Performance: End-to-end Latency
• Security: Confidentiality Analysis
• Data Quality: Temporal Data Consistency
• Availability & Reliability: Fault Tree Analysis

Conclusions
Predictable Embedded System Engineering

- Tracking of requirements, planned and actual resource usage
- Analysis at multiple levels of fidelity
- Multiple analysis perspectives from annotated architecture model
- Validated timing & data stream semantics for control algorithm and software implementation
- Codified engineering rules of thumb
Model-Based Engineering Benefits

- Analyzable models drive development
  Prediction of runtime characteristics at different fidelity
  Bridge between control & software engineer
  Prediction early and throughout lifecycle
  Reduced integration & maintenance effort

- Benefits of AADL as SAE standard
  Common modeling notation across organizations
  Single architecture model augmented with properties
  Interchange & integration of architecture models
  Tool interoperability & integrated engineering environments

© 2006 by Carnegie Mellon University
SEI AADL Application & Education

• Open source AADL Tool environment (OSATE)
• Embedded Engineering With AADL Tutorial
• Public two-day course offering by SEI
  – Model-based Engineering with SAE AADL
• Pilot projects with customers
• Case studies of Model-based Engineering
• AADL User Guide & Embedded Systems Engineering Handbook
For more information:

Peter H. Feiler
Email: phf@sei.cmu.edu
Phone: 412-268-7790
SEI: www.sei.cmu.edu
AADL: www.aadl.info