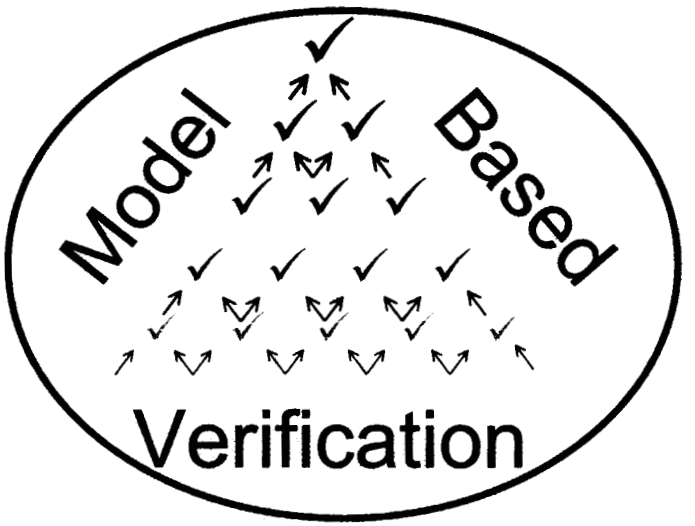
Automated Model-Based Verification of Mission-Critical Flight Software



Paula J. Pingree
Jet Propulsion Laboratory (JPL), Pasadena, CA

The Objectives

- Extend the capabilities for flight software (FSW) verification by introducing formal method model checking
- Evaluate and implement software tools that will help automate the process
- Apply tools and methods to mission-critical FSW domains implemented in Stateflow® as a prototype
- Enable new software verification technology infusion in future projects

The Approach

- Utilization of the Spin model checker with automatically translated state-charts provided as input by the HiVy tool set
- Validation examples selected and scoped to offer maximum demonstration benefit to flight projects within the capabilities of our team, tools and methods
- Translation of system design and environment models from Stateflow to Promela (the input language of Spin), integration of the closed-loop system including C-code interfaces, specification of Linear Temporal Logic (LTL) correctness properties to validate, and model checking results with Spin

What is Stateflow® ?

- Stateflow is an interactive design tool for modeling and simulating complex reactive systems
- Provided by The Mathworks; is tightly integrated with Simulink and MATLAB
- Based on Finite State Machine Representation
  - State Based
  - Formal
  - Graphical

... a.k.a **State-charts**

More on State-charts

The Structure of State-charts

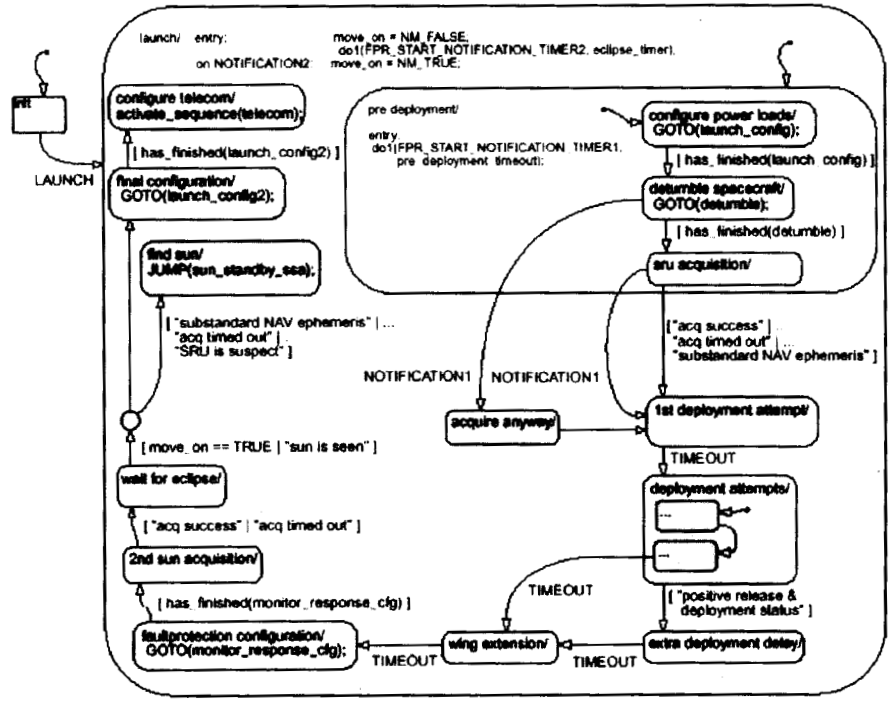
- States
- Transitions
- Conditions
- Events (or Transition Labels)
- Connective Junctions
- Data
- Actions

The Behavior of State-charts

- States have boolean behavior (active or inactive)
- Transitions define the logic flow of a system
- Events control the execution of the state; when a transition is labeled with an event name, the transition is only valid when that event occurs
- Actions are associated with either transitions or states and include assignments, function calls, and event broadcasts

FSW Development Using State-charts

- DS1's "13th Technology"
  - Model-based code generation of Fault Protection (FP) Monitors & Responses
  - Accomplished using Stateflow and Stateflow Coder by The Mathworks
  - Highly successful implementation



A section of the launch statechart, showing sun acquisition and pre-deployment of the DS1 solar array panels.

More on Development with State-charts

- Benefits:
  - Enforces standard diagrammatic conventions
  - Allows design, implementation and simulation by Systems Engineers
  - Provides concise design notation for easier review
  - Open and customizable architecture exists for auto-code generation
- However! - verification of auto-generated code follows traditional testing methods
  - Iteration for bug fixes occurs downstream in development cycle
  - Test case generation process is informal

Question:

Can we take advantage of model-based verification methods?

Summary of Spin

- Spin is a widely distributed software package that supports the formal verification of distributed systems
- The software was developed at Bell Labs in the formal methods and verification group starting in 1980
- Promela (Pro*cess* Me*ta* La*nguage*) is the Spin input language
- The Spin software is written in ANSI standard C, and is portable across all versions of the UNIX operating system. It can also be compiled to run on any standard PC running Linux, Windows95/98, or WindowsNT.
- http://netlib.bell-labs.com/netlib/spin/whatispin.html

Summary of HiVy

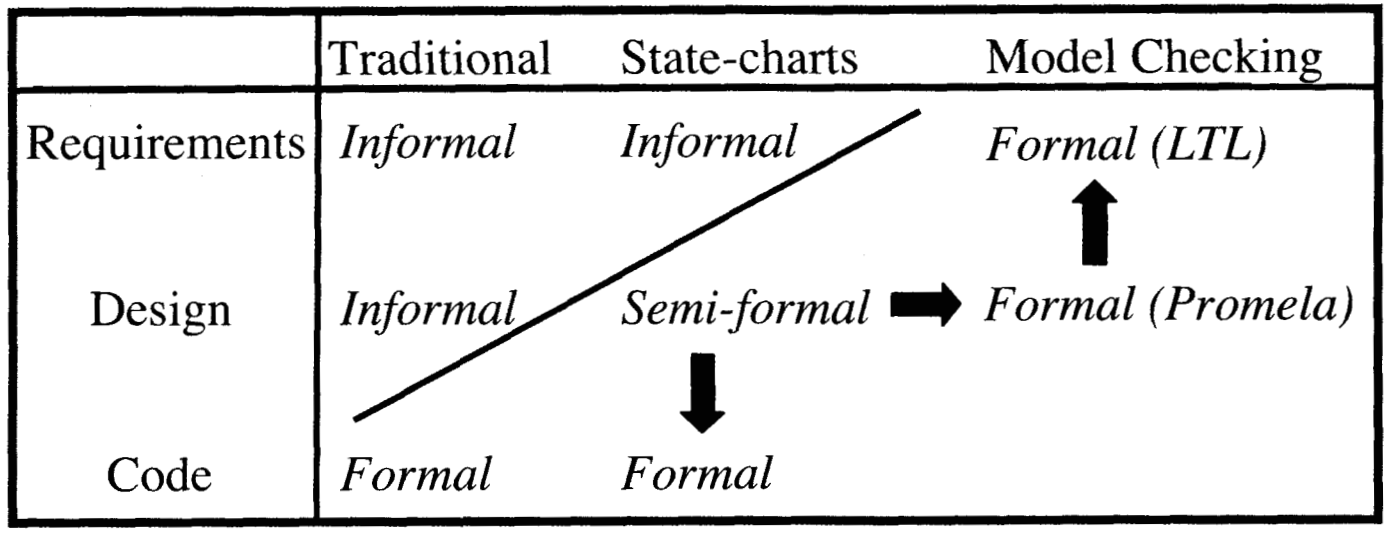
- HiVy is based on the new Hierarchical Sequential Automata (HSA) format and provides automatically translated models for input to Spin
- HiVy was developed by JPL and Erich Mikk (independent consultant) in FY02
- The HiVy toolset consists of the programs:
  - *SfParse* extracts pertinent data from the Stateflow model file
  - *sf2hsa* translates parsed output into HSA (intermediate format)
  - *hsa2pr* translates HSA into Promela
  - and the HSA Merge Facility

Summary of HiVy - 2

- HiVy translated models consist of sequential automata and their hierarchy and parallel composition
- HiVy translation handles the semantics of inter-level transitions and the priority rule of state-charts
- The communication media in HiVy translated models as in state-charts is instantaneous broadcast of uninterrupted events received either from the environment or as the result of taking a transition

A New Approach

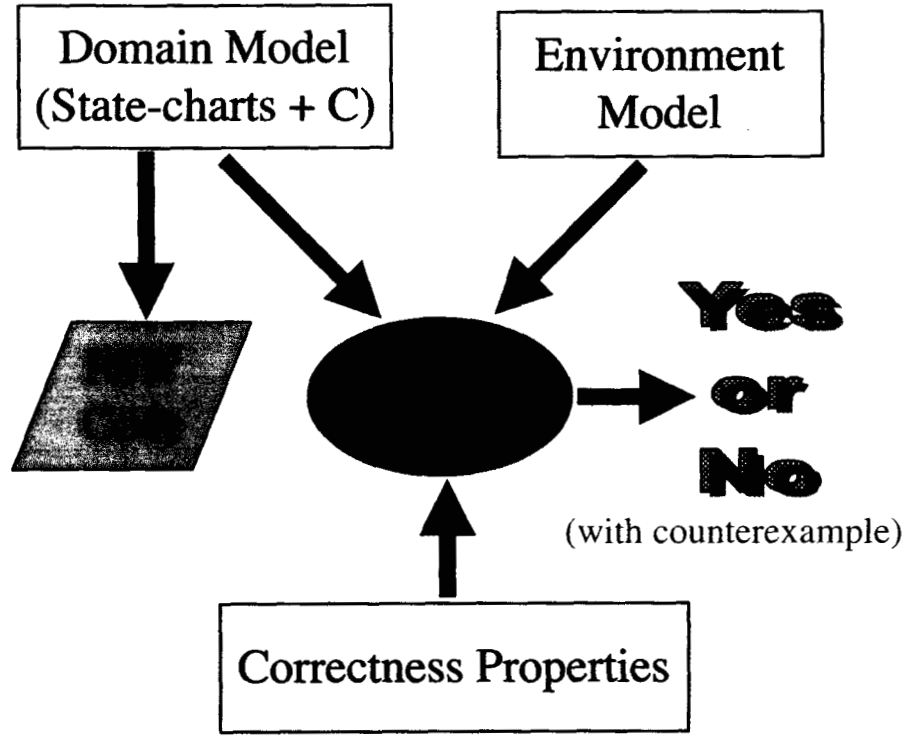
- Apply "Lightweight" Formal Methods to FSW Verification
- Use the Spin Model Checker
  - Developed at Bell Labs by Dr. Gerard Holzmann
  - Spin can exhaustively examine the state space of a model and detect violations of the user-specified properties, e.g. unreachable states



Applying Model Checking to FSW

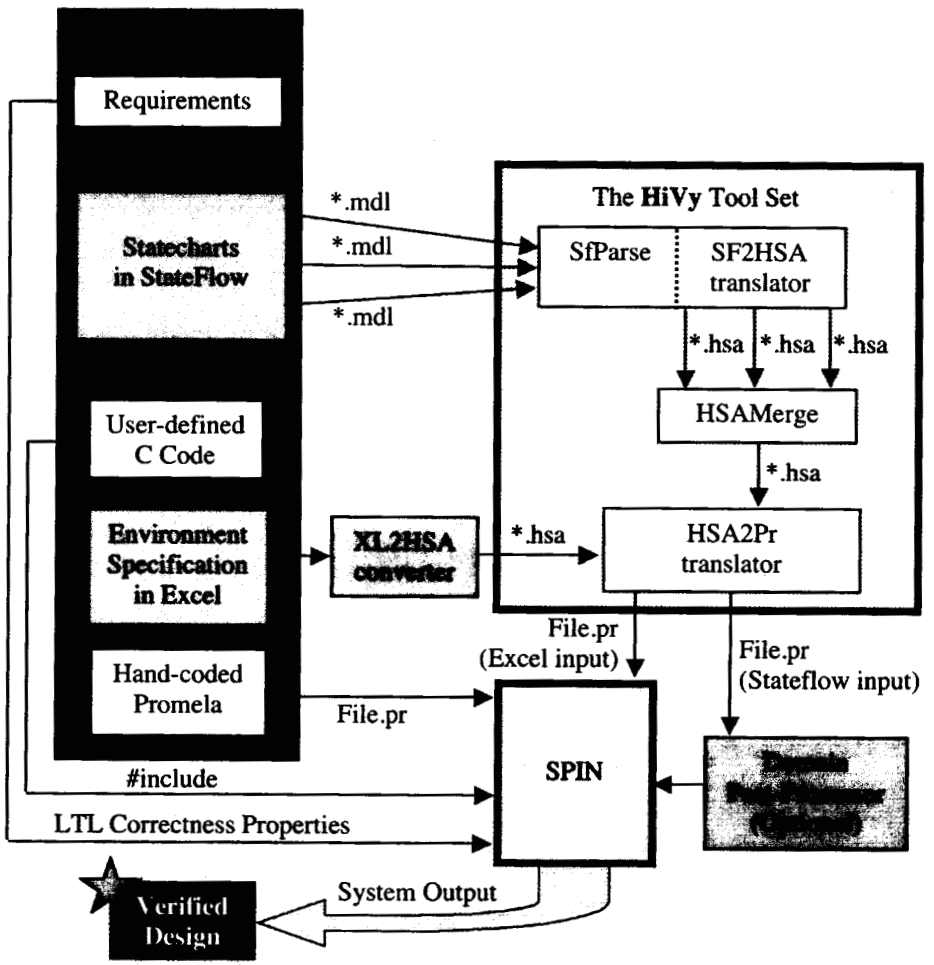
- We provide automated translation of the state-chart model from Stateflow to Promela, the input modeling language of Spin

- Key Benefits:
  - SPIN validation model and FSW code, now both auto-generated, have the same source (the Stateflow statechart)
  - Validation of statechart design can occur earlier in development cycle and without use of valuable testbed resources



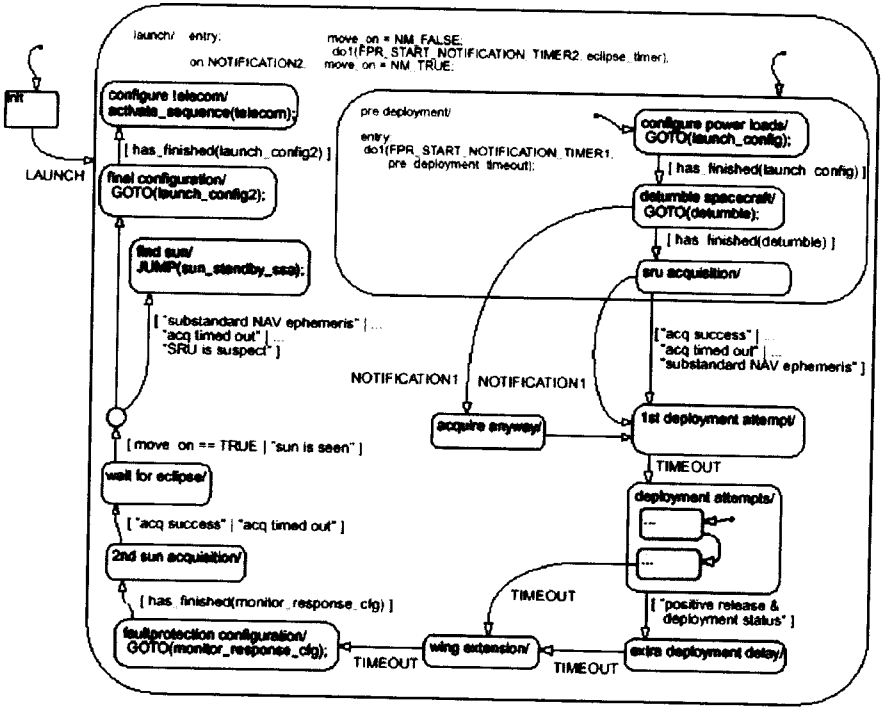
- *Based on Requirements*
- *Expressed in Linear Temporal Logic (LTL)*

HiVy Tool Set & Interfaces



HiVy Features

1. 'SfParse' recognizes all relevant elements from Stateflow *.mdl files
2. 'sf2hsa' transforms parsed elements into HSA for further processing
3. 'hsa2pr' program supports:
   - Sequential automata with states, transitions and default transitions
   - Transition labels with conditions and actions varying over boolean and integer variables
   - Hierarchy
   - AND-states
   - Event handling
   - Inter-level transitions
   - Junctions
   - Condition actions
   - User-defined C code
4. HSA Merge facility allows integration of multiple HSA files into a single Promela model upon translation



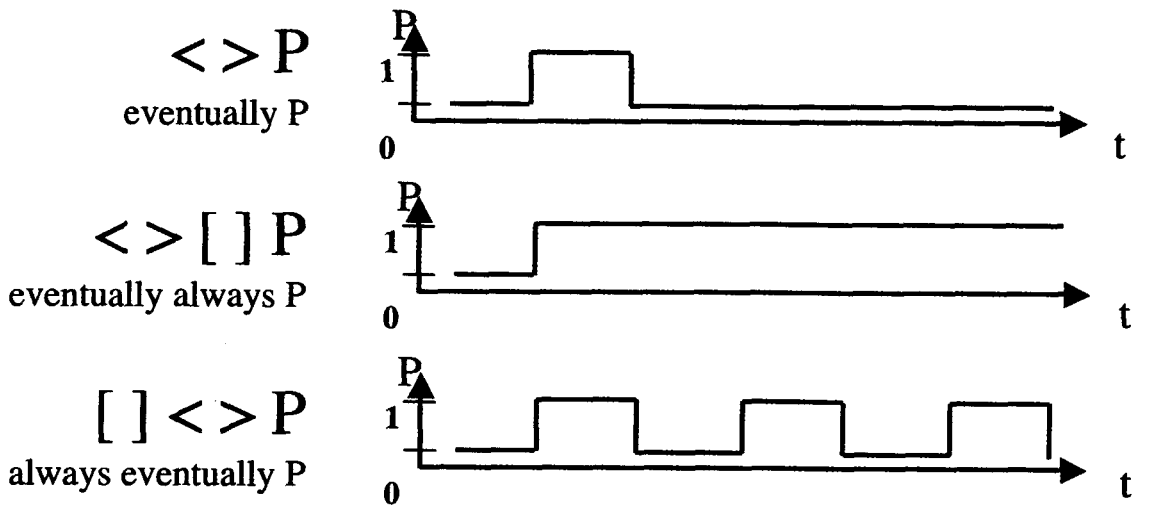
A section of the launch statechart, showing sun acquisition and pre-deployment of the DS1 solar array panels.

Translation Process

- make <file>.pr

Inputs:	<files>.mdl	
Intermediate files:	<files>.sf <files>.hsa	
Outputs:	merge.pr	: contains Promela model of translated statecharts
	propositions	: describes propositions generated for each state and each event
	prop_list	: list of proposition names only (not their definitions); suitable for automatic generation of correctness properties

Verification: Correctness Properties



- Correctness Properties (CP) are formal statements of the expected behavior of a system
- The accuracy of verification results depends on the accuracy and completeness of the CPs
- CP events and states must be linked to concrete events and states in the model

hsa2pr produces prop_list for generating CPs

Success Criteria

- Spin model checking validation findings for several flight system designs:
  - 2004 Deep Impact (DI) Fault Protection (FP)
  - 2003 Mars Exploration Rover (MER) Surface Activity & Resource Arbitration
  - Mission Data System (MDS) Rocky 7 Adaptation system design (portions thereof)
- Honest assessment of model checking applicability to each domain
  - Lessons learned
  - Limitations
  - Recommendations for continued work
- Improved and documented processes for our verification approach along with Flight Project advocacy from DI, MER & MDS
- Position model checking and supporting automated verification tools toward mainstream use within a future JPL flight project - targeting 2009 Mars Smart Lander (MSL)

Publications

- K. Barltrop, P. Pingree, Model Checking Investigations for Fault Protection System Validation. *Paper accepted for 2003 International Conference on Space Mission Challenges for Information Technology,* July 2003
- P. Pingree, E. Benowitz, Experiences in Integrating Auto-Translated State-Chart Designs for Model-Checking. *Paper accepted for 2003 Workshop on Model-Checking for Dependable Software-Intensive Systems, June 2003*
- P. Pingree, E. Mikk, G. Holzmann, M. Smith, D. Dams, Validation of Mission Critical Software Design And Implementation Using Model Checking. *The 21st Digital Avionics Systems Conference,* October 2002.
- P. Pingree, E. Mikk, The HiVy Tool Set. *Pending NASA Tech Brief Publication.*

References

- D. Harel, Statecharts: A Visual Formalism for Complex Systems. *Science of Computer Programming*, ASSP-34(2):362, 1986.
- E. Mikk. HSA-Format, *private communication* 2002.
- E. Mikk, Semantics and Verification of Statecharts. PhD Thesis. *Technical Report of Christian-Albrechts-University in Kiel,* October 2000.
- E. Mikk, Y. Lakhnech, M. Siegel and G. Holzmann, Implementing Statecharts in PROMELA/SPIN. In *Proceedings of the 2nd IEEE Workshop on Industrial-Strength Formal Specification Techniques.* pages 90-101. IEEE Computer Society 1999.
- G.J. Holzmann. The Model Checker Spin. *IEEE Trans. on Software Engineering*, 23(5):279-295, May 1997. Special issue on Formal Methods in Software Practice.
- The Mathworks Stateflow Users Guide, http://www.mathworks.com