Flight Systems are Cyber-Physical Systems

Dr. Christopher Landauer
Software Systems Analysis Department
The Aerospace Corporation

Computer Science Division / Software Engineering Subdivision
08 November 2012

Summary

- We show how to model flight systems (software and hardware) as cyber-physical systems, by combining appropriate modeling paradigms: discrete interactions for software and other events, continuous processes for movement and hardware, and story lines for scenarios.

chris.landauer@aero.org

Outline

- Software-Intensive Embedded Systems
- The Real World
- What is Hard
- A New(ish) Approach: Com+Ode
  - *A New Hard Mathematical Problem*
- Other Well-Known Modeling Mechanisms:
  - *AADL, DEVS, Modelica, and SysML*
- What is missing in these approaches?
- Modeling Process Lessons Learned
- Conclusions and Recommendations
- Some References

chris.landauer@aero.org

Software-Intensive Embedded Systems

- Software is well-modeled and even well-defined by formal and systematic mechanisms
  - *Event patterns*
  - *Alternatives and contingencies*
  - *Iterations and rendezvous*
- Scenarios are stories of what happens that is not controlled by the system
  - *They also define what the system is expected to do in response*
- Embedded systems need context observation and interpretation to help predict their environment
  - *Making internal choices and responding to external direction*
  - *These systems need to be able to build their own models*

chris.landauer@aero.org

The Real World

- Not well-modeled by any formal or systematic mechanisms
  - *No matter how good the mathematical foundation is (MAUDE, CSP, ...)*
- Mostly smooth, occasional abrupt changes (``modes'')
  - *Things break and otherwise spin out of control*
- The operational environment of any embedded system is
  - *Largely unpredictable and uncontrollable and mostly unknowable*
  - *(``the slings and arrows of outrageous fortune'')*
- Scenarios are stories of what happens that is not controlled by the system, including
  - *Activities of other agents or actors*
  - *Component failures and other errors*
  - *Unexpected environmental phenomena*
  - *Other nominal and off-nominal activities*

chris.landauer@aero.org

What is Hard

- Combination of abstract software transition models with very concrete hardware / environment models
  - *Software is about the sequencing (or partial ordering) of discrete events*
    - Concurrent and possibly distributed
    - Behavior is generally assumed to be independent of platform
  - *Hardware / environment is about continuous or even smooth processes*
    - Evolving state of the system in an uncertain environment
  - *Concept of operations is about how the system will be used by its human and other operators*
    - Scenarios illustrate various desires and expectations for the system
    - There are almost always not enough of these
- These three aspects of system development generally use completely different paradigms
  - *Many difficulties and errors in integrating them to predict system performance*

chris.landauer@aero.org

A New(ish) Approach: Com+Ode

- Combination modeling method to reduce the difficulty and increase the reliability of these modeling efforts
  - *Com is a formal notation for software modeling for simulation and verification*
  - *ODE is a collection of formal methods for solving differential equations*
  - *Story interpreter (simulation engine)*
- Integration among these model styles is explicit
  - *Interference spaces: physical, electromagnetic, resource contention*
  - *Influence mappings from each one to the others*
    - Some matter more than others

chris.landauer@aero.org

Com

- Event-based programming
  - *Developed by the presenter in mid-1980’s (see references)*
  - *Based on Hoare's CSP = Communicating Sequential Processes*
  - *Altered for better separation of processes*
    - (Hoare also made this change for his CSP book)
  - *Concurrency and synchronization*
- Hierarchical model definition
  - *Extended to allow time intervals and asynchronous interaction*
- Mathematical foundation
- Simulation and verification from the same model
  - *Translation into C for simulation*
  - *Translation into various temporal logics for verification*

chris.landauer@aero.org

Ode

- Ordinary differential equations
  - *Movement*
  - *Gradual changes*
  - *Certain other temporal effects may matter*
- State based instantaneous movement
  - *Smooth changes at various rates*
- Singularities affect the way equations can be solved
  - *Special methods are needed for solving equations near singularities*
- Different solver strategies depending on different properties of equations
  - *This is why we want explicit integration instead of implicit*
- Global time may not be definable
  - *When the system is sufficiently distributed compared to the time resolution*
  - *When propagation time becomes non-trivial compared to the rest of the computations*

chris.landauer@aero.org

A New Hard Mathematical Problem

- Real / discrete space of high and variable dimensionality
- Importance space has different and dynamic measure of significance for each coordinate
  - *``Design drivers" are an example*
- Smooth movement into a region may change the equations or just their importances
  - *Exploratory differential geometry provides some methods (simplicial complexes)*
- Software transitions may also change the space
  - *New sets of variables and constraints matter*
- Singularity indications and warnings
  - *Singularity predictions*

chris.landauer@aero.org

Other Modeling Mechanisms - AADL

http://www.aadl.info
https://wiki.sei.cmu.edu/aadl

- Predictable model-based engineering of performance-critical real-time and embedded systems
- Text notation with graphics
  - *Defined in English with reference implementations*
  - *Systematic but not mathematically formal*
  - *There are also XML descriptions that formally define the syntax (not the semantics)*
  - *Highly extensible (annexes can specialize application domain)*
- Developed by SAE specifically for manufacturing
  - *Large world-wide user community*
- Component abstractions in three categories
  - *Application software*
  - *Execution platform (hardware)*
  - *Composite (system integration elements)*

chris.landauer@aero.org

Other Modeling Mechanisms – DEVS

http://www.acims.arizona.edu/

- Text notation with graphical display
  - *Defined in English, but with a formal mapping to discrete dynamic systems*
  - *Continuous models are also possible*
- Three basic objects derived from the real world system
  - *Model, simulator, experimental frame*
- Hierarchical construction of models
- Basic model has
  - *Input and output ports, internal state variables and parameters*
  - *Time variable defines time until next internal transition (can be 0 or infinity)*
  - *Internal transition function defines state changes at transition time*
  - *External transition function defines response rules to external inputs*
    - Internal state changes and a new wait time
- Geared towards separating simulation from model, but still fairly specific to simulation

chris.landauer@aero.org

Other Modeling Mechanisms - Modelica

http://www.modelica.org
http://www.openmodelica.org

- Language for modeling of complex cyber-physical systems for simulation and other analyses
- Graphical notation and text annotation
  - *Structure from graph; behavior from text*
  - *Defined in English, but mapped into a differential-algebraic equational system, with typed variables and explicit scope and volatility and conditional equations*
- Hybrid discrete - continuous modeling
  - *Acausal (no implied order of computations)*
  - *Components, interconnections*
- Many tools exist for access to external languages
- Combined differential-algebraic equational systems may not be most appropriate
  - *Local context defined by conditional equations vs global validity*

chris.landauer@aero.org

AEROSPACE

Other Modeling Mechanisms - SysML

http://www.omg.org/spec/SysML/
http://omgsysml.org/

- Customized and extended modification of UML 2.0 for system engineering
- Graphical notation with annotations
  - *Defined in English with reference implementations*
  - *Systematic but not mathematically formal*
- Four aspects
  - *Structure = Parts and connections*
    - Hierarchy of physical or logical components and environment functionality
  - *Behavior (discrete only) defined by interaction, state machines, activity / function*
  - *Requirement relationships include hierarchies, refinements, derivation, satisfaction, verification*
  - *Parametrics are constraints on system parameter values*
- Ports include discrete data ports and continuous flow ports
  - *Rate restrictions and probabilities*

chris.landauer@aero.org

What is missing in these approaches?

- All allow internal hierarchical view of an embedded system
  - *Structure, interfaces, local state, and reactions to external interactions*
- Few expect an external view of system in environment
  - *All can model some of the relevant effects of the environment*
  - *None can model all of the relevant effects of the environment*
- Few clearly separate the model definition from its interpretation by a simulation or other analysis tool
  - *Few have formal definitions that support proofs of behavior*
  - *Few (or none) have much tool support for proofs of local behaviors in context*
  - *Systems operate in a tiny subspace of the vast possibilities defined by their parameters*
  - *Proofs of constraints are useful in limiting searches and monitoring requirements (and also for simplifying descriptions and decisions)*
- Few (or no) integration processes exist to map one approach into another, or to use a model in a different context

chris.landauer@aero.org

Modeling Process Lessons Learned

- Hierarchical modeling is extremely useful
- Early modeling can discover unexpected scenarios or definition gaps
- Model changes should always be mapped to all existing model resolutions
  - *Even back-mapping to older models that were used for analyses*
  - *This is a kind of regression model testing*
- Choosing a level of resolution adequate for the analysis at hand
  - *Usually a bit more than that required to state the analysis problem and its likely answers (sometimes a lot more)*
- Validating the relationships among different resolution levels
- The formal foundations of com+ode, DEVS, and Modelica allow some properties to be proved
  - *Then they can be used in the simulation programs and other analyses*
- Visualizations are important, but should not drive the computation
  - *There are forces and futures that we cannot see in the images*

chris.landauer@aero.org

Conclusions and Recommendations

- Commonalities and differences in these notations should be better described
  - *There may also be other notations used in other domains*
- There should be integration mechanisms to bridge their different notations and semantics
  - *Especially the basic computational models*
  - *Integration commonality proofs will remove several incompatibility barriers*
- Common model interchange formats need to be defined based on physical commonalities
  - *Since they all purport to model physical systems*
  - *Also mapping of software behaviors as discrete dynamical systems (or some other mathematical objects)*
- General modeling notations are not as useful because they are cumbersome
  - *Special purpose notations, context conditions, and explicit integration methods*

chris.landauer@aero.org

Some References

- C. A. R. Hoare, Communicating Sequential Processes, Prentice-Hall (1985)
- Leslie Lamport, ``Time, Clocks, and the Ordering of Events in a Distributed System'', Comm. ACM Vol.21 No.7, pp. 558-565 (July 1978)
- Christopher Landauer, ``Communication Network Simulation Tools'', Part 3, pp. 995-1001 in Proceedings of the 16th Annual Pittsburgh Conference on Modeling and Simulation, April 1985, Pittsburgh, Instrument Society of America (Fall 1985)
- Christopher Landauer, ``Network and Protocol Modeling Tools'', pp. 87-93 in Proceedings of the 1984 IEEE / NBS Computer Networking Symposium, December 1984, NIST, Gaithersburg, Maryland (December 1984)

chris.landauer@aero.org

AEROSPACE

Some More References

- Christopher Landauer, ``Performance Modeling of Protocols'', Paper 16.2, vol. 2, pp. 219-221 in Proceedings of MILCOM'84: The 1984 IEEE Military Communications Conference, October 1984, Los Angeles (October 1984)
- Christopher Landauer, Kirstie L. Bellman, ``Integration Systems and Interaction Spaces'', pp. 161-178 in Proceedings of FroCoS'96: The First International Workshop on Frontiers of Combining Systems, 26-29 March 1996, Munich, Germany (March 1996)
- J. Stoer, R. Bulirsch, Introduction to Numerical Analysis, 2nd ed., Translated by R. Bartels, W. Gautschi, C. Witzgall, Springer (1993)

chris.landauer@aero.org

Questions?