

Composing, Analyzing and Validating Models to Assess the Performability of Competing Design Candidates



Electrical Engineering and
Computer Science Department

CS 500 ProSeminar

April 17, 1999

Frederick T. Sheldon
Assistant Professor of Computer Science
Washington State University

Agenda

Goal: *Verification* and *validation*
of *systems* and *software*

Modern high-assurance systems

Advantages of a formal approach

How do we get there from here: **Modeling Cycle**

Safety and reliability analysis:

Railroad Switching System *including* **Design-to-Cost**

Braking/Traction/Steering Control System

Operating System with Dynamic Priority Mechanism

Summary of ongoing work



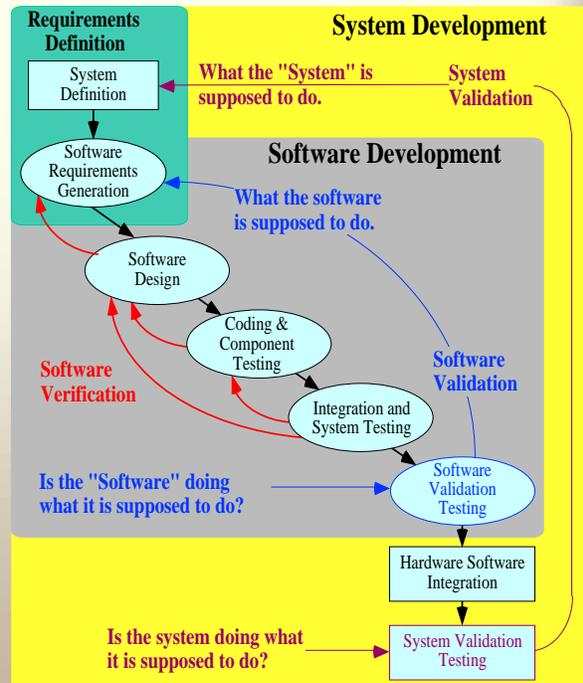
Verification and Validation

Verification determines if the products of a given phase of the **SW life cycle** fulfill the **requirements** established during the previous phase.

Formal proof of **program** correctness

Reviewing, inspecting, **testing**, checking, auditing, or otherwise establishing and documenting whether or not items, **processes**, services, or **documents** conform to specified **requirements** (ANSI/ASQC A3-1978).

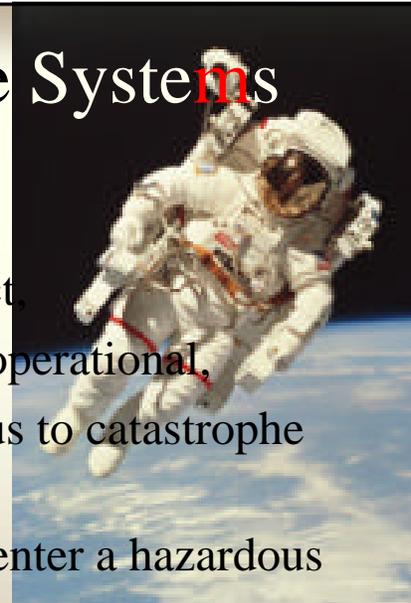
Validation checks *if* the program, as implemented, meets the expectations of the customer in such a way to ensure compliance with software requirements.



Modern High-Assurance Systems

Share five key attributes:

- Reliable**, meaning they are correct
- Available**, meaning they remain operational,
- Safe**, meaning they are impervious to catastrophe (fail-safe),
- Secure**, meaning they will never enter a hazardous state,
- Timely**, meaning their results will be produced on time and satisfy deadlines (timing correctness).



Advantages of Formal Specification

Provides insights into the requirements / design

Specifications may be analyzed mathematically

Demonstrate *consistency* and *completeness*

Prove the implementation corresponds to the specification

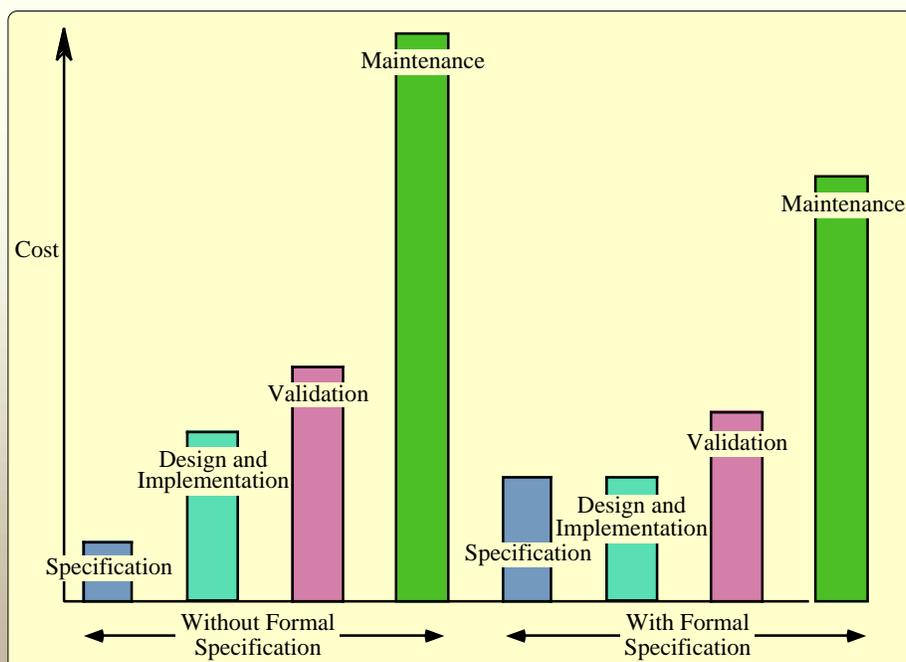
Help identify appropriate *test cases*

Characterize aspects of the specification more precisely:

- Structural, Functional, and Logical
- Behavioral
 - Dynamic: timing combined with probabilistic nature
- Data oriented.

And, the potential for cost savings....

Expenditure Profile Changes



The Vision

Methods and tools are needed for the creation of safe and correct systems. . .

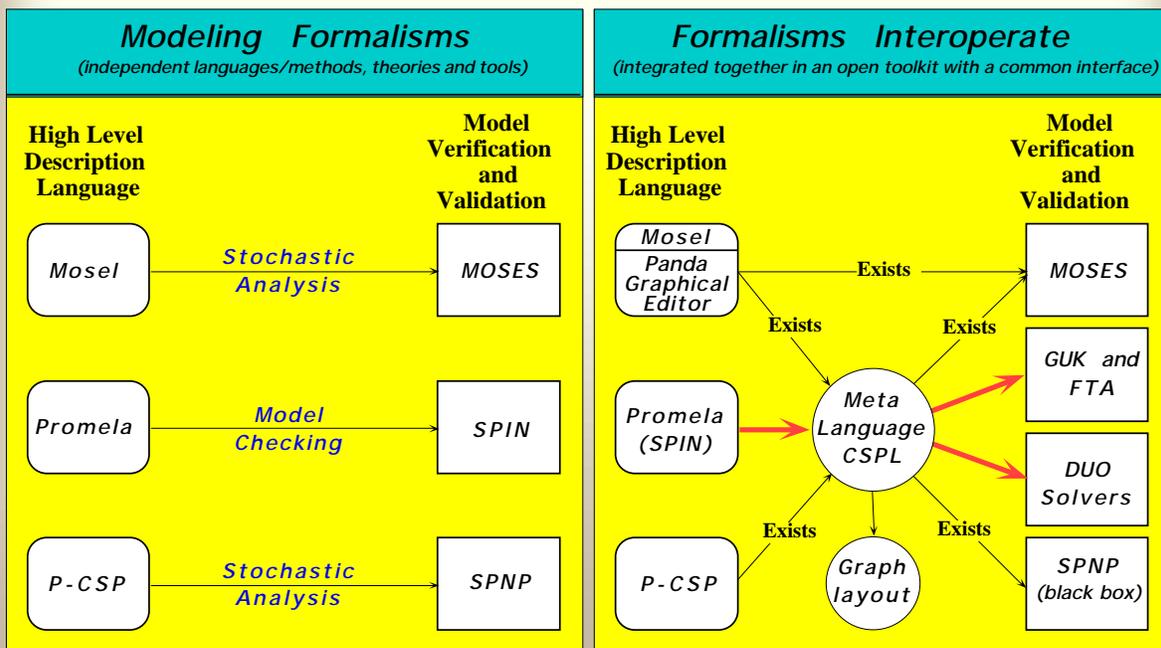
Reduce the effort of constructing reliable models for . . .

- Application level safety, performance and reliability analysis
- Improved tractability for verifying correctness and for solving large stochastic models
- Reasoning about unambiguous specifications and designs

Need for an integrated environment to provide interoperability among formalisms

- Link stochastic analysis with correctness checking
- Allow various formal methods to be applied independently based on a common representation form.
- Demonstrate on industrial strength problems
- Learn what works and what doesn't

Integrated Environment to Provide Interoperability



The Modeling Cycle

Descriptive modeling

Computational modeling

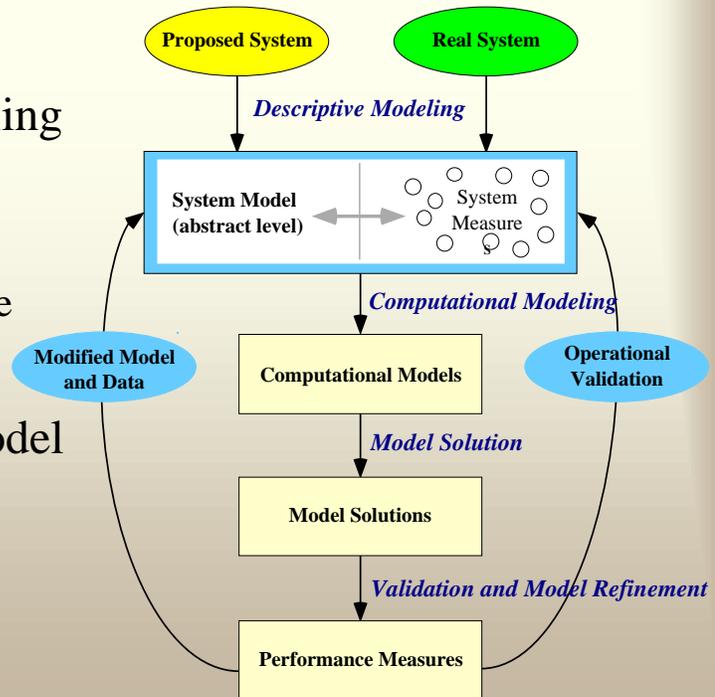
Making it tractable

Model solution

Validation and model refinement

Operational

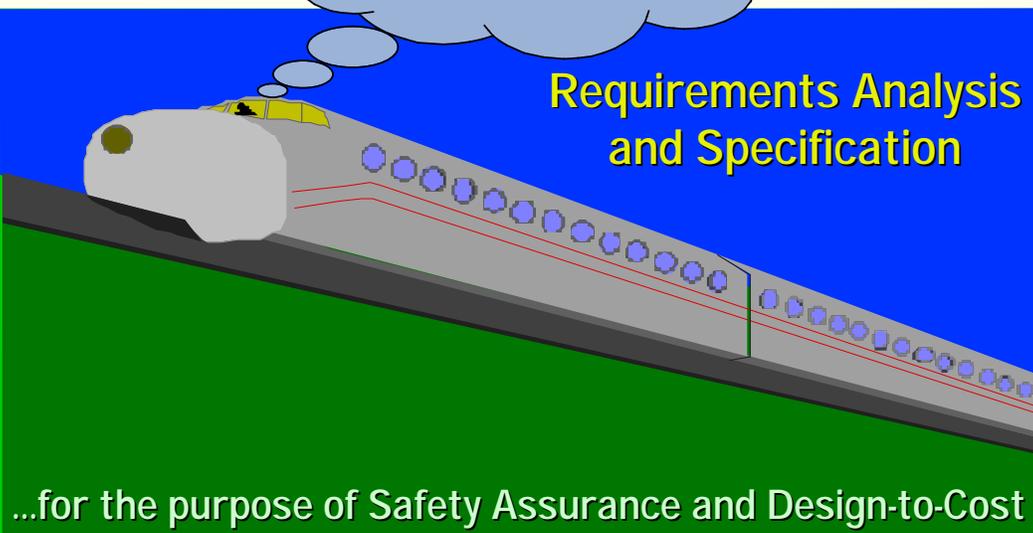
Proposed



Railway Switching System

Hope that gate closes in **time!**

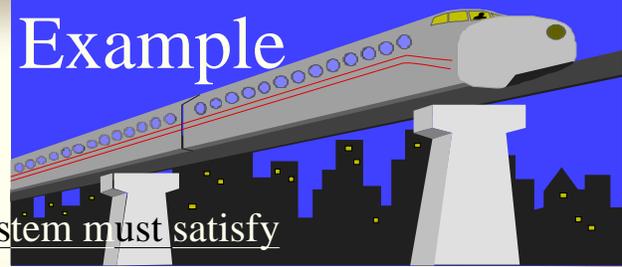
Requirements Analysis
and Specification



...for the purpose of Safety Assurance and Design-to-Cost

Railway Switching Example

Requirements:



Two Basic Properties the system must satisfy

Safety property – the gate is down during all occupancy intervals

Utility property – the gate is open when no train is in the crossing

The Solution in General Terms:

- + Two Processes: The TRAIN and the GATE
- + TRAIN sends an "arriving" signal to the GATE as it nears the intersection and proceeds towards the intersection.
- + GATE, upon receiving the signal, closes the gate and remains closed until the train departs.
- + TRAIN sends a "departing" signal after leaving the intersection.
- + GATE, upon receiving the signal opens the gate and remains open.
- + The two processes repeat continuously.

This model encompasses the environment which includes the train(s) and the gate, as well as the interface between them.

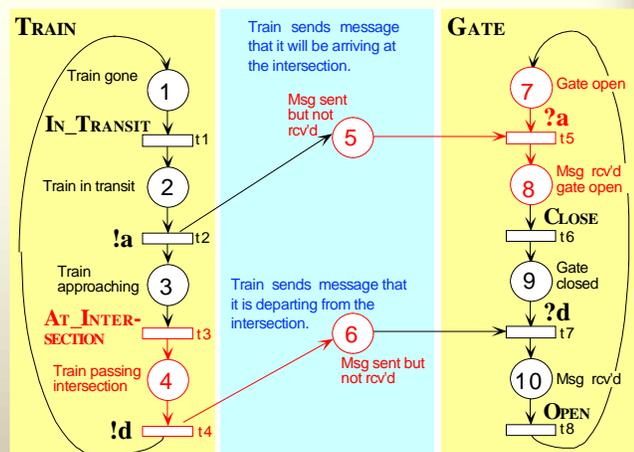
Compose a Functional Model Using the Process Algebra CSP translated to SPNs

```

TRAIN =
  (IN_TRANSIT);
  (GATE ! a  AT_INTERSECTION);
  (GATE ! d  TRAIN)

GATE =
  (TRAIN ? a  CLOSE);
  (TRAIN ? d  OPEN  GATE)

RAIL_ROAD_CROSSING = TRAIN ||_{a,d} GATE
    
```



- + **Problem:** A hazard exists which becomes *more* evident viewed as a Petri net

Several possible failure modes exist: (1) communication failure [t₂, t₄, t₅ and t₇], (2) mechanical failure [t₆ and t₉], and (3) timing failure [t₃ occurs before t₆] (i.e., train arrives at intersection before the gate has closed).

Refined System Model

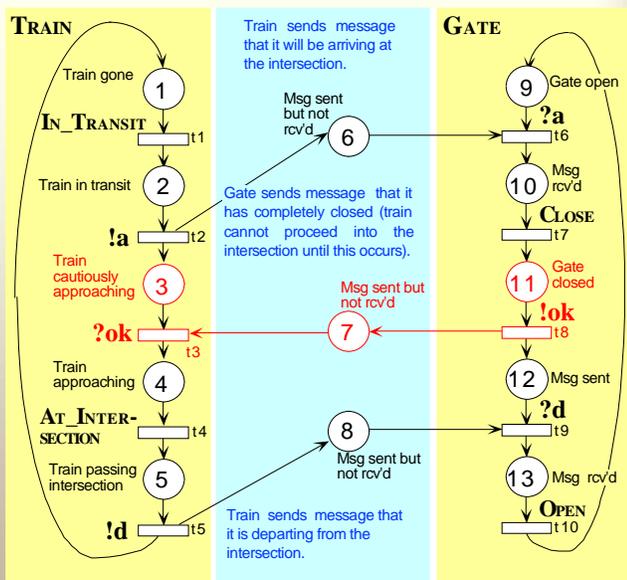
Hazard Removed

```

TRAIN =
  (IN_TRANSIT);
  (GATE ! a  GATE ? ok
   AT_INTERSECTION);
  (GATE ! d  TRAIN)

GATE =
  (TRAIN ? a  CLOSE  TRAIN ! ok);
  (TRAIN ? d  OPEN  GATE)

SAFER_RAIL_ROAD_CROSSING =
  TRAIN ||_{a,ok,d} GATE
  
```



Lower Level Abstraction

Timing hazard version

Mechanical Failures

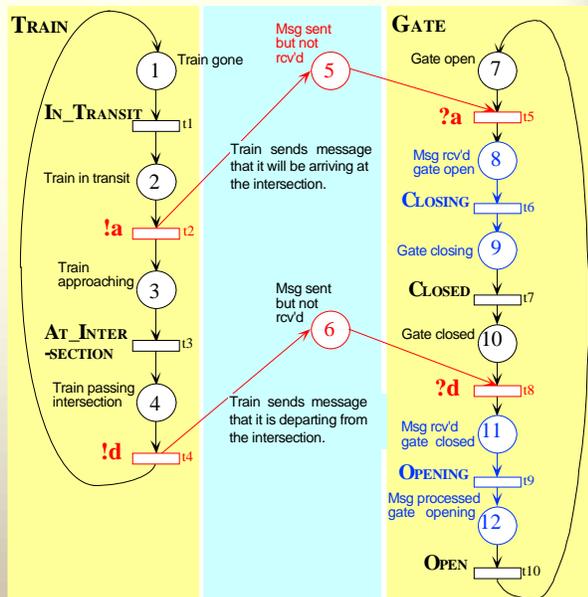
Safety Critical (closing)

Cost Critical (opening)

Communication Failures

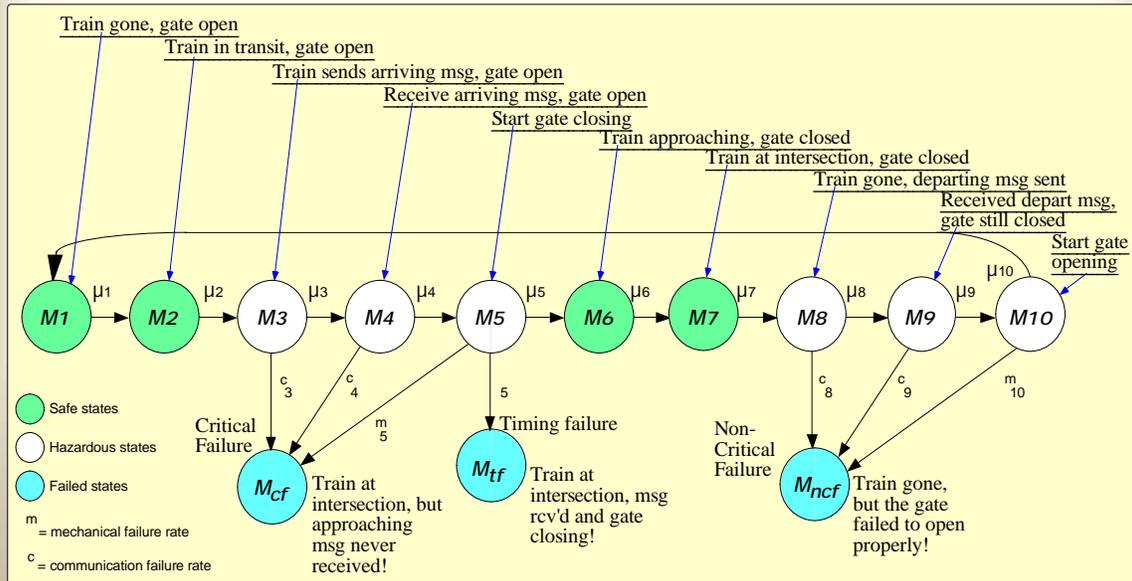
Safety Critical (arrival message [and OK message])

Cost Critical (departing message)

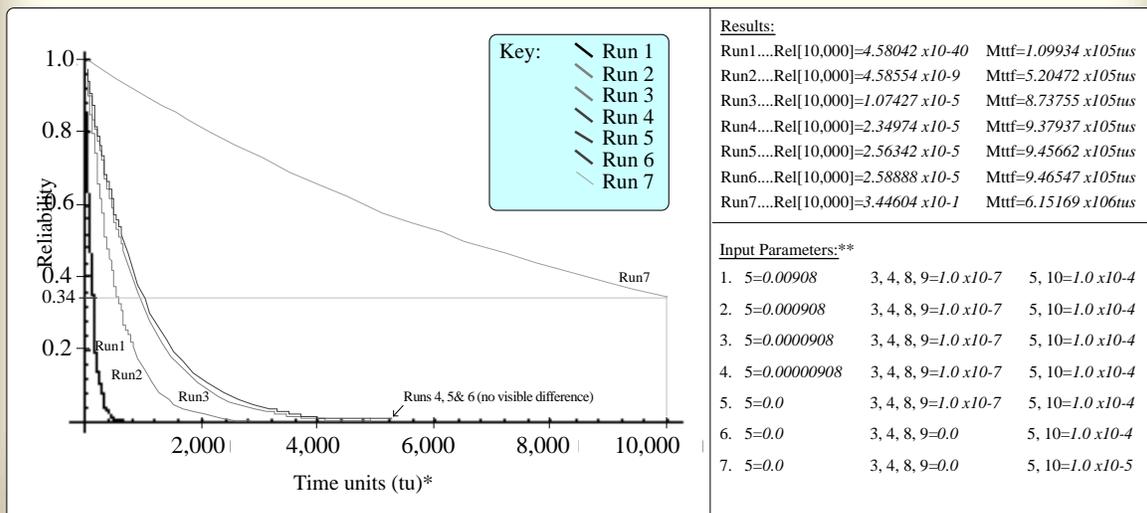


Several possible failure modes exist: (1) communication failure [t₂, t₄, t₅ and t₈], (2) mechanical failure [t₆ and t₉], and (3) timing failure [t₃ occurs before t₇] (i.e., train arrives at intersection before the gate has completely closed).

Generate the ERG/RG Markov



Reliability Prediction

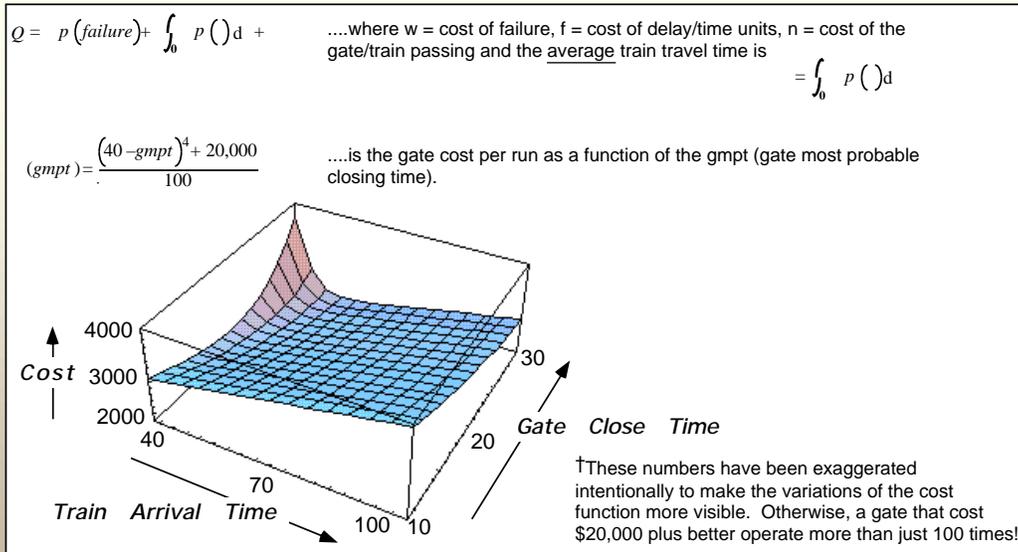


*Time units: each x-axis tick is 1000tus. If 1 tu = second, then ~16mins/tick, or 10,000 ticks ~2778hrs (full range of data).

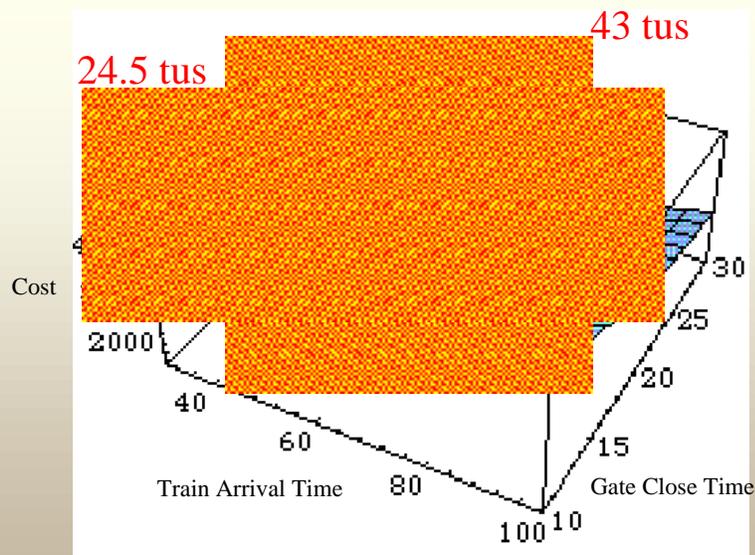
**Constants: $\mu_1=0.0001$, $\mu_2=4, 7, 8=1.0$, $\mu_9, 10=1.0$, while μ_5 and $\mu_6=$ were held set at 0.1 and 0.01 respectively.

Design-to-Cost

Evaluate (judiciously) the costs (and benefits) for providing fault-avoidance and/or fault-tolerance using a cost function to optimize design parameters.



Costs May Be Correlated to Design Parameters



Braking/Traction/Steering Control

System



TC/ABS Functional Description

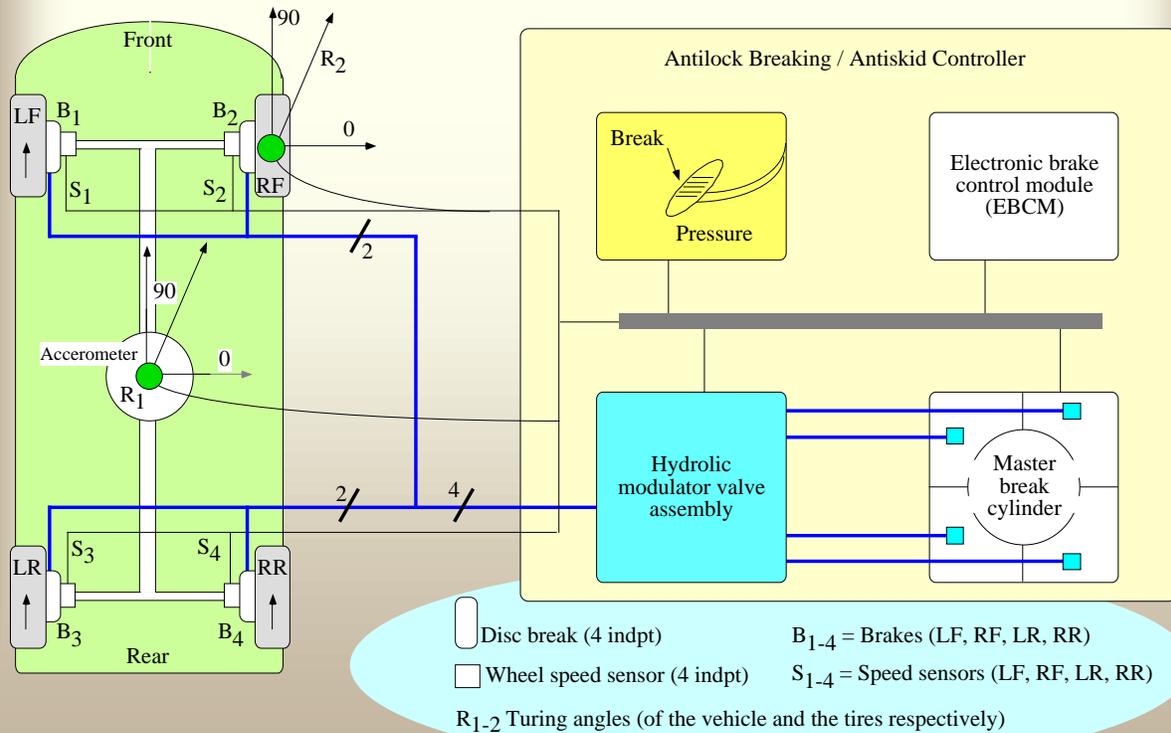
(Traction Control / Antilock Brake System)

ABS maintains steer-ability and driving stability under skidding conditions

Anti-Slip control maintains adhesion to the road and driving stability

Electronic Stability program maintains limits among yaw-rate, steering-angle, and lateral velocity preventing under/over-steer

TC/ABS Schematic



Skid+Steering Control System

If Any-Wheel-Locks then

Pulsate-Locked-Wheel

If Either-Rear-Wheel-Slips then

Brake-Slipping-Wheel

If Under-Steer-Left then

Brake(Left-Front, Left-Rear)

If Under-Steer-Right then

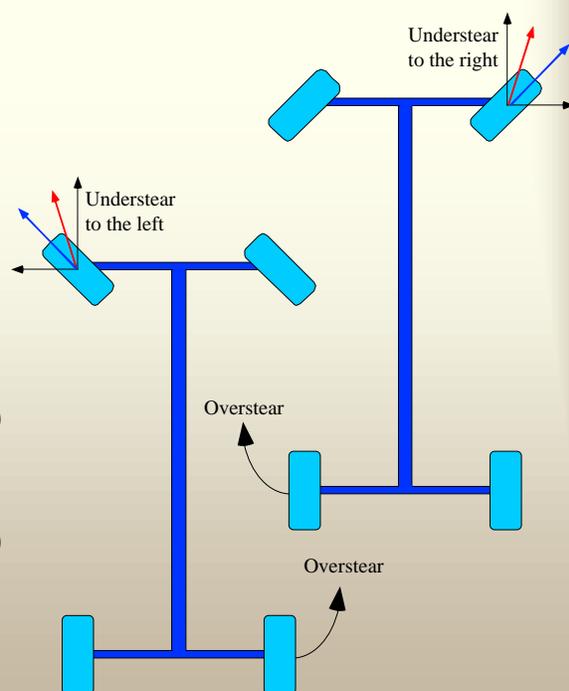
Brake(Right-Front, Right-Rear)

If Over-Steer-Left then

Brake(Right-Rear, Right-Front)

If Over-Steer-Right then

Brake(Left-Rear, Left-Front)

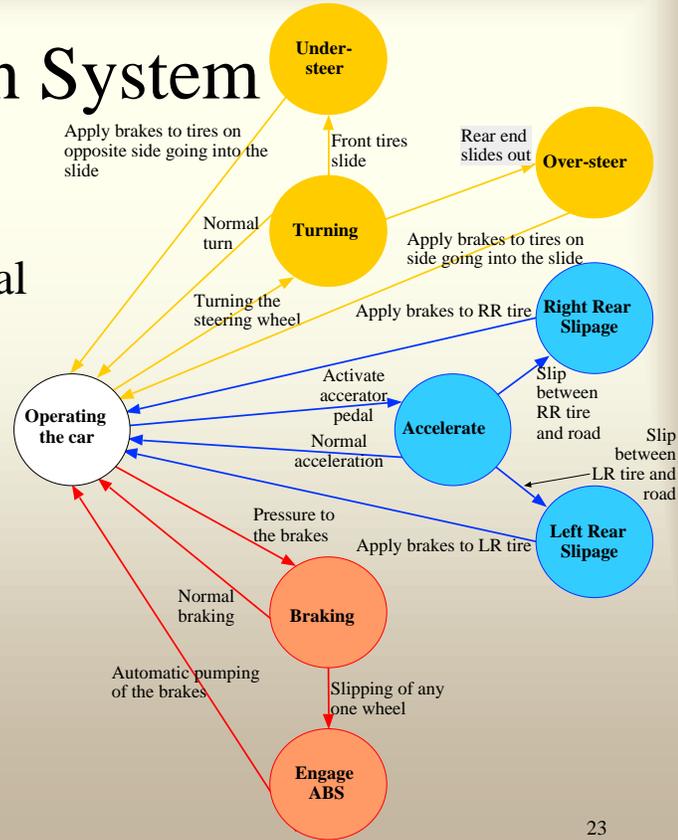


State Transition System

Deciding how the faults affect nominal and off nominal operation

Failure modes

- Loss of vehicle
- Loss of stability
- Degraded function
- Over/Under-steer



Entity Life History Diagram

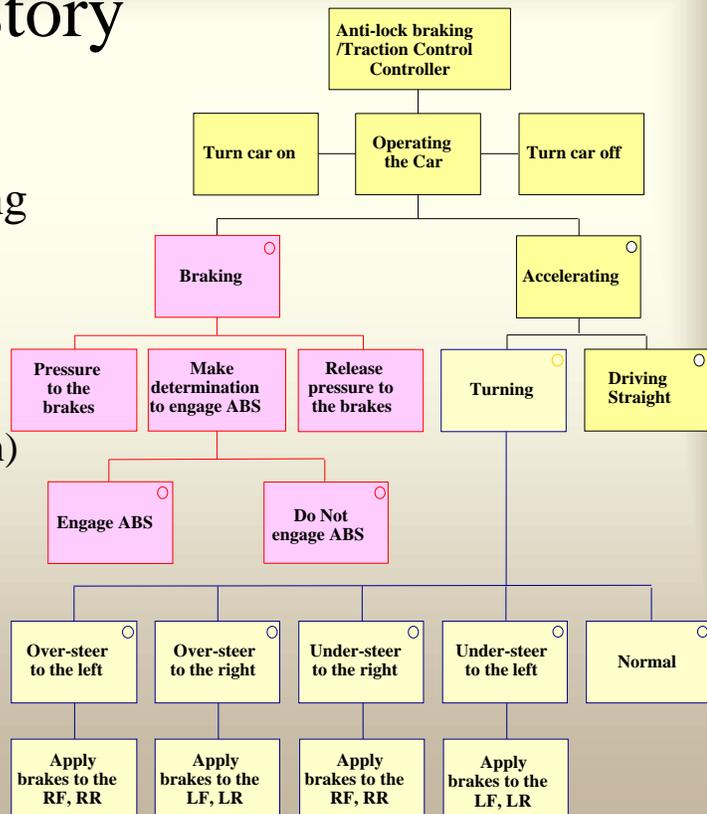
Descriptive Modeling

View of the system

- Braking
- Steering
- Skidding (not shown)

Structure Chart

- Invocation structure
- Choices (pathways)
- Flow



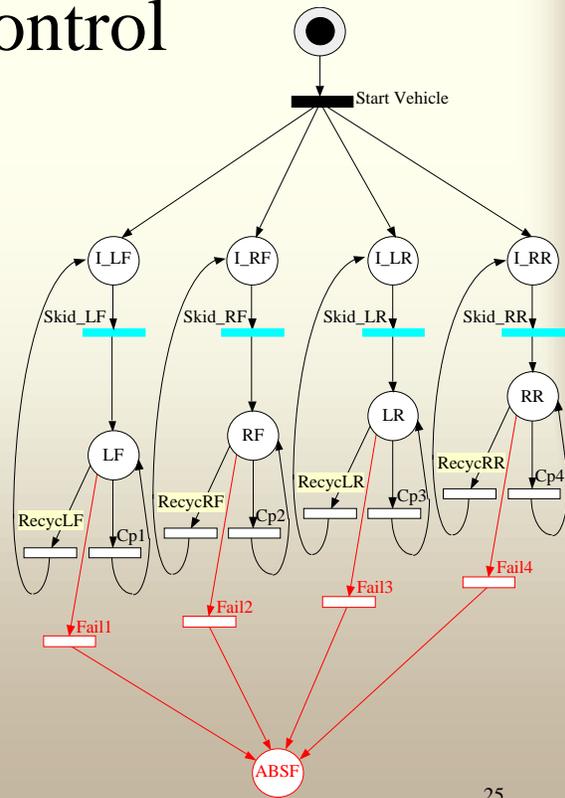
ABS Skidding Control

Computational Modeling
Skidding of any tire may be detected

Compensation mechanism cycles (loop counter-clock-wise) until skidding ceases

Fault may occur activating a failure mode causing:

- Loss of vehicle
- Loss of stability
- Degraded function
- Over/Under-steer



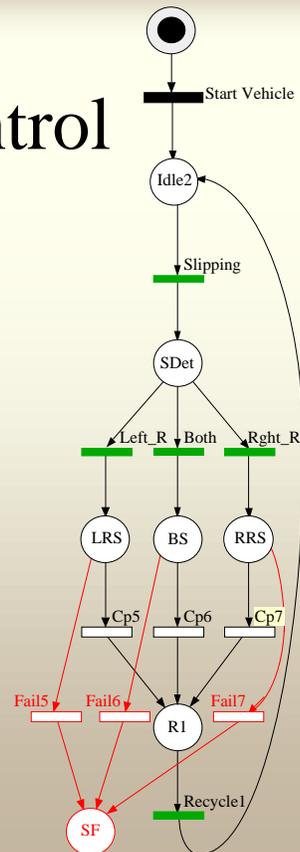
Slipping/Traction Control

Rear wheels lose traction

Compensation mechanism is one shot process

Fault may occur activating a failure mode causing:

- Loss of stability
- Degraded function



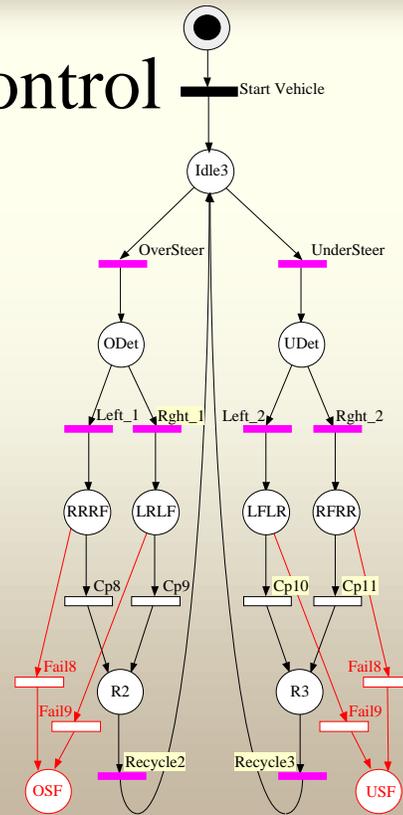
Over/Under-Steer Control

When over/under-steer threshold is detected

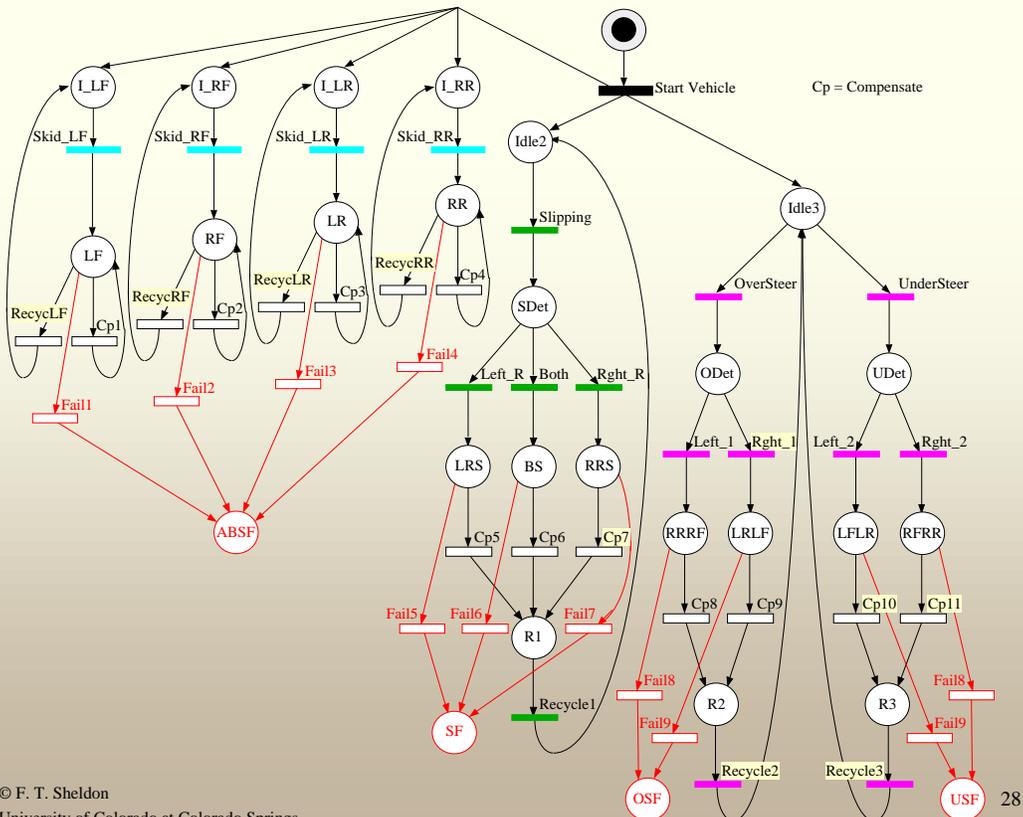
Compensation mechanism is a one shot process

Fault may occur activating a failure mode causing:

- Loss of stability
- Degraded function
- Over/Under-steer



TC/ABS Combined



Derive Failure Rate

Mappings

Determine causality

Fault

Symptom

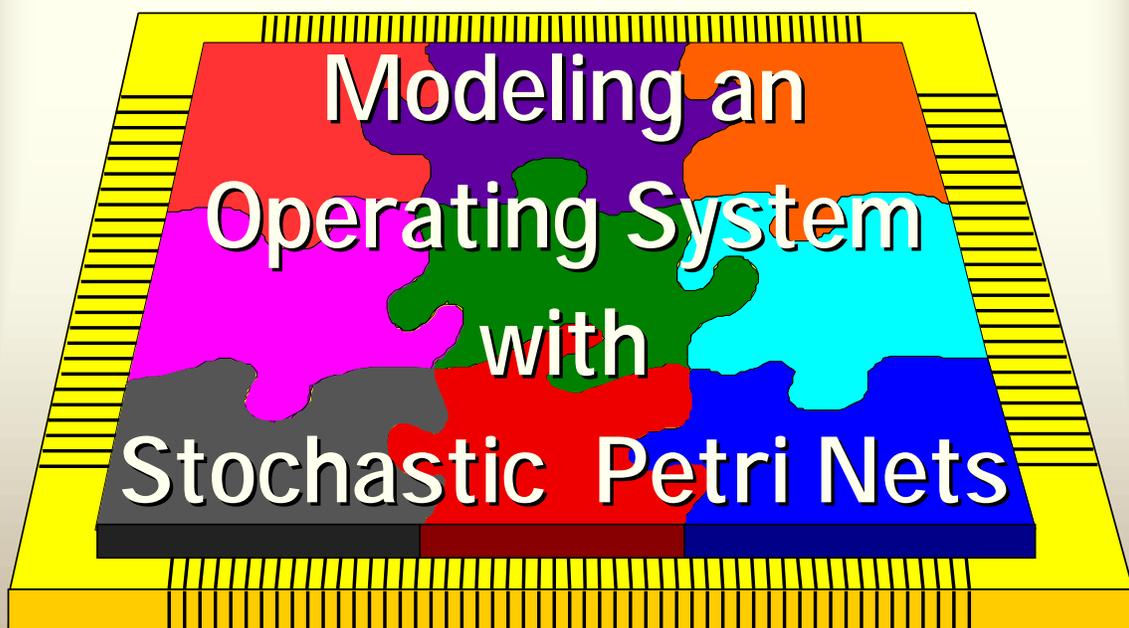
Suspect component

Calculate cumulative failure rates

Assign to failure transitions in SPN

Fault >	One Wheel (PL)	One Wheel (LB)	One Axle (PL)	One Axle (LB)	Both Axles (PL)	Both Axles (LB)
Symptom >	Degraded Function	Over/Under-Steer of the Car	Loss of Vehicle	Loss of Stability	Loss of Vehicle	Loss of Stability
Component						
Wheel Speed Sensor	2.00E-10	2.00E-10				
Pressure Sensor			1.50E-10	1.50E-10	1.50E-10	1.50E-10
Main Brake Cylinder			1.00E-10		1.00E-10	
Pressure Limiting Valve			6.00E-12	6.00E-12		
Inlet Valve			6.00E-12	6.00E-12		
Drain Valve			6.00E-12	6.00E-12		
Toggle Switching Valve	6.00E-12	6.00E-12				
Hydraulic Pump			6.80E-10		6.80E-10	
Pressure Tank					2.00E-11	
Controller	6.00E-11	6.00E-11	6.00E-10	6.00E-11	6.00E-11	6.00E-11
Steering Angle Sensor						
Lateral Accel Sensor						
Yaw Rate Sensor						
Tubing	3.00E-11		3.00E-11		3.00E-11	
Piping	4.00E-11		4.00E-11		4.00E-11	
Cumulative Failure Rate	3.36E-10	2.66E-10	1.62E-09	2.28E-10	1.08E-09	2.10E-10

© F. T. Sheldon
University of Colorado at Colorado Springs



© F. T. Sheldon
University of Colorado at Colorado Springs

Dynamic Priority OS

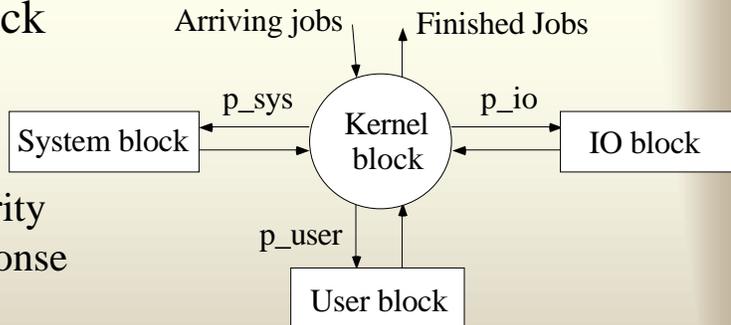
Functional Level Abstraction

Each elementary block

Analytic Sub-model

Dynamic Priorities

Guarantee high priority jobs get shorter response times



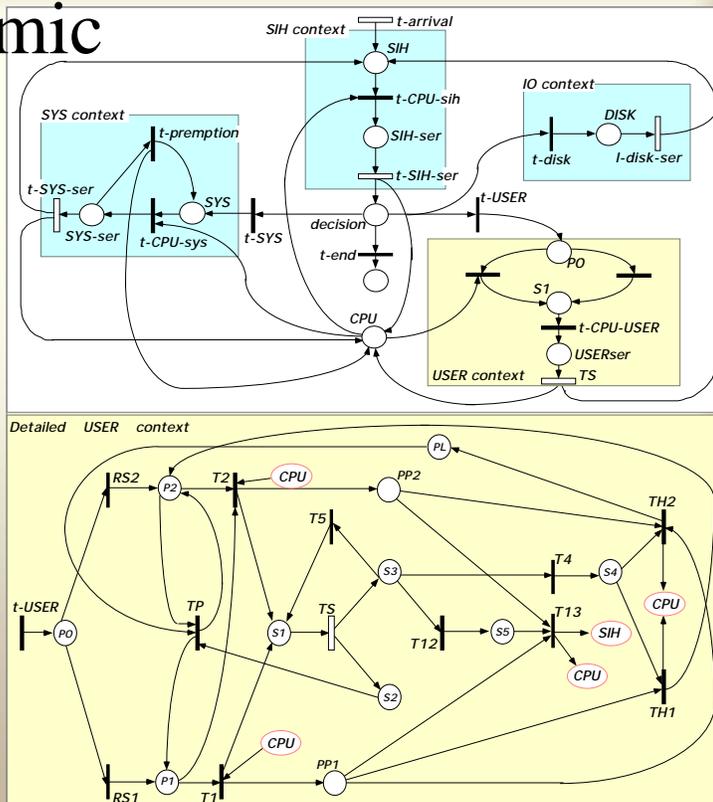
❁ Goal: Evaluate dynamic increasing/decreasing priority assignments.

SPN of Dynamic Priority OS

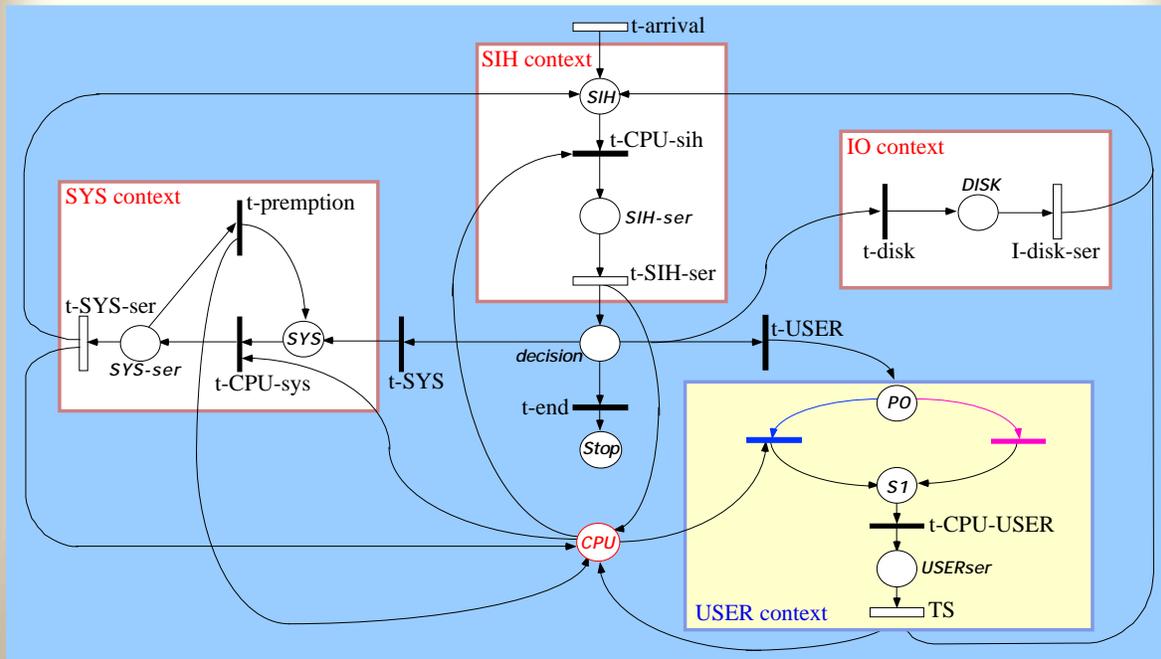
Top: complete system contexts

Kernel (SIH)
System (SYS)
IO
User

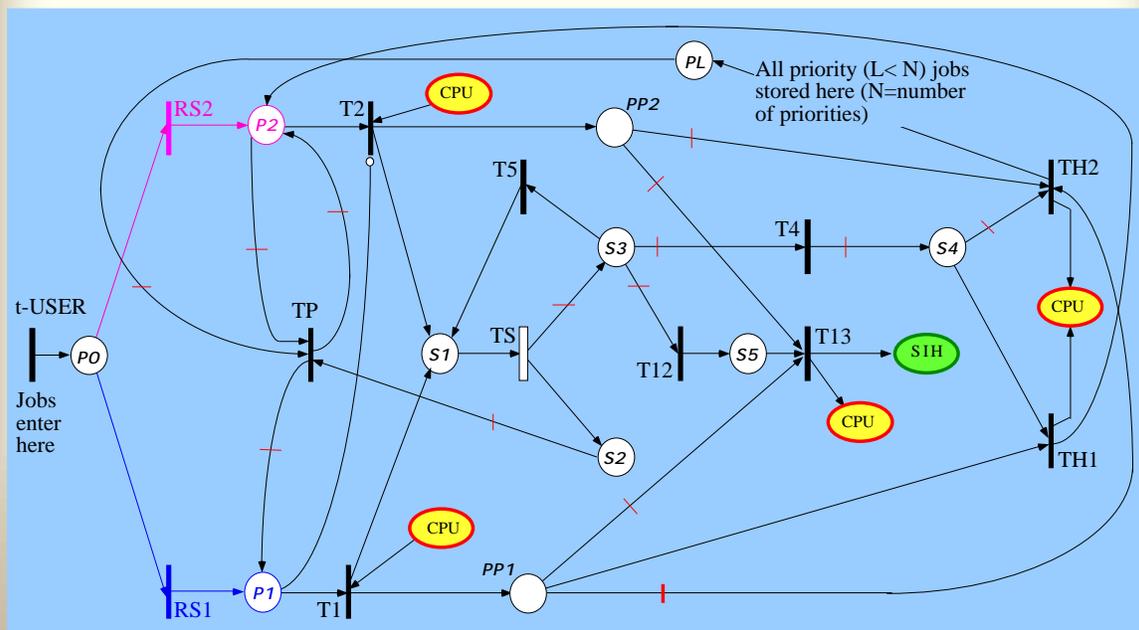
Bottom:
Detailed User Context



Complete System SPN



Detailed User Context



User Context: Basic Characteristics

Lower priority than other contexts

Gets CPU when there are no jobs to be processed in other contexts.

Lower priority is assigned to transitions T_i ... than to transitions t_CPU_sys and t_CPU_sih .

Transitions T_i ... enabled when no other jobs are being served number of tokens in places $PP_i = 0$.

When transition T_i ... fires a token in the CPU place is removed.

Jobs are processed in priority order.

Inhibitor arc from $P1 (P_i)$ to $T2 (T_{i+1})$ guarantees a priority class i job is processed before class $i+1$.

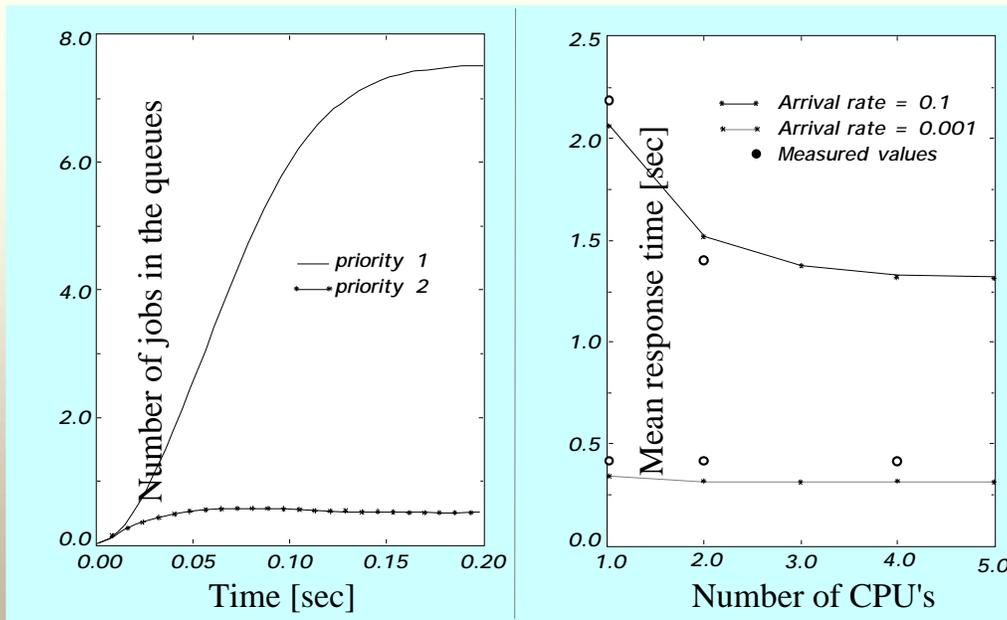
Token in $S1$ the CPU is processing a USER context job of priority i (by token in PP_i).

System Parameters

<i>System Parameters (job arrival rate $arrival = 0.005$)</i>		
Component Definition	Transition Probability	Service Time
I/O Subsystem Context	$p_{io} = 0.05$	$s_{io} = 20$
System Context	$p_{sys} = 0.40$	$s_{sys} = 1.0$
User Subsystem Context	$p_{user} = 0.54$	$s_{user} = 1.0$
Kernel Subsystem Context	$p_{end} = 0.01$	$s_{sih} = 0.5$

Predicted vs. Measured Results

Transient + Steady State Analysis



Summary of Ongoing Work

Ongoing

- Extending the CSPN language
- GUI with SPN Editor CSPL
- Promela-based models SPNs (i.e., CSPL)
- CSPL ERG RG Q-matrix Solved analytically
- Fault-tree analysis (Erlangen)
- Implementation of solution methods (Erlangen)

Exploring the concept of

- Relate stochastic results back (mechanically) original model as a process of refinement in light of prior runs (sensitivity analysis)
- CGI Web-based access to CSPN (and other components)

The end... time to shut down!

Questions?

