Test and Verification

Brian Nielsen

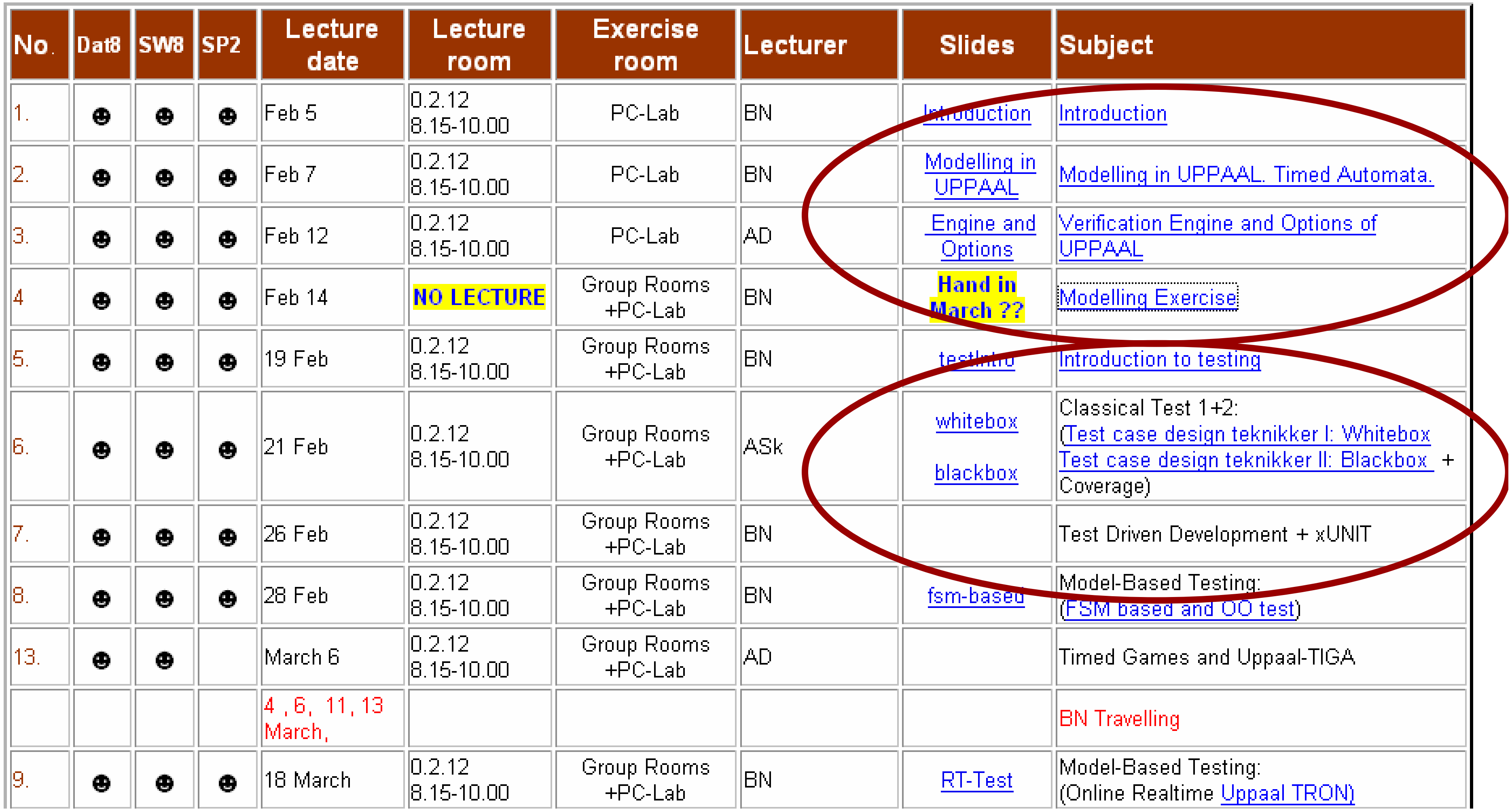
bnielsen@cs.aau.dk

AALBORG UNIVERSITY DENMARK

BRICS
Basic Research
in Computer Science

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

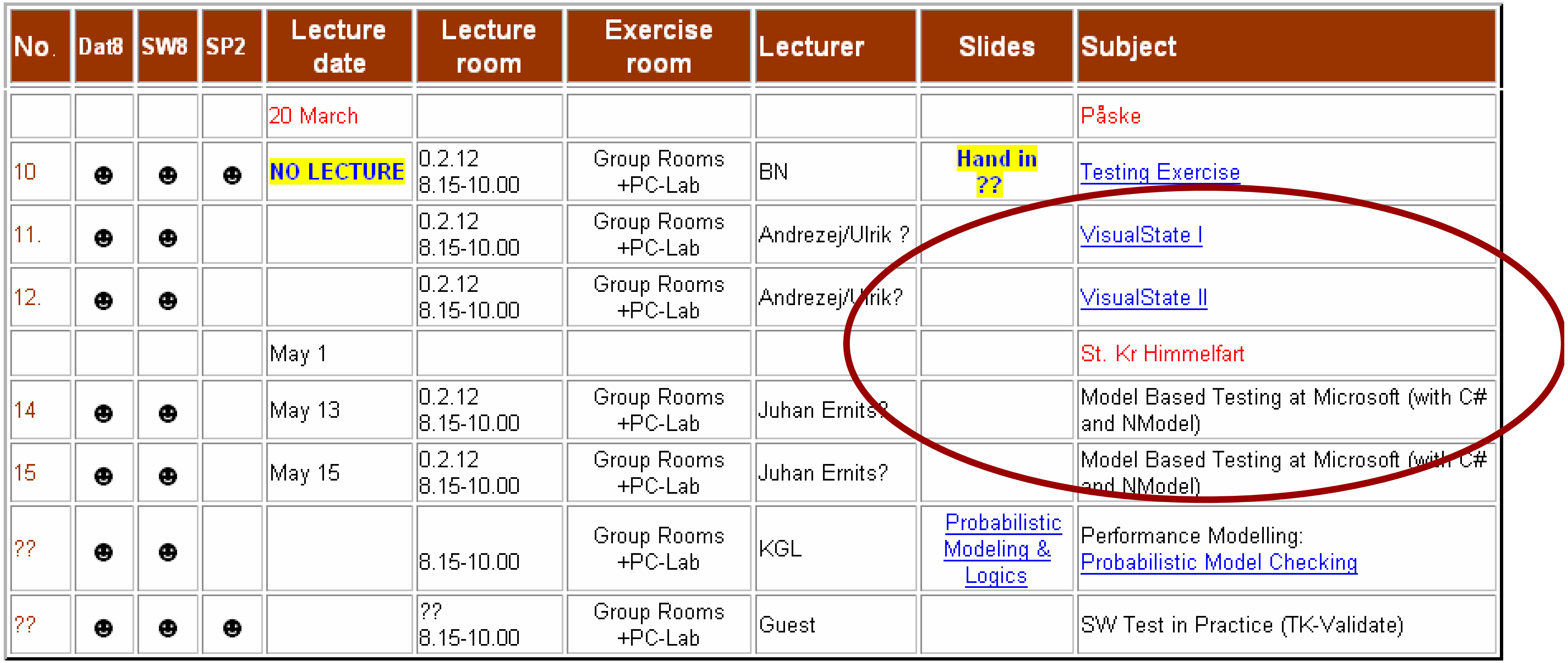
Preliminary Plan



No.	Dat8	SW8	SP2	Lecture date	Lecture room	Exercise room	Lecturer	Slides	Subject
1.	☻	☻	☻	Feb 5	0.2.12 8.15-10.00	PC-Lab	BN	Introduction	Introduction
2.	☻	☻	☻	Feb 7	0.2.12 8.15-10.00	PC-Lab	BN	Modelling in UPPAAL	Modelling in UPPAAL. Timed Automata.
3.	☻	☻	☻	Feb 12	0.2.12 8.15-10.00	PC-Lab	AD	Engine and Options	Verification Engine and Options of UPPAAL
4	☻	☻	☻	Feb 14	**NO LECTURE**	Group Rooms +PC-Lab	BN	**Hand in March ??**	Modelling Exercise
5.	☻	☻	☻	19 Feb	0.2.12 8.15-10.00	Group Rooms +PC-Lab	BN	testIntro	Introduction to testing
6.	☻	☻	☻	21 Feb	0.2.12 8.15-10.00	Group Rooms +PC-Lab	ASk	whitebox blackbox	Classical Test 1+2: (Test case design teknikker I: Whitebox Test case design teknikker II: Blackbox + Coverage)
7.	☻	☻	☻	26 Feb	0.2.12 8.15-10.00	Group Rooms +PC-Lab	BN		Test Driven Development + xUNIT
8.	☻	☻	☻	28 Feb	0.2.12 8.15-10.00	Group Rooms +PC-Lab	BN	fsm-based	Model-Based Testing: (FSM based and OO test)
13.	☻	☻		March 6	0.2.12 8.15-10.00	Group Rooms +PC-Lab	AD		Timed Games and Uppaal-TIGA
				4 , 6, 11, 13 March,					BN Travelling
9.	☻	☻	☻	18 March	0.2.12 8.15-10.00	Group Rooms +PC-Lab	BN	RT-Test	Model-Based Testing: (Online Realtime Uppaal TRON)

CISS

Preliminary Plan



No.	Dat8	SW8	SP2	Lecture date	Lecture room	Exercise room	Lecturer	Slides	Subject
				20 March					Påske
10	☻	☻	☻	**NO LECTURE**	0.2.12 8.15-10.00	Group Rooms +PC-Lab	BN	**Hand in ??**	Testing Exercise
11.	☻	☻			0.2.12 8.15-10.00	Group Rooms +PC-Lab	Andrezej/Ulrik ?		VisualState I
12.	☻	☻			0.2.12 8.15-10.00	Group Rooms +PC-Lab	Andrezej/Ulrik?		VisualState II
				May 1					St. Kr Himmelfart
14	☻	☻		May 13	0.2.12 8.15-10.00	Group Rooms +PC-Lab	Juhan Ernits?		Model Based Testing at Microsoft (with C# and NModel)
15	☻	☻		May 15	0.2.12 8.15-10.00	Group Rooms +PC-Lab	Juhan Ernits?		Model Based Testing at Microsoft (with C# and NModel)
??	☻	☻			8.15-10.00	Group Rooms +PC-Lab	KGL	Probabilistic Modeling & Logics	Performance Modelling: Probabilistic Model Checking
??	☻	☻	☻		?? 8.15-10.00	Group Rooms +PC-Lab	Guest		SW Test in Practice (TK-Validate)

Plan

- Background
  - Research Group and Projects
- Why (and what) test and verification
- Model-based approach
  - Finite State Machines (review)
  - Interacting State Machines
- Verification=Model Checking (1st glance)

Who are we?

AALBORG UNIVERSITY DENMARK

BRICS
Basic Research
in Computer Science

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

Lecturers

Alexandre David

Brian Nielsen

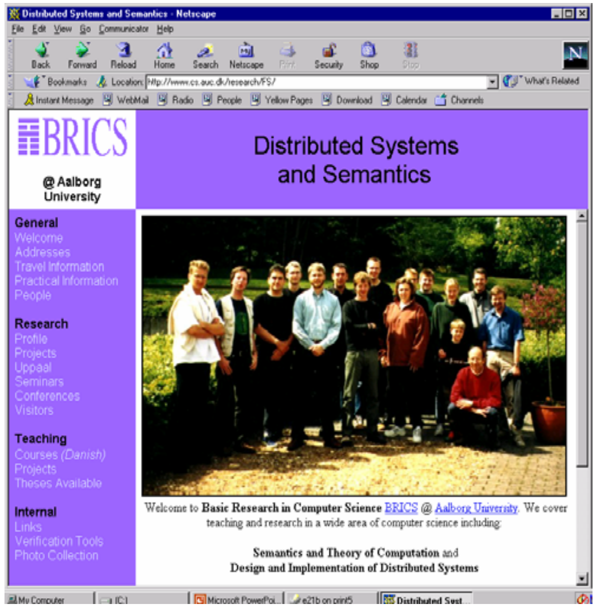
Arne Skou

... and guests

CISS

Research Profile

Distributed Systems & Semantics Unit


Distributed Systems and Semantics - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Location: http://www.cs.auc.dk/research/FS/
What's Related
Instant Message WebMail Radio People Yellow Pages Download Calendar Channels
BRICS
My Computer (C:) Microsoft PowerPoi... e21b on print5 Distributed Syst...

@ Aalborg University

Distributed Systems and Semantics

General
Welcome
Addresses
Travel Information
Practical Information
People

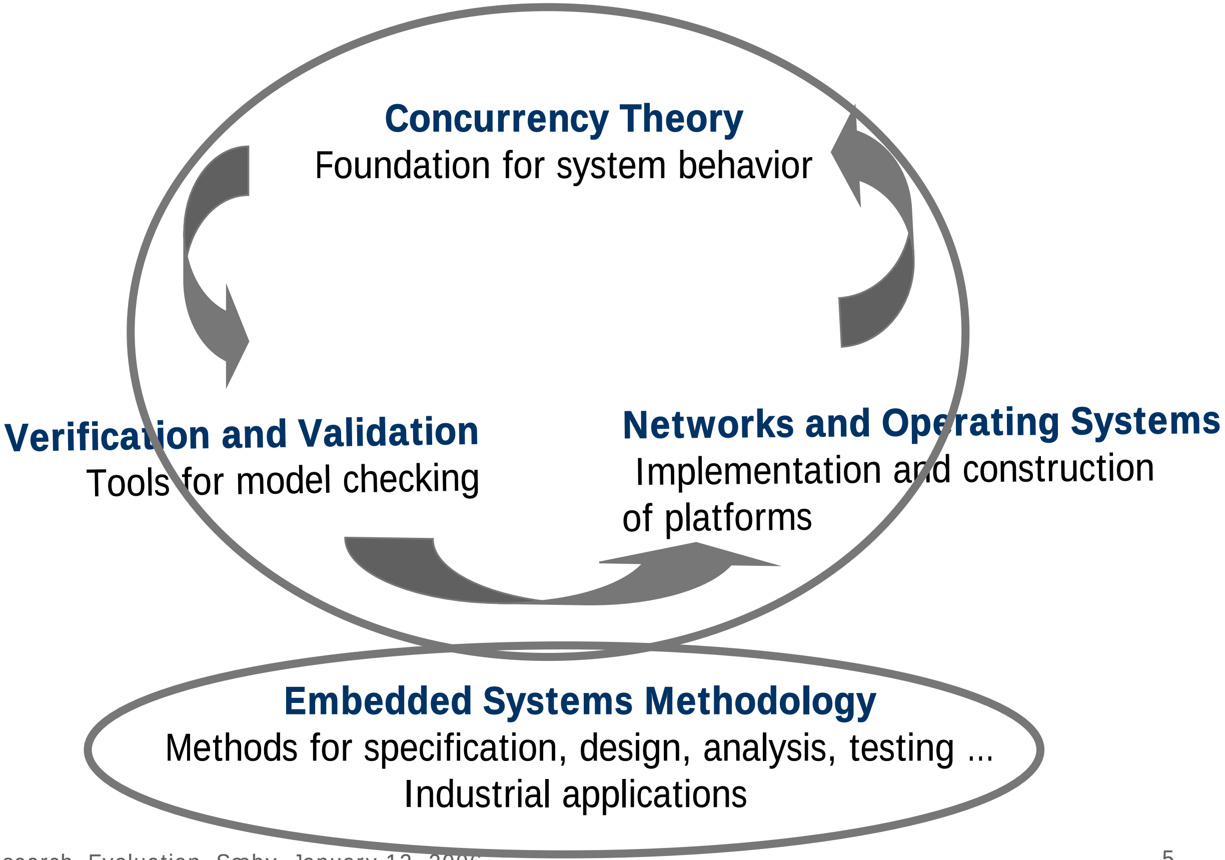
Research
Profile
Projects
Uppaal
Seminars
Conferences
Visitors

Teaching
Courses *(Danish)*
Projects
Theses Available

Internal
Links
Verification Tools
Photo Collection

Welcome to **Basic Research in Computer Science** BRICS @ Aalborg University. We cover teaching and research in a wide area of computer science including:

Semantics and Theory of Computation and **Design and Implementation of Distributed Systems**



Concurrency Theory
Foundation for system behavior

Verification and Validation
Tools for model checking

Networks and Operating Systems
Implementation and construction of platforms

Embedded Systems Methodology
Methods for specification, design, analysis, testing …
Industrial applications

CISS

Center for Indlejrede Software Systemer

Why CISS ?

- 80% of all software is embedded
- Demands for increased functionality with minimal resources
- Requires multitude of skills
  - Software construction
  - Hardware platforms
  - Control theory
  - Comm. technology
- **Goal**: Give a qualitative lift to current industrial practice !!!!!

CISS

CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

CISS Structure

MVTU
25.5 MDKK

ster

Nordjyllands Amt
Aalborg Kommune
12 MDKK

ES Oldenborg
ES Holland
ARTIS

AAU
12.75 MDKK

IKT Virksomheder

Virksomheder
12.75 MDKK

Institut for Datalogi

Institut for Elektroniske Systemer

BRICS@Aalborg
Modelling and Validation;
Programming Languages;
Software Engineering

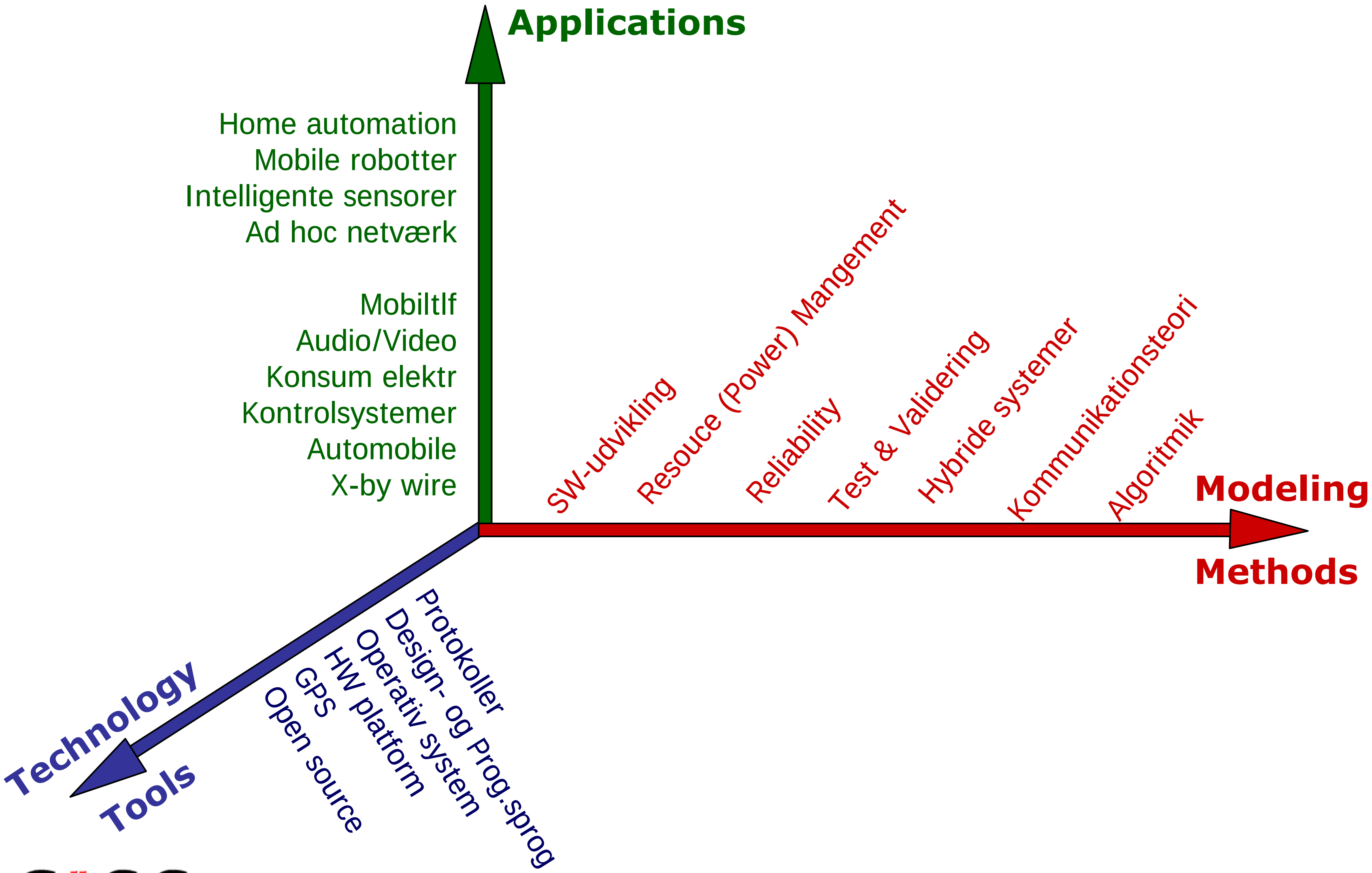
Distributed Real Time Systems
Control Theory;
Real Time Systems;
Networking.

Embedded Systems
Communication;
HW/SW
Power Management

Partners

- Aeromark
- Analog Devices
- Blip Systems
- Danfoss
- Ericsson Telebit
- ETI
- Exhausto
- FOSS
- GateHouse
- Grundfos
- IAR Systems
- MAN B&W
- Novo Nordisk
- Motorola
- Panasonic
- RTX Telecom
- S-Card
- Simrad
- Skov
- SpaceCom
- TK Systemtest
- TDC Totalløsninger
- Aalborg Industries

Focus Areas



Applications

Home automation
Mobile robotter
Intelligente sensorer
Ad hoc netværk

MobiltIf
Audio/Video
Konsum elektr
Kontrolsystemer
Automobile
X-by wire

Modeling

Methods

SW-udvikling
Resouce (Power) Mangement
Reliability
Test & Validering
Hybride systemer
Kommunikationsteori
Algoritmik

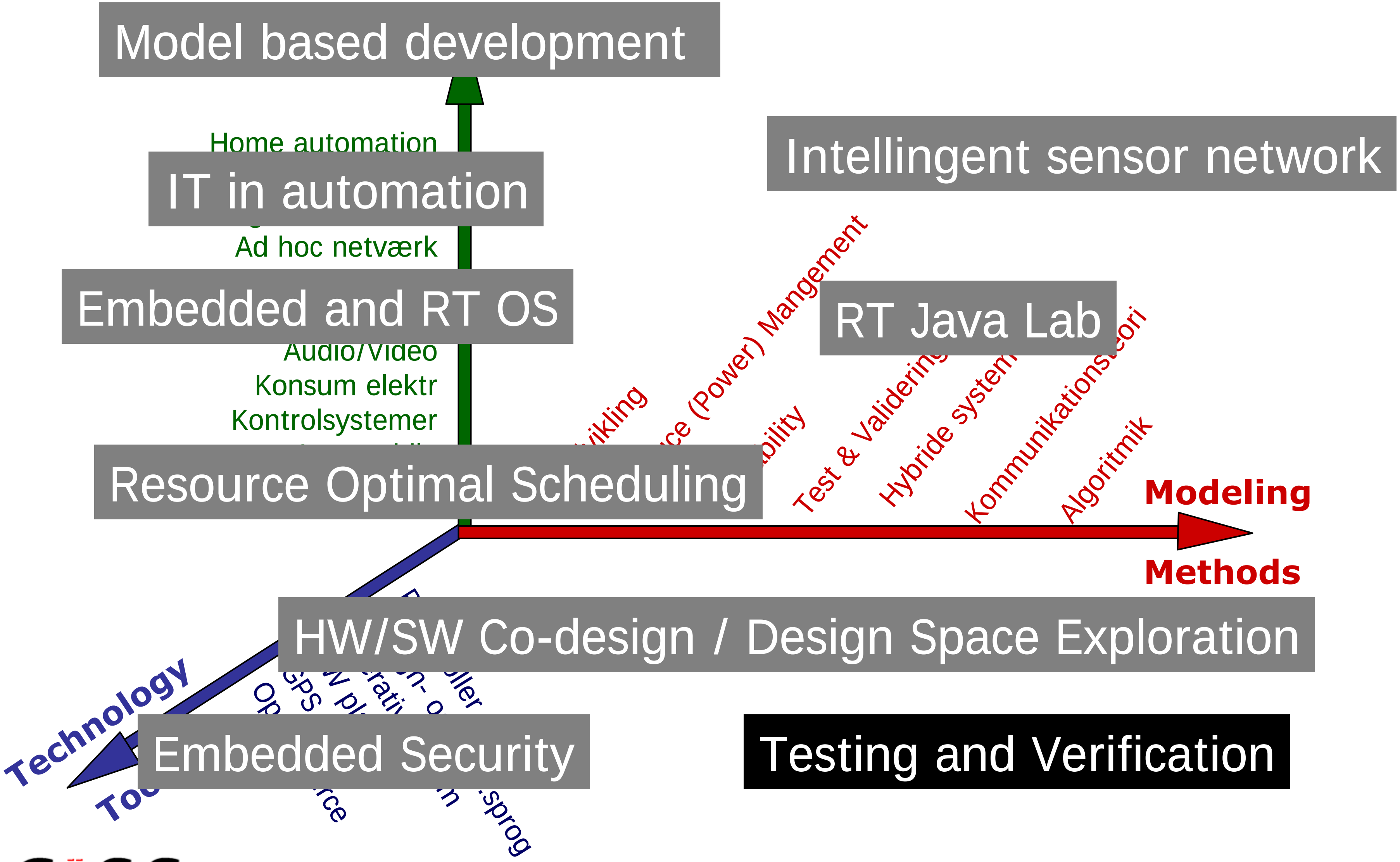
Technology

Tools

Protokoller
Design- og Prog.sprog
Operativ system
HW platform
GPS
Open source

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

Focus Areas



Model based development

IT in automation

Embedded and RT OS

Resource Optimal Scheduling

Intellingent sensor network

RT Java Lab

HW/SW Co-design / Design Space Exploration

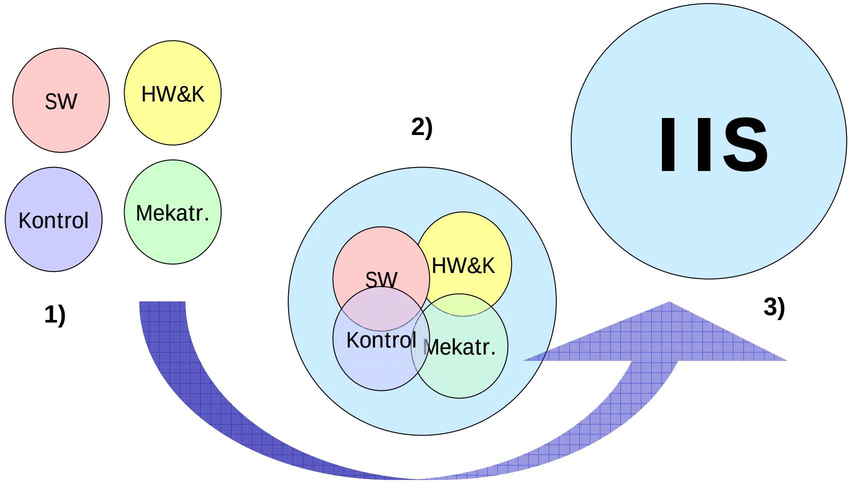
Embedded Security

Testing and Verification

CISS

CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

Local → Regional → National



DaNES

- Danish Network for Intelligent Embedded Systems
- **PARTNERS**

  CISS, IMM, MCI,
  PAJ Systemteknik
  GateHouse A/S
  ICE Power
  Skov A/S
  Terma A/S
  Novo Nordisk A/S
  IO Technologies

- **Funded** by Højteknologifonden

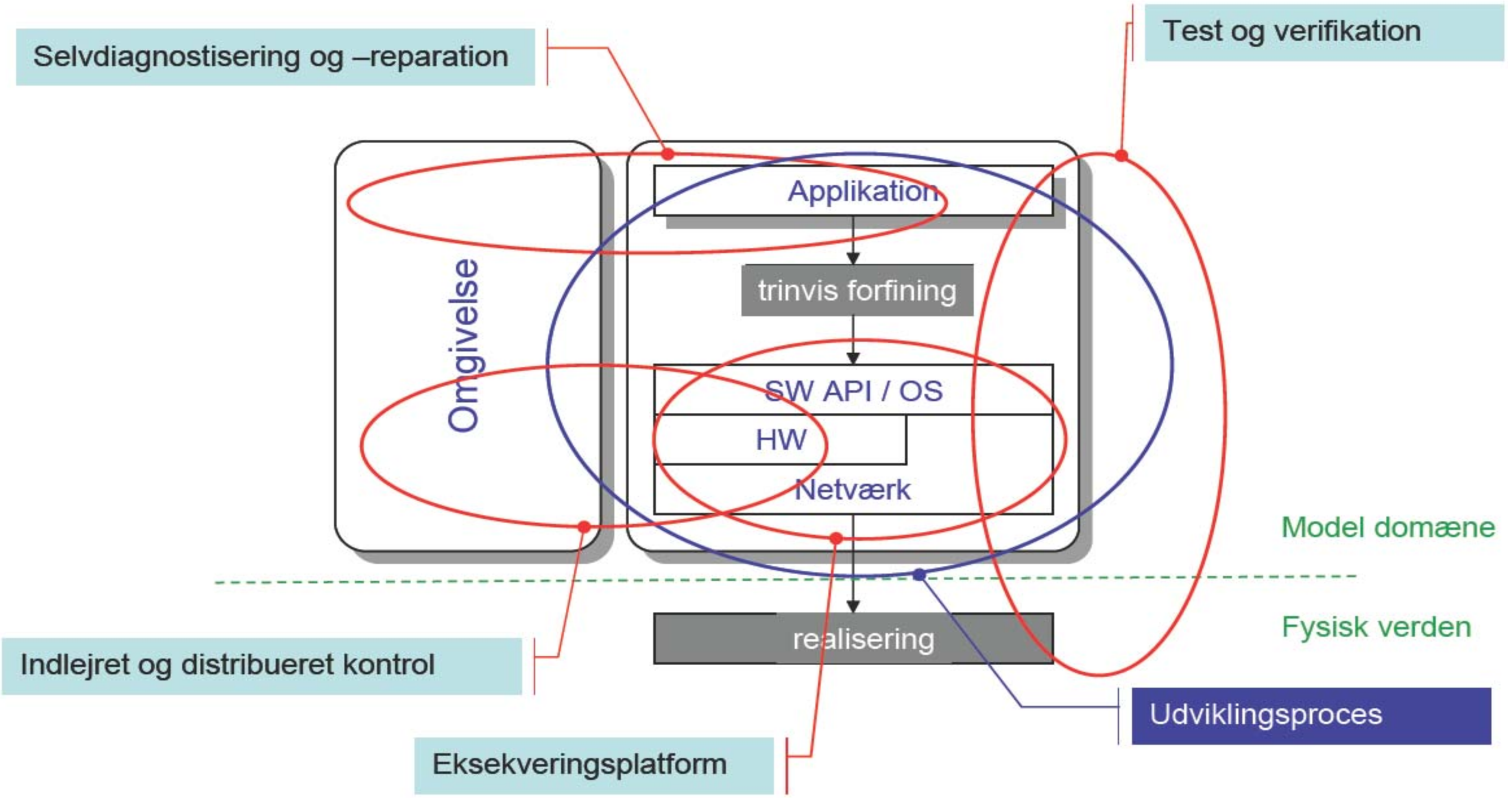
- **Budget**

  63 MDKK / 4 years

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

Local → Regional → National

DaNES

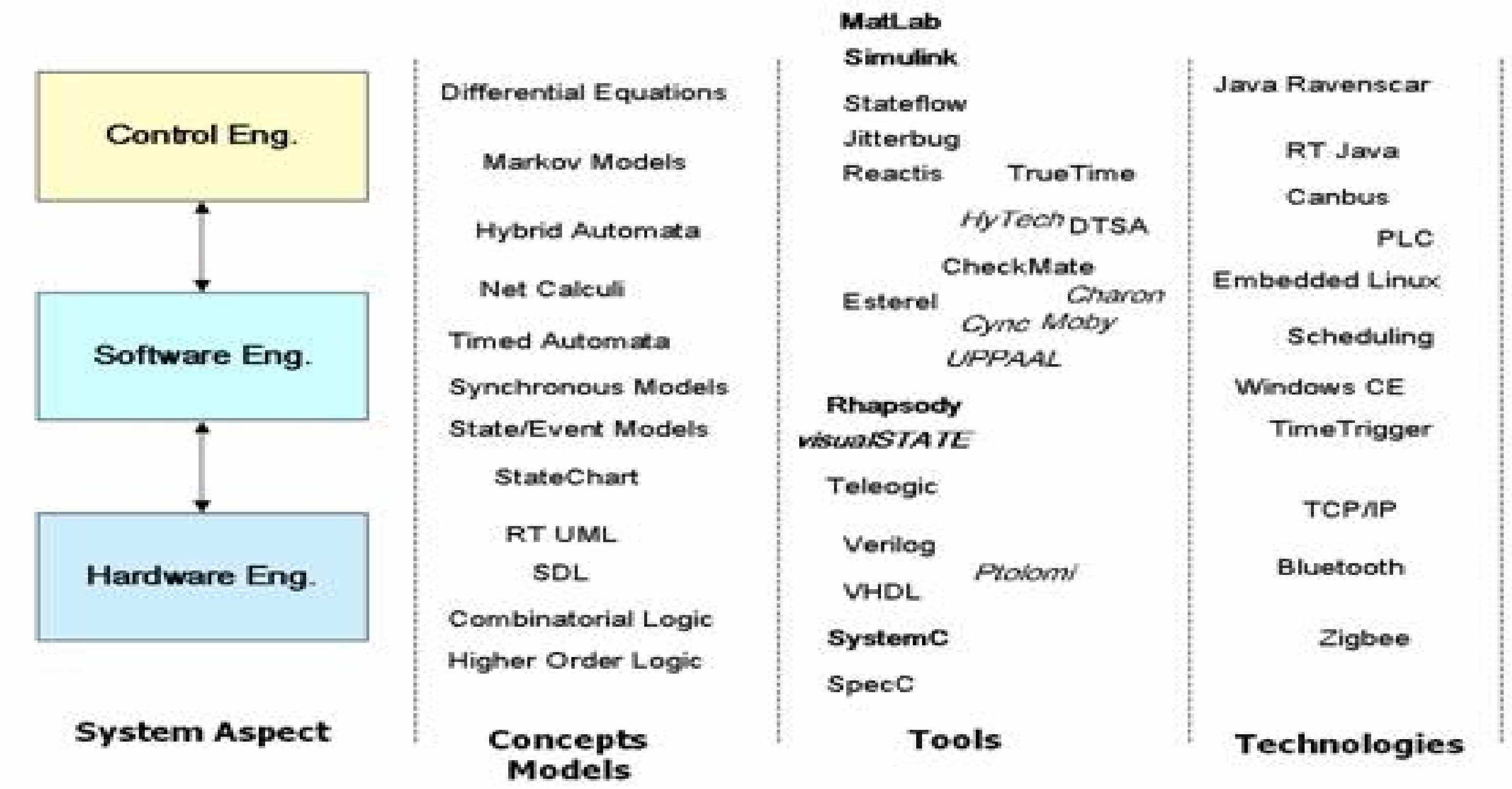


CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

MoDES

Model Driven Development of Intelligent Embedded Systems

| MoDES | Partners | Companies | Research Activities | Researchers | Events | Related Projects | Sponsors |



System Aspect

Control Eng.

Software Eng.

Hardware Eng.

Concepts Models

Differential Equations

Markov Models

Hybrid Automata

Net Calculi

Timed Automata

Synchronous Models

State/Event Models

StateChart

RT UML

SDL

Combinatorial Logic

Higher Order Logic

Tools

MatLab

Simulink

Stateflow

Jitterbug

Reactis

TrueTime

HyTech DTSA

CheckMate

Esterel

Charon

Cync Moby

UPPAAL

Rhapsody

visualSTATE

Teleogic

Verilog

Ptolomi

VHDL

SystemC

SpecC

Technologies

Java Ravenscar

RT Java

Canbus

PLC

Embedded Linux

Scheduling

Windows CE

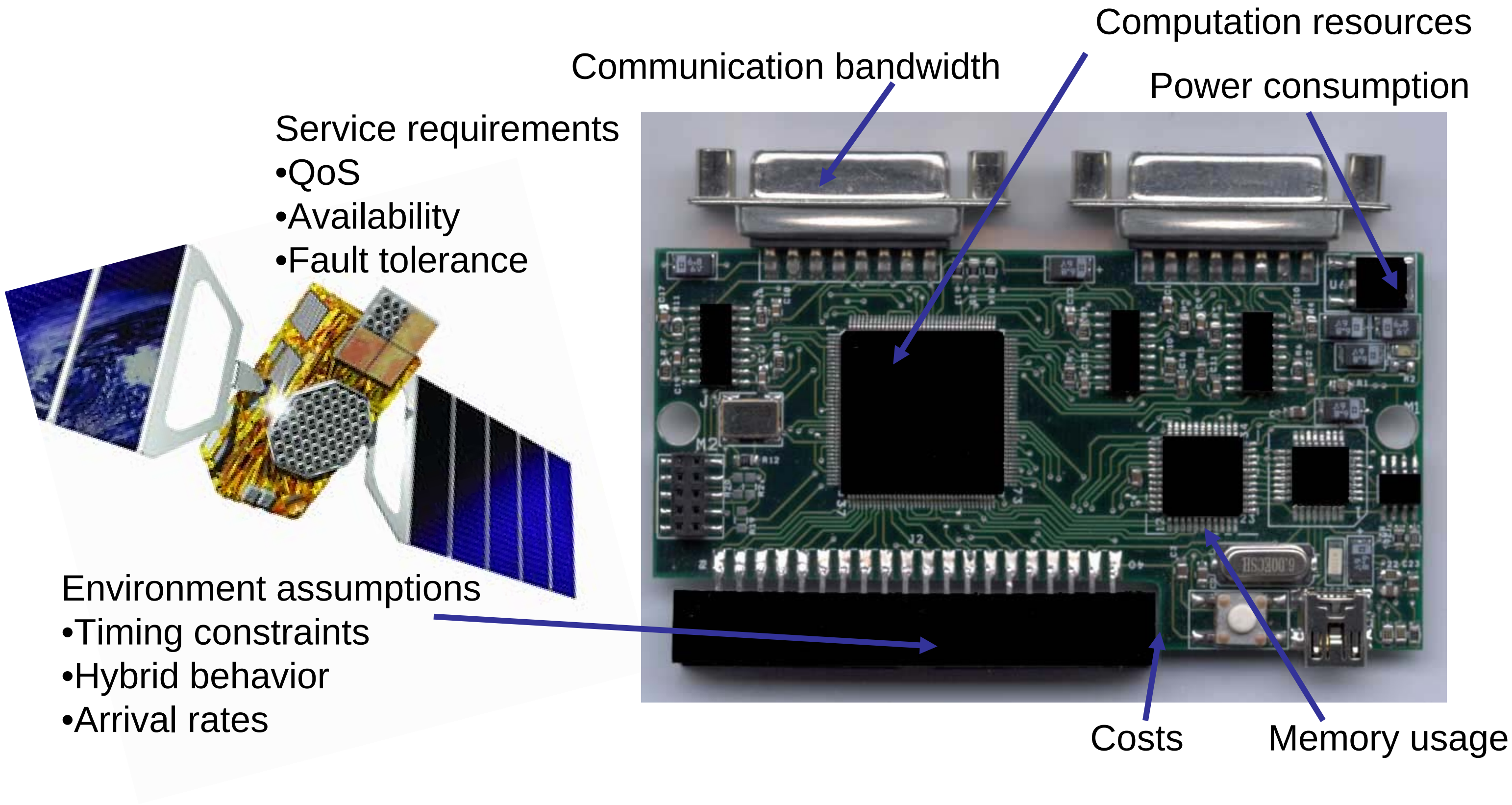
TimeTrigger

TCP/IP

Bluetooth

Zigbee

Quantitative System Properties in Model-Driven-Design of Embedded Systems



Computation resources

Communication bandwidth

Power consumption

Service requirements
- QoS
- Availability
- Fault tolerance

Environment assumptions
- Timing constraints
- Hybrid behavior
- Arrival rates

Costs

Memory usage

Complex Systems

A very complex system

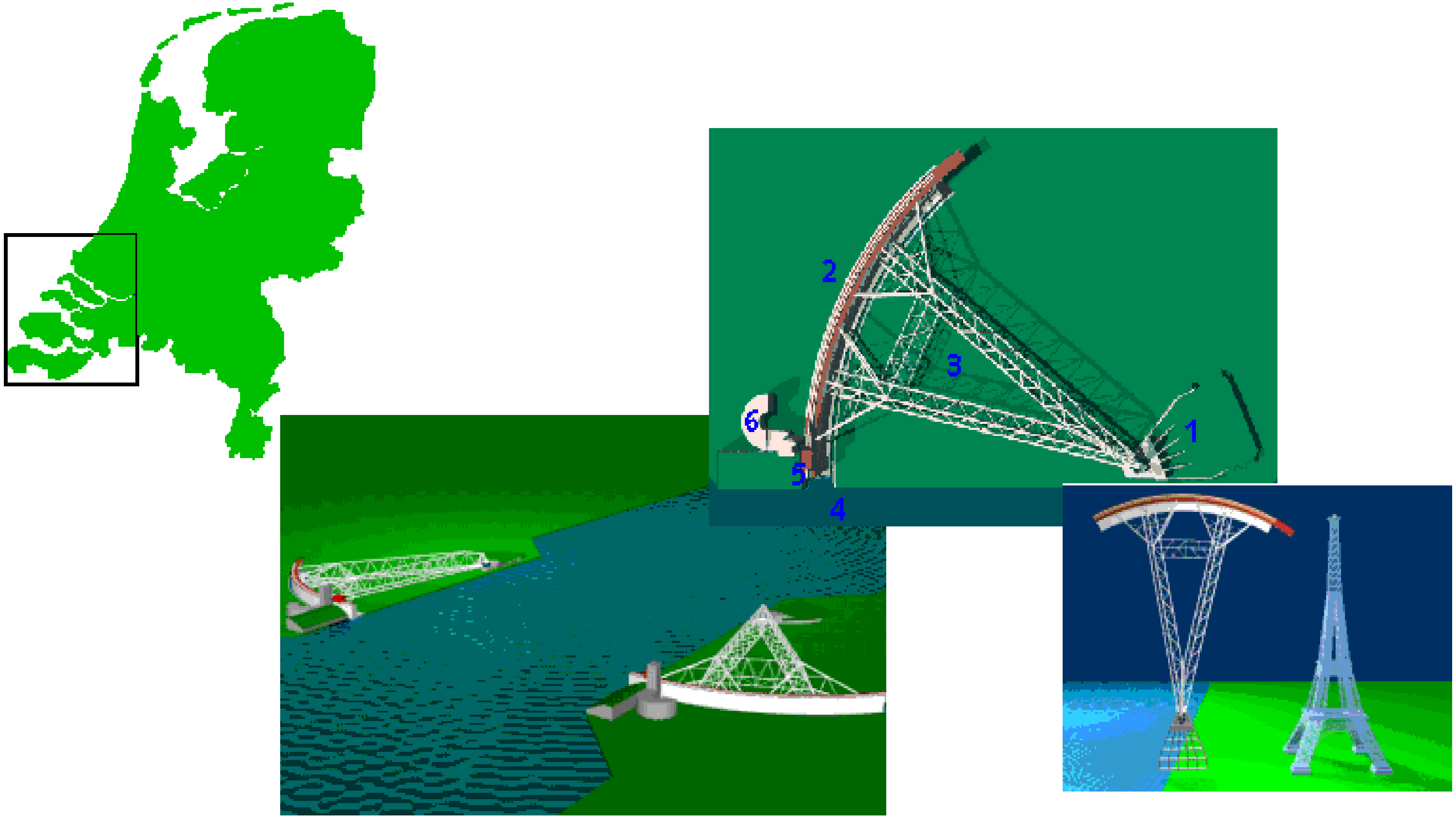
Klaus Havelund, NASA

Spectacular software bugs
Ariane 5

- The first Ariane 5 rocket was launched in June, 1996. It used software developed for the successful Ariane 4. The rocket carried two computers, providing a backup in case one computer failed during launch. Forty seconds into its maiden flight, the rocket veered off course and exploded. The rocket, along with $500 million worth of satellites, was destroyed.

- Ariane 5 was a much more powerful rocket and generated forces that were larger than the computer could handle. Shortly after launch, it received an input value that was too large. The main and backup computers shut down, causing the rocket to veer off course.

Rotterdam Storm Surge Barrier



CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

Spectacular software bugs U.S.S. Yorktown, U.S. Navy

- In 1998, the USS Yorktown became the first ship to test the US Navy's Smart Ship program. The Navy planned to use off-the-shelf computers and software instead of expensive U.S.S. Yorktown, courtesy of U.S. Navy custom-made machines. A sailor mistakenly entered a zero for a data value on a computer. Within minutes, Yorktown was dead in the water. It was several hours before the ship could move again.
- When the sailor entered the mistaken number, the computer tried to divide by zero, which isn't possible. The software didn't check to see if the inputs were valid before computing and generated an invalid answer that was used by another computer. The error cascaded several computers and eventually shut down the ship's engines.

Spectacular software bugs
Moon or Missiles

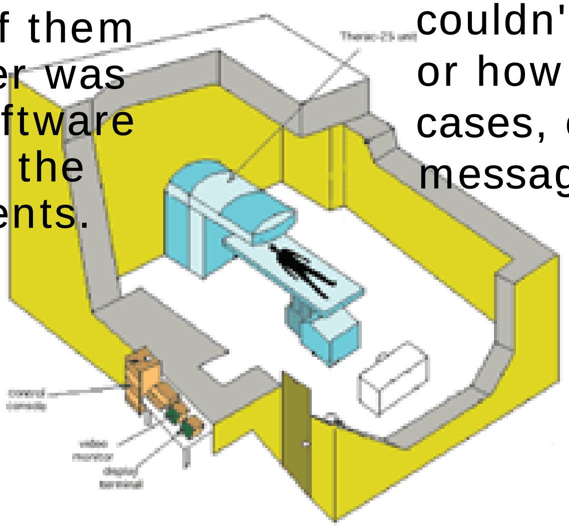
- The United States established the Ballistic Missile Early Warning System (BMEWS) during the Cold War to detect a Soviet missile attack. On October 5, 1960 the BMEWS radar at Thule, Greenland detected something. Its computer control system decided the signal was made by hundreds of missiles

- The radar had actually detected the Moon rising over the horizon. Unfortunately, the BMEWS computer had not been programmed to understand what the moon looked like as it rose in the eastern sky, so it interpreted the huge signal as Soviet missiles. Luckily for all of us, the mistake was realized in time.

Spectacular software bugs
Therac 25

- The Therac-25 radiation therapy machine was a medical device that used beams of electrons or photons to kill cancer cells. Between 1985-1987, at least six people got very sick after Therac-25 treatments. Four of them died. The manufacturer was confident that their software made it impossible for the machine to harm patients.

- The Therac-25 was withdrawn from use after it was determined that it could deliver fatal overdoses under certain conditions. The software would shut down the machine before delivering an overdose, but the error messages it displayed were so unhelpful that operators couldn't tell what the error was, or how serious it was. In some cases, operators ignored the message completely.



"H-tilt"

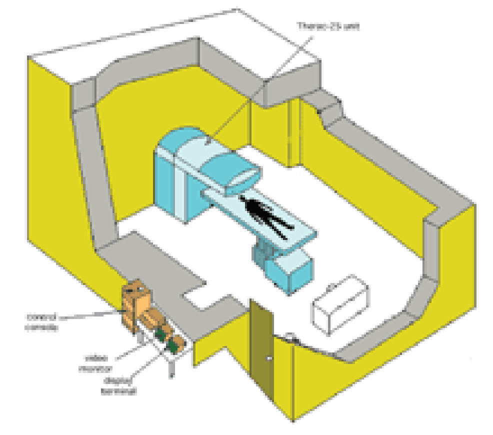
"Malfunction 54"

IEEE Computer, Vol. 26, No. 7, July 1993, pp. 18-41

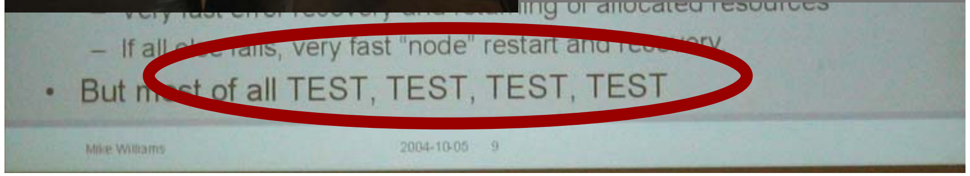
Spectacular Software Bugs .... continued

- INTEL Pentium II floating-point division
  470 Mill US $
- Baggage handling system, Denver
  1.1 Mill US $/day for 9 months
- Mars Pathfinder
- ....

Why T&V?



Michael Williams
Research Director, Ericsson,
SE



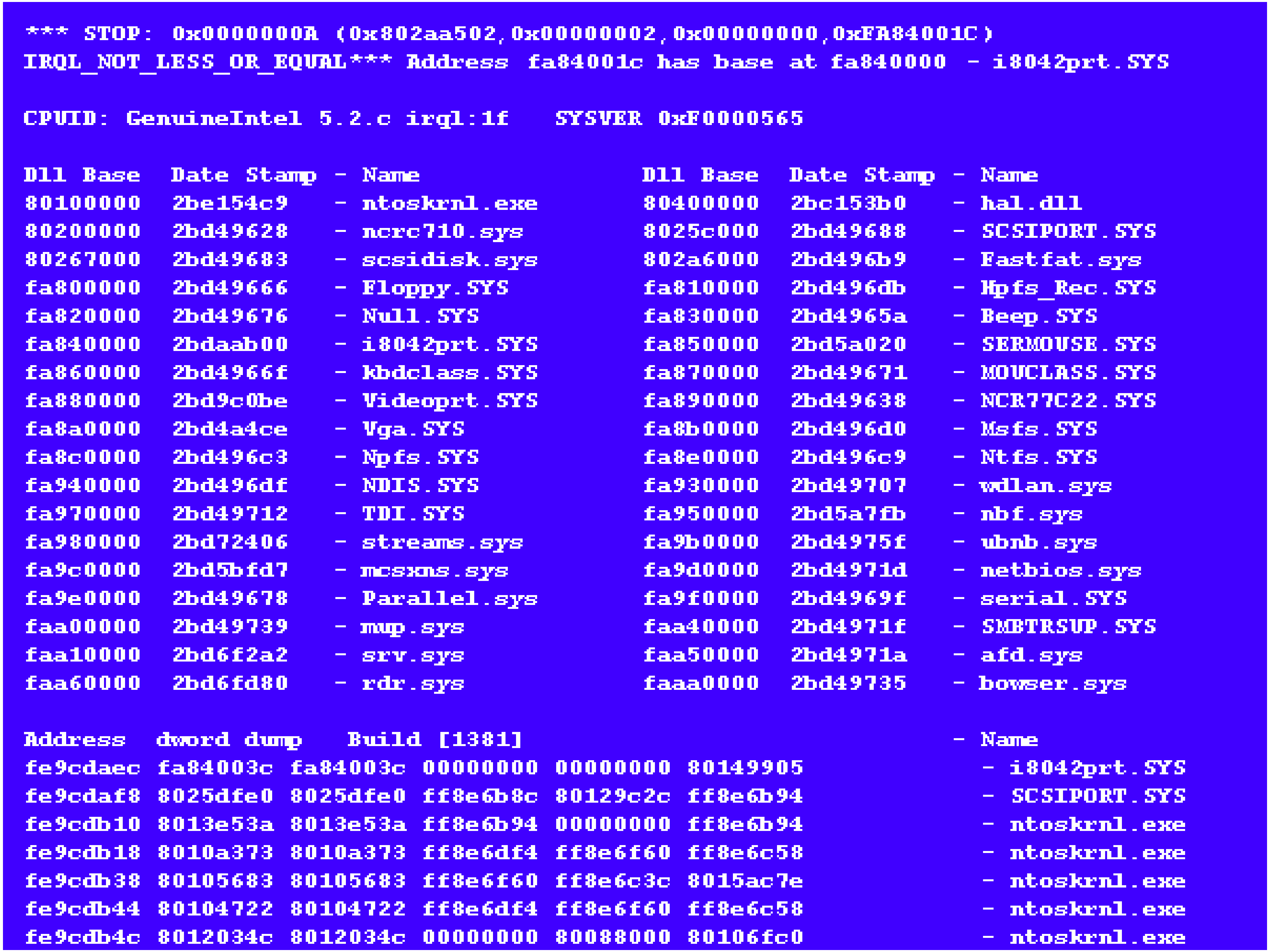
– Very fast error recovery and returning of allocated resources

– If all else fails, very fast "node" restart and recovery

- But most of all TEST, TEST, TEST, TEST

Mike Williams 2004-10-05 9

- Errors in (Embedded) software are extremely expensive



```
*** STOP: 0x0000000A (0x802aa502,0x00000002,0x00000000,0xFA84001C)
IRQL_NOT_LESS_OR_EQUAL*** Address fa84001c has base at fa840000 - i8042prt.SYS

CPUID: GenuineIntel 5.2.c irql:1f   SYSVER 0xF0000565

Dll Base  Date Stamp - Name               Dll Base  Date Stamp - Name
80100000  2be154c9   - ntoskrnl.exe       80400000  2bc153b0   - hal.dll
80200000  2bd49628   - ncrc710.sys        8025c000  2bd49688   - SCSIPORT.SYS
80267000  2bd49683   - scsidisk.sys       802a6000  2bd496b9   - Fastfat.sys
fa800000  2bd49666   - Floppy.SYS         fa810000  2bd496db   - Hpfs_Rec.SYS
fa820000  2bd49676   - Null.SYS           fa830000  2bd4965a   - Beep.SYS
fa840000  2bdaab00   - i8042prt.SYS       fa850000  2bd5a020   - SERMOUSE.SYS
fa860000  2bd4966f   - kbdclass.SYS       fa870000  2bd49671   - MOUCLASS.SYS
fa880000  2bd9c0be   - Videoprt.SYS       fa890000  2bd49638   - NCR77C22.SYS
fa8a0000  2bd4a4ce   - Vga.SYS            fa8b0000  2bd496d0   - Msfs.SYS
fa8c0000  2bd496c3   - Npfs.SYS           fa8e0000  2bd496c9   - Ntfs.SYS
fa940000  2bd496df   - NDIS.SYS           fa930000  2bd49707   - wdlan.sys
fa970000  2bd49712   - TDI.SYS            fa950000  2bd5a7fb   - nbf.sys
fa980000  2bd72406   - streams.sys        fa9b0000  2bd4975f   - ubnb.sys
fa9c0000  2bd5bfd7   - mcsxns.sys         fa9d0000  2bd4971d   - netbios.sys
fa9e0000  2bd49678   - Parallel.sys       fa9f0000  2bd4969f   - serial.SYS
faa00000  2bd49739   - mup.sys            faa40000  2bd4971f   - SMBTRSUP.SYS
faa10000  2bd6f2a2   - srv.sys            faa50000  2bd4971a   - afd.sys
faa60000  2bd6fd80   - rdr.sys            faaa0000  2bd49735   - bowser.sys

Address  dword dump   Build [1381]                             - Name
fe9cdaec fa84003c fa84003c 00000000 00000000 80149905            - i8042prt.SYS
fe9cdaf8 8025dfe0 8025dfe0 ff8e6b8c 80129c2c ff8e6b94            - SCSIPORT.SYS
fe9cdb10 8013e53a 8013e53a ff8e6b94 00000000 ff8e6b94            - ntoskrnl.exe
fe9cdb18 8010a373 8010a373 ff8e6df4 ff8e6f60 ff8e6c58            - ntoskrnl.exe
fe9cdb38 80105683 80105683 ff8e6f60 ff8e6c3c 8015ac7e            - ntoskrnl.exe
fe9cdb44 80104722 80104722 ff8e6df4 ff8e6f60 ff8e6c58            - ntoskrnl.exe
fe9cdb4c 8012034c 8012034c 00000000 80088000 80106fc0            - ntoskrnl.exe
```

Why T&V?

Michael Williams
Research Director, Ericsson, SE

- Very fast error recovery and restarting of allocated resources
  - If all else fails, very fast "node" restart and recovery
- But most of all TEST, TEST, TEST, TEST

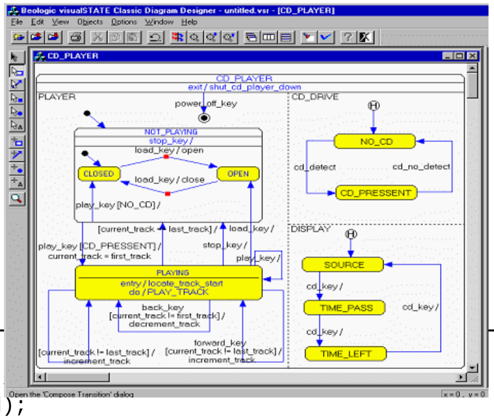
Mike Williams 2004-10-05 9

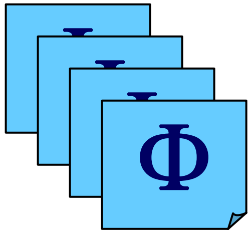
- Errors in (Embedded) software are extremely expensive
- 30-40% of development time spent on (often ad-hoc) testing.
- There is a enormous potential for improved methods and tools.
- “Time-to-market” can be reduced through earli verification and performance analysis

Testing vs. Verification

Verification and Test

Model


Beologic visualSTATE Classic Diagram Designer - untitled.vsr - [CD_PLAYER]
File Edit View Objects Options Window Help
CD_PLAYER
exit / shut_cd_player_down
PLAYER
power_off_key
NOT_PLAYING
stop_key /
load_key / open
CLOSED
OPEN
load_key / close
play_key [NO_CD] /
[current_track = last_track] /
load_key /
play_key [CD_PRESSENT] / current_track = first_track
stop_key /
play_key /
PLAYING
entry / locate_track_start
do / PLAY_TRACK
back_key [current_track != first_track] / decrement_track
forward_key [current_track != last_track] / increment_track
CD_DRIVE
NO_CD
cd_detect
cd_no_detect
CD_PRESSENT
DISPLAY
SOURCE
cd_key /
TIME_PASS
TIME_LEFT
Open the 'Compose Transition' dialog



Spec

```
/* Wait for event
void OS_Wait(void);

/* Operating system visualSTATE process. Mimics a OS process for a
 * visualSTATE system. In this implementation this is the mai
 * interfacing to the visualSTATE basic API. */
void OS_VS_Process(void);

/* Define completion
unsigned char cc;

void HandleError(unsig
{
  printf("Error code %c
  exit(ccArg);
}


/* In d-241 we only use t                    o simulate a
 * system. It purpose is t                   . How this is done is up to
 * you.
 */
void OS_Wait(void)
{
  /*  Ignore the parameters; just retrieve events from the keyboard
and
   *  put them into the queue. When EVENT_UNDEFINED is read from the
   *  keyboard, return to the calling process. */
  SEM_EVENT_TYPE event;
  int num;
```

- Verifikation

  Kode/Model mht Spec

- Test

  System mht Model/Spec

Kode

System

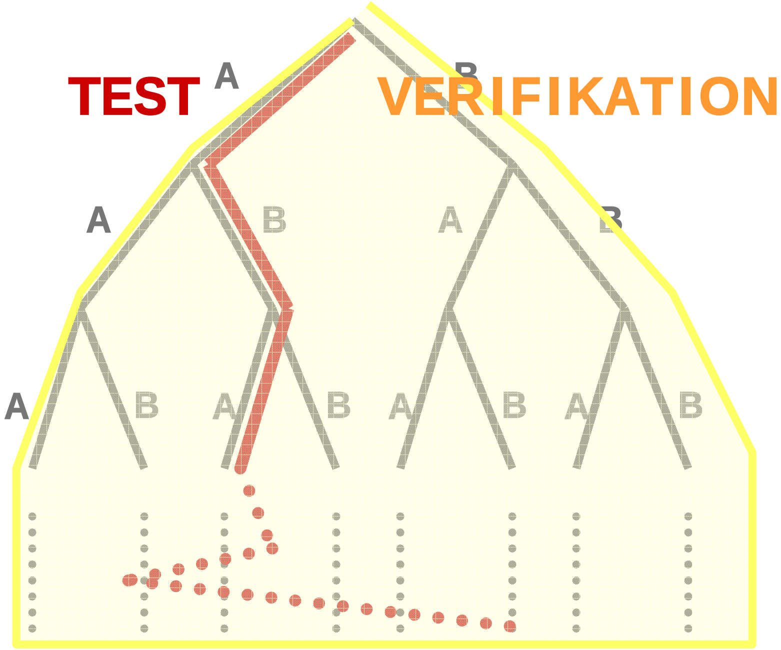
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

Test versus Verification

E F E E G H ... H A

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

T1 T3 T5 T1 ... T4 T3



2^n sequences of length n

Deadlock identified by
VERIFICATION
after sequence of
2000
msgs / < 1min.

UPPAAL

More complex systems

A simple program

```
int x=100;

Process INC
        do
        :: x<200 --> x:=x+1
        od

Process DEC
        do
        :: x>0 --> x:=x-1
        od

Process RESET
        do
        :: x=200 --> x:=0
        od

( INC || DEC || RESET )
```

Which values may x take ?

Questions/Properties:

```
E<>(x>100)
E<>(x>200)
A[](x<=200)
E<>(x<0)
A[](x>=0)
```

Possibly

Always

Another simple program

```
int x=0;

Process P
   do
      x:=x+1
   10 times

( P || P )
```

What are the possible final values of x ?

```
int x=0;

Process P
int r
   do
      r:=x; r++; x:=r
   10 times

( P || P )
```

Atomic stm.

Model-based Approach

AALBORG UNIVERSITY DENMARK

BRICS
Basic Research
in Computer Science

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

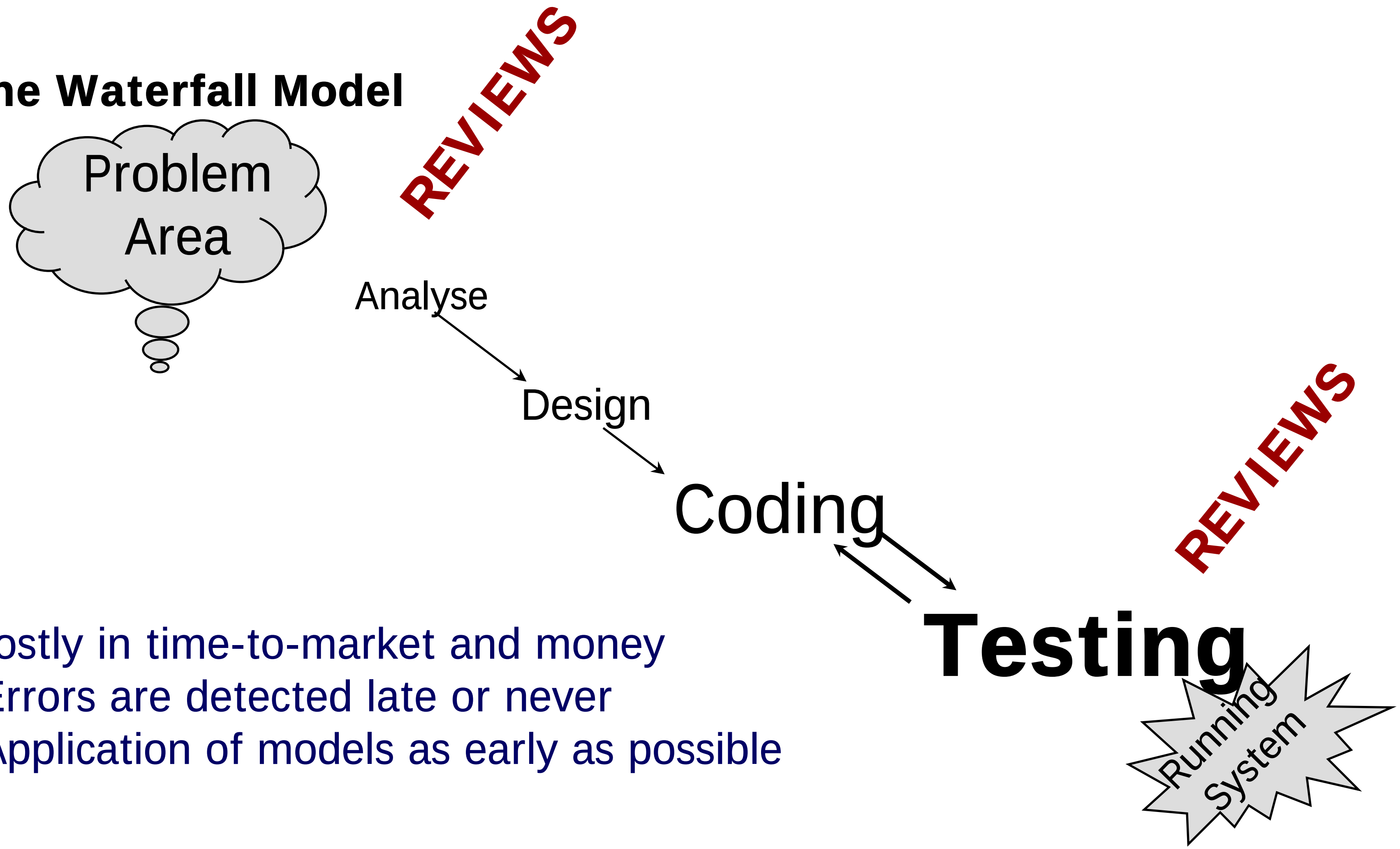
Suggested Solution?

Model based

validation, verfication and testing
of software and hardware

Traditional Software Development

The Waterfall Model



Problem Area

REVIEWS

Analyse

Design

Coding

Testing

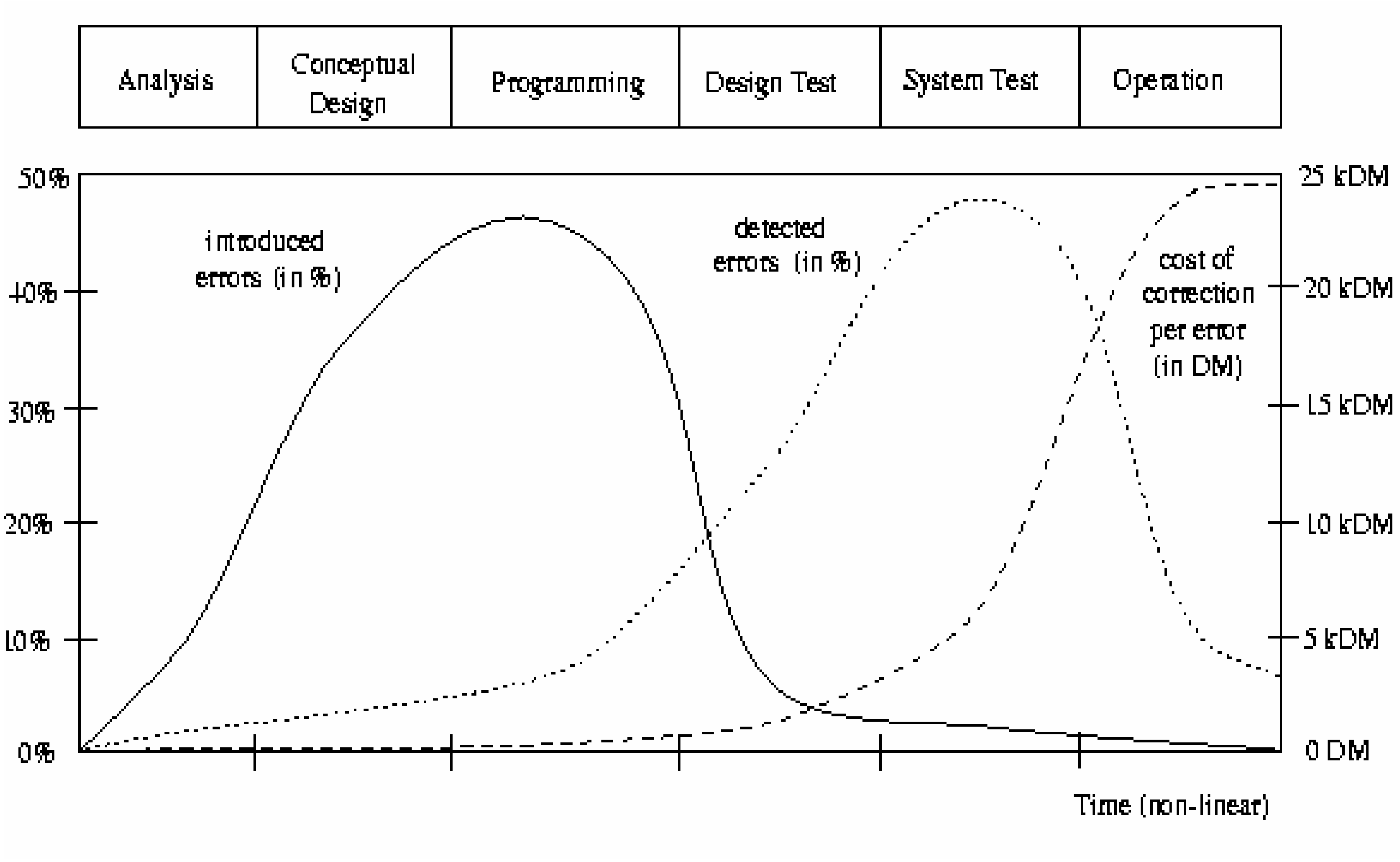
REVIEWS

Running System

- ♦Costly in time-to-market and money
- ♦ Errors are detected late or never
- ♦ Application of models as early as possible

Introducing, Detecting and Repairing Errors

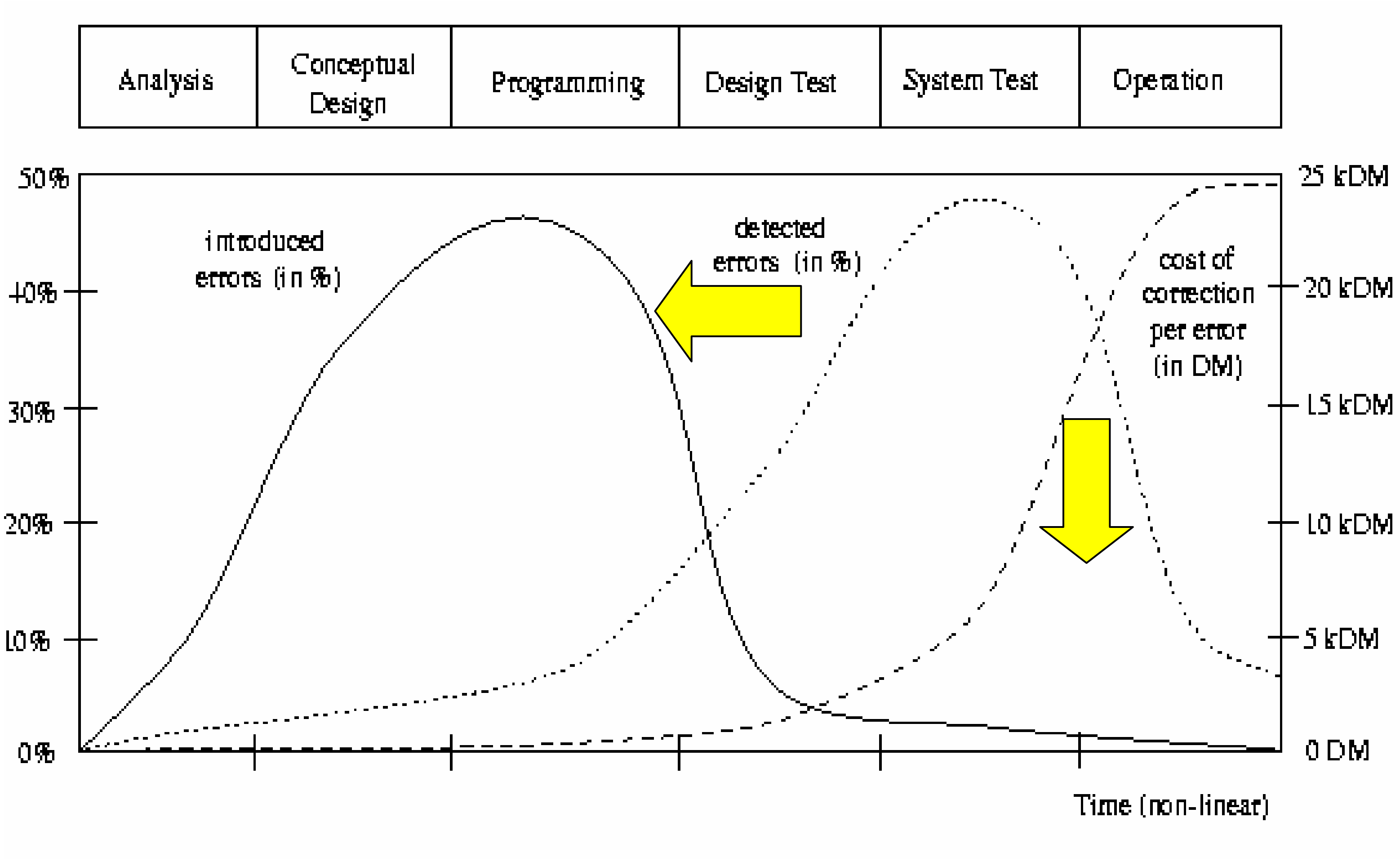
Liggesmeyer 98



CISS

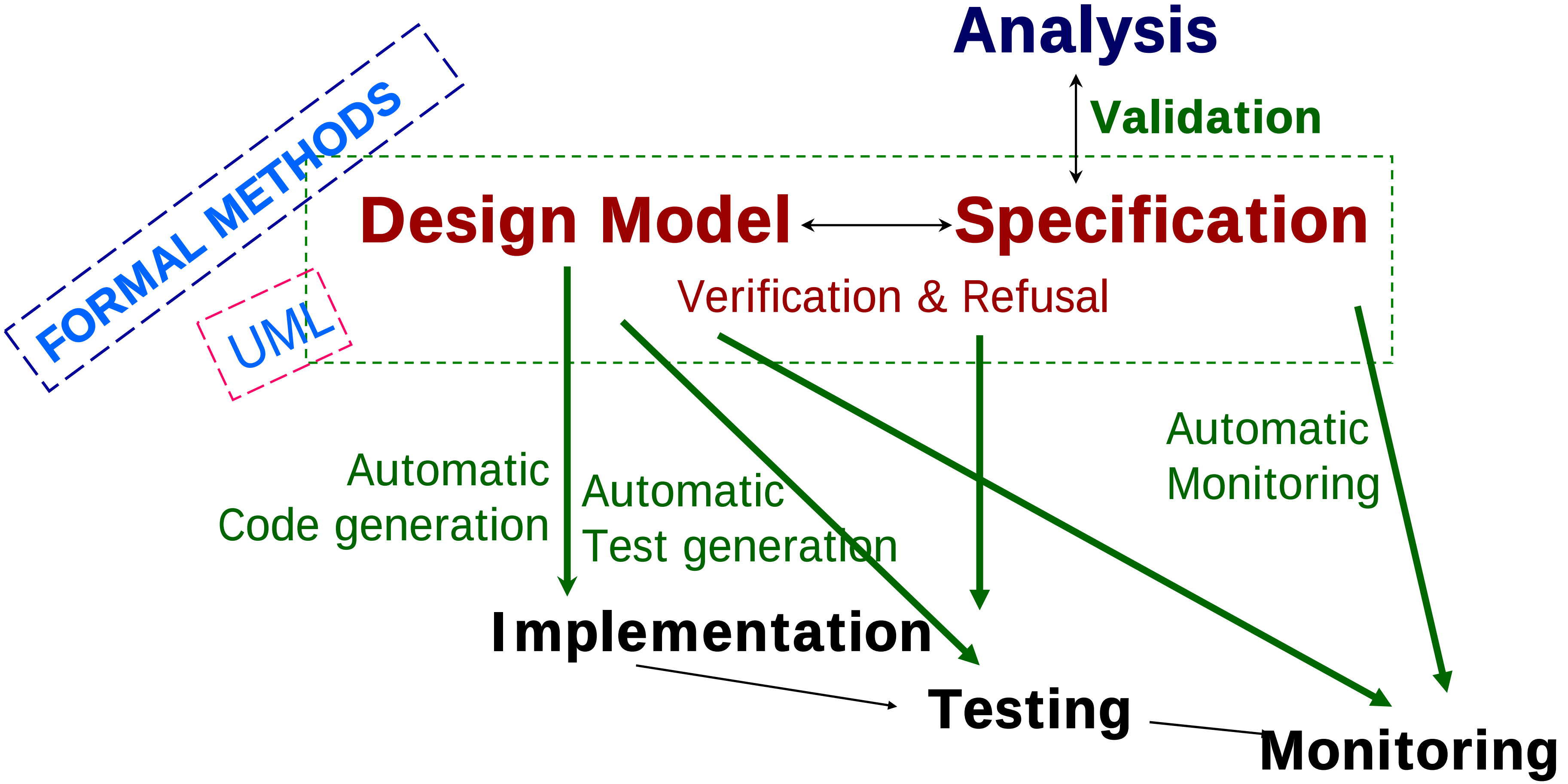
Introducing, Detecting and Repairing Errors

Liggesmeyer 98



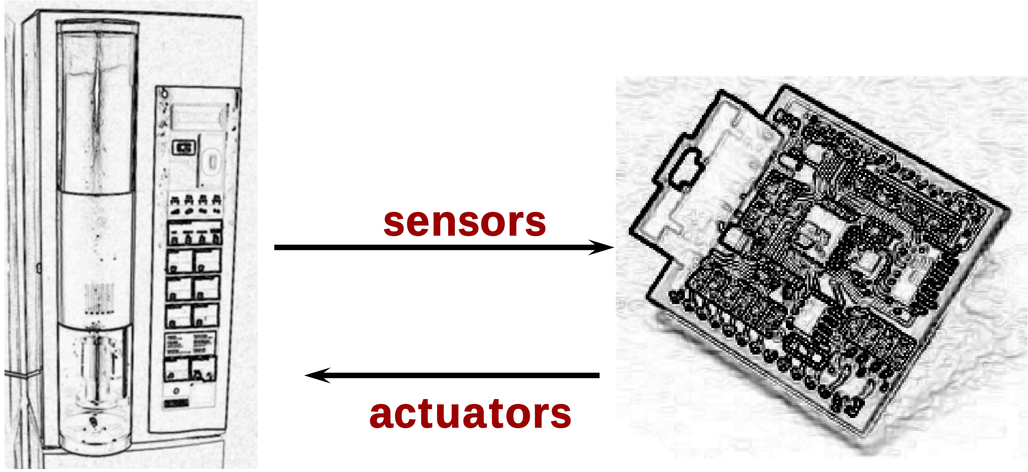
CISS

Model-Driven Development



CISS

Real-time Systems



sensors

actuators

Plant/Env

Continuous

Controller Program

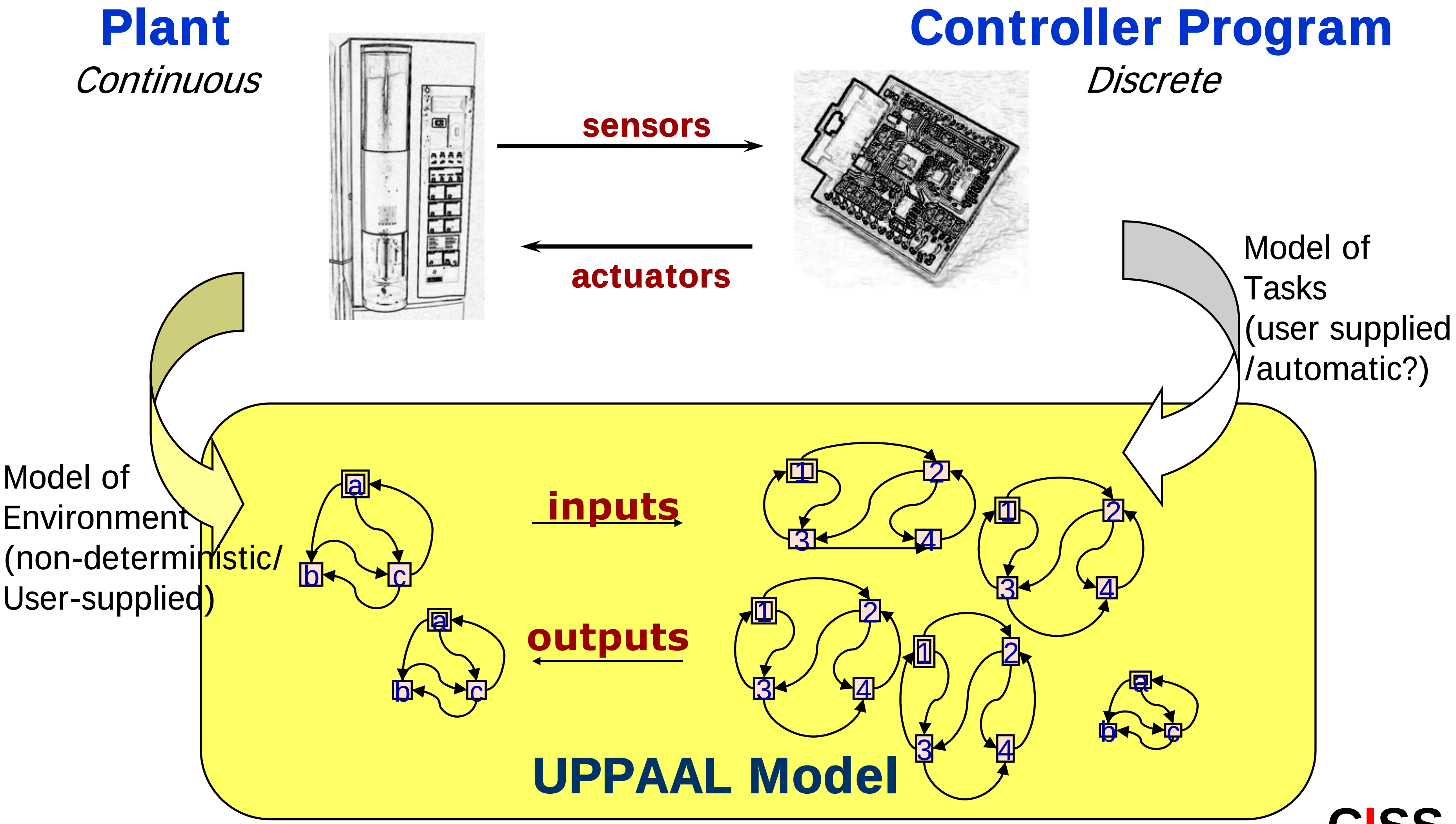
Discrete

Eg.:

- Realtime Protocols
- Pump Control
- Air Bags
- Robots
- Cruise Control
- ABS
- CD Players
- Production Lines

Real Time System
A system where correctness not only depends on the logical order of events but also on their **timing**!!

Real-time Modeling



Plant

Continuous

Controller Program

Discrete

sensors

actuators

Model of Tasks (user supplied /automatic?)

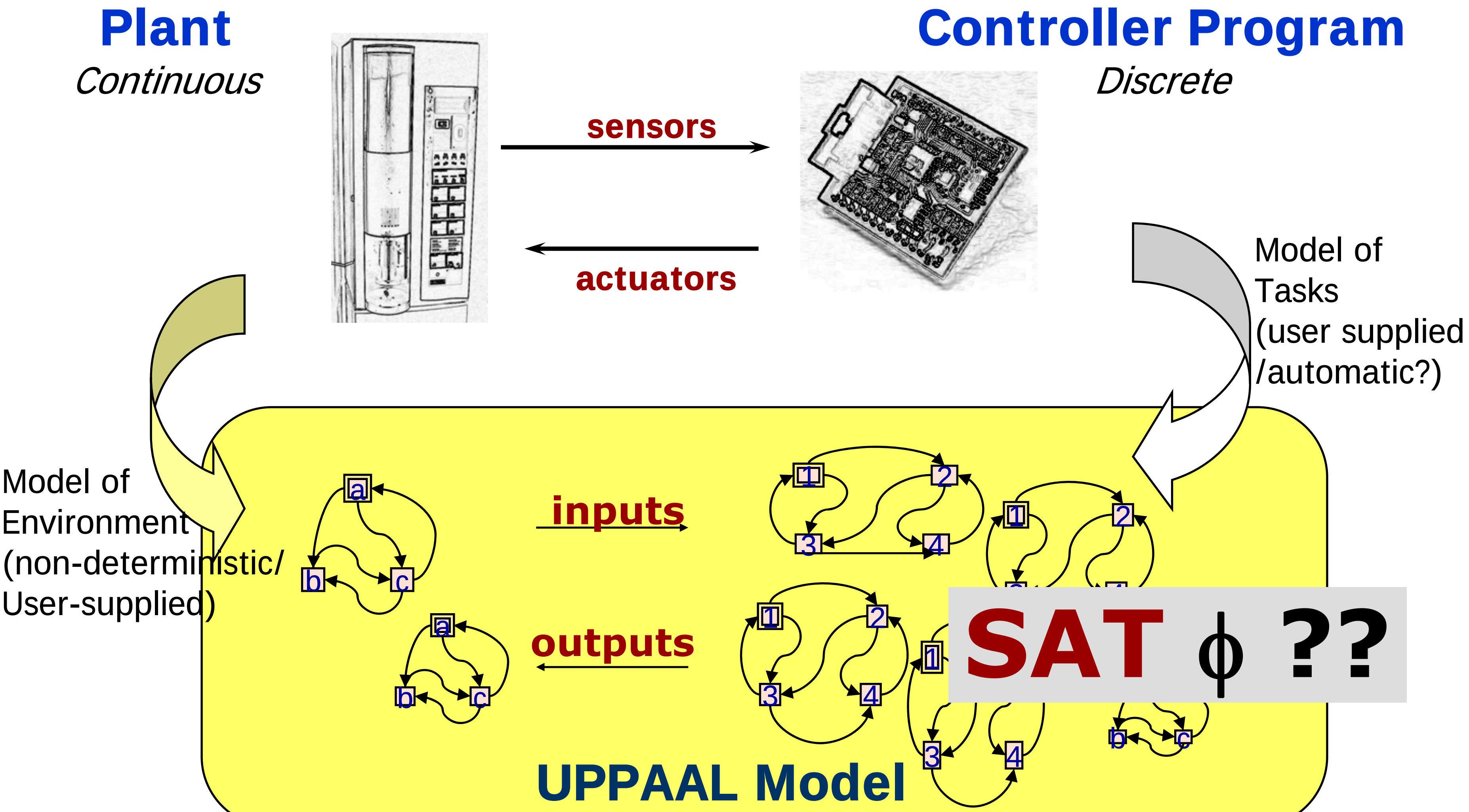
Model of Environment (non-deterministic/ User-supplied)

inputs

outputs

UPPAAL Model

Real-time Model-checking



Plant

Continuous

Controller Program

Discrete

sensors

actuators

Model of Tasks (user supplied /automatic?)

Model of Environment (non-deterministic/ User-supplied)

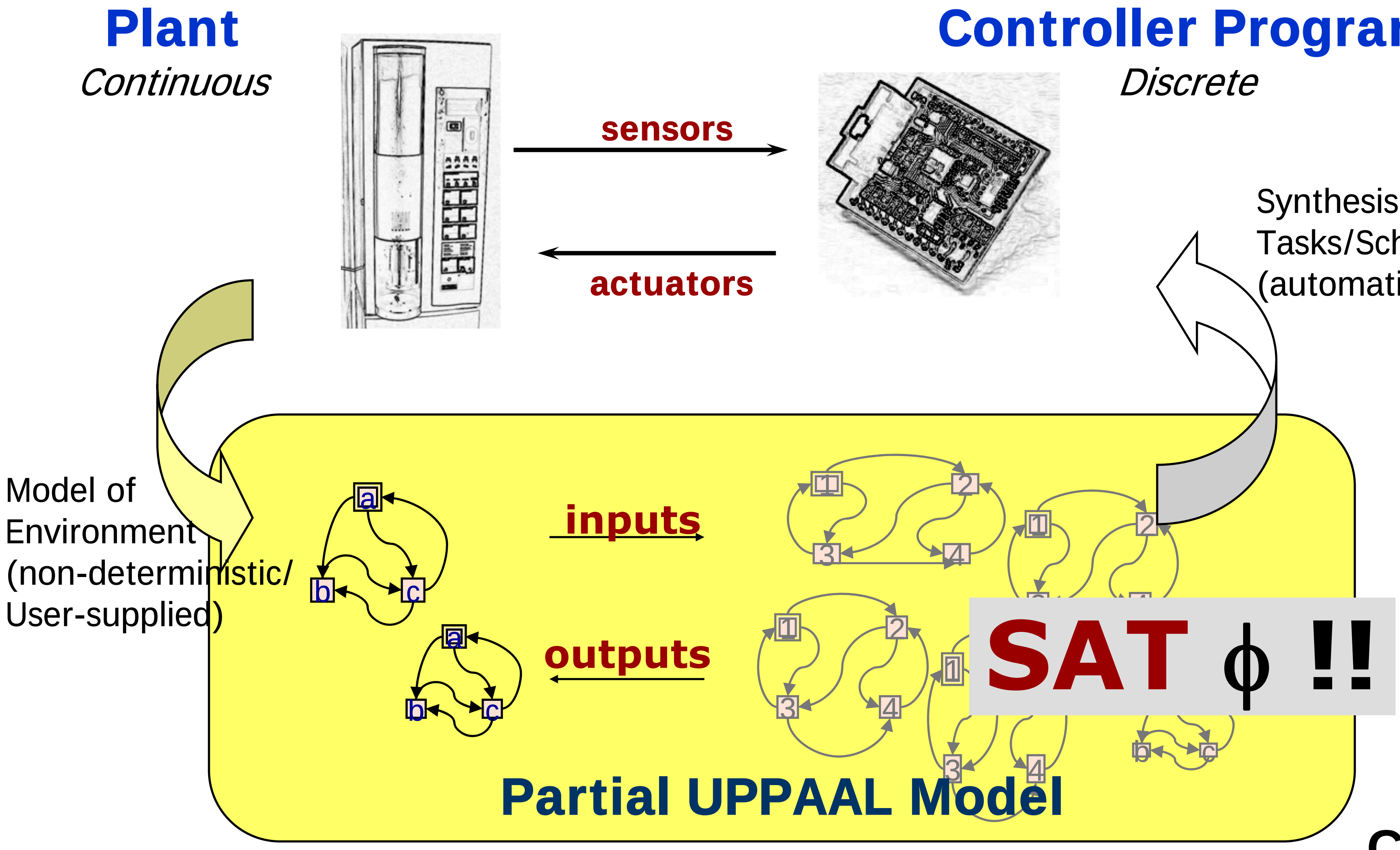
inputs

outputs

SAT ϕ ??

UPPAAL Model

Real-time Controller Synthesis



Plant

Continuous

Controller Program

Discrete

sensors

actuators

Synthesis of Tasks/Scheduler (automatic)

Model of Environment (non-deterministic/ User-supplied)

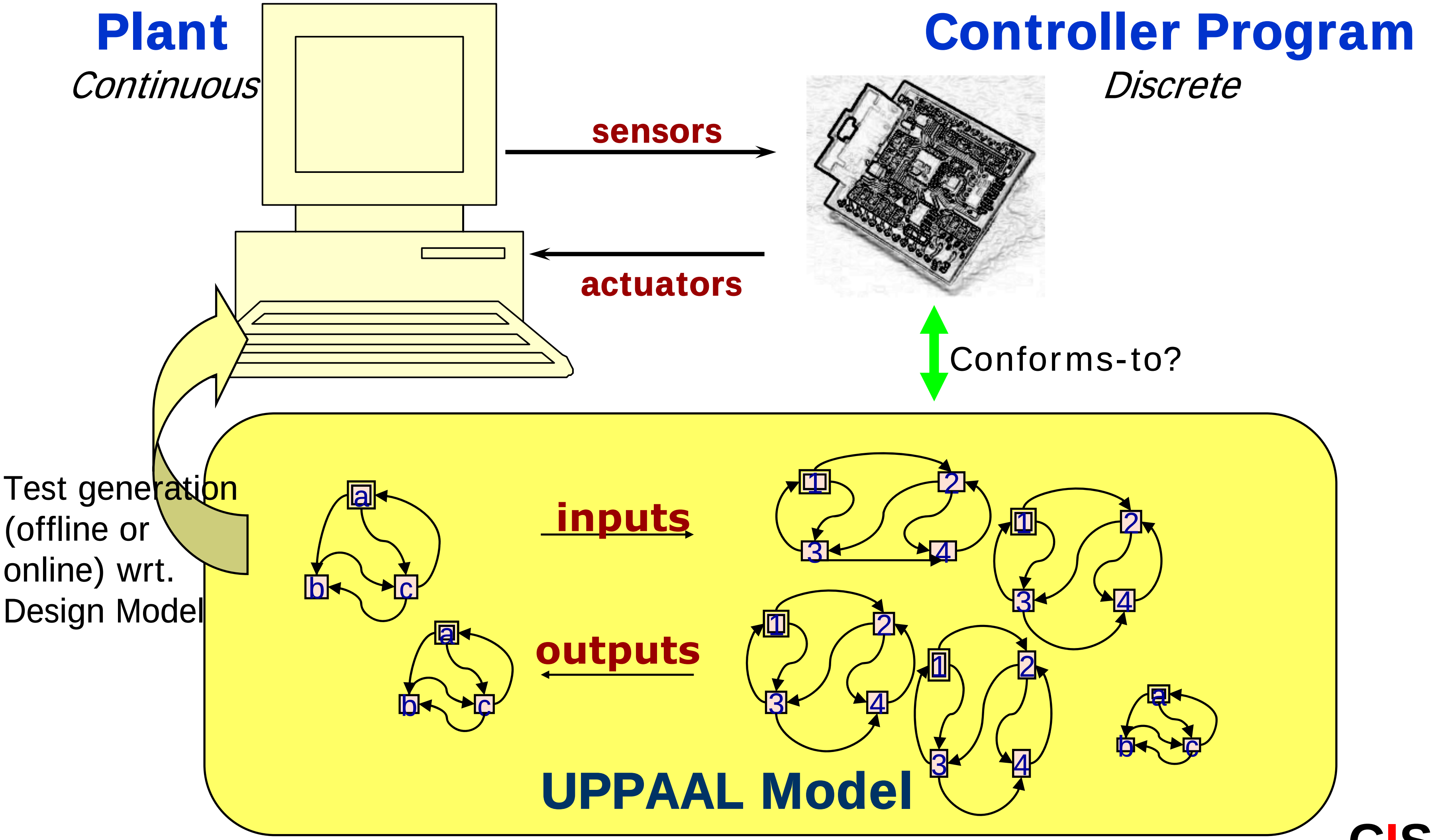
inputs

outputs

SAT ϕ !!

Partial UPPAAL Model

Real-time Model-Based Testing



Plant

Continuous

Controller Program

Discrete

sensors

actuators

Conforms-to?

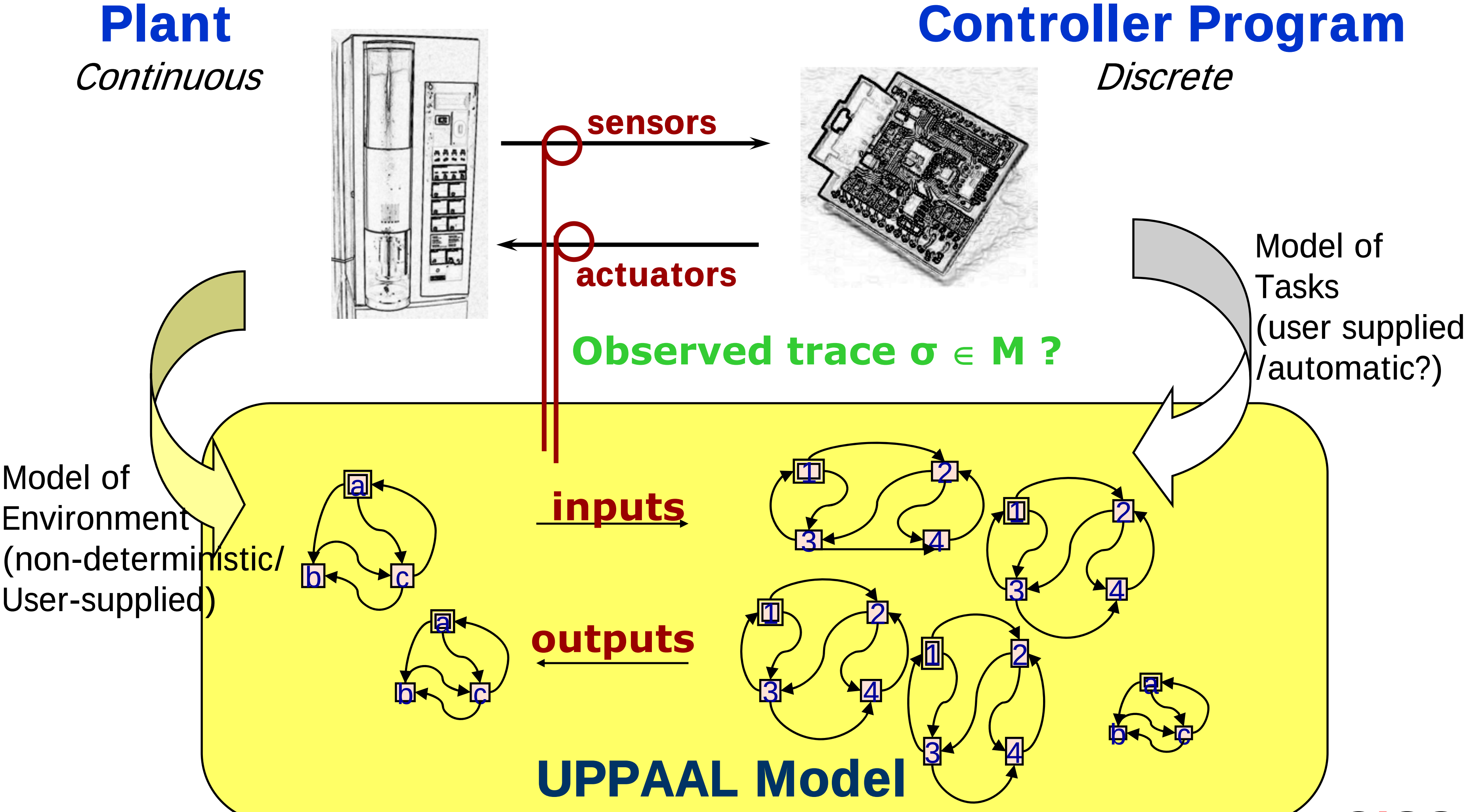
Test generation (offline or online) wrt. Design Model

inputs

outputs

UPPAAL Model

Real-time Monitoring



Plant

Continuous

Controller Program

Discrete

sensors

actuators

Observed trace σ ∈ M ?

Model of Tasks (user supplied /automatic?)

Model of Environment (non-deterministic/ User-supplied)

inputs

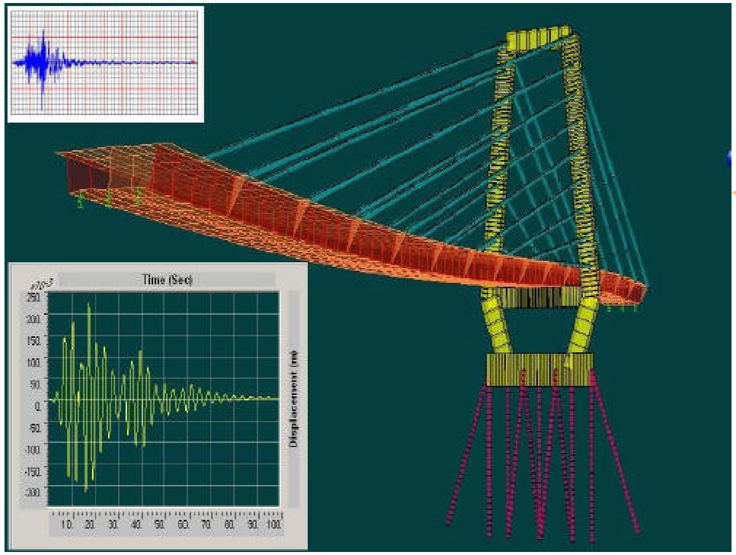
outputs

UPPAAL Model

Models

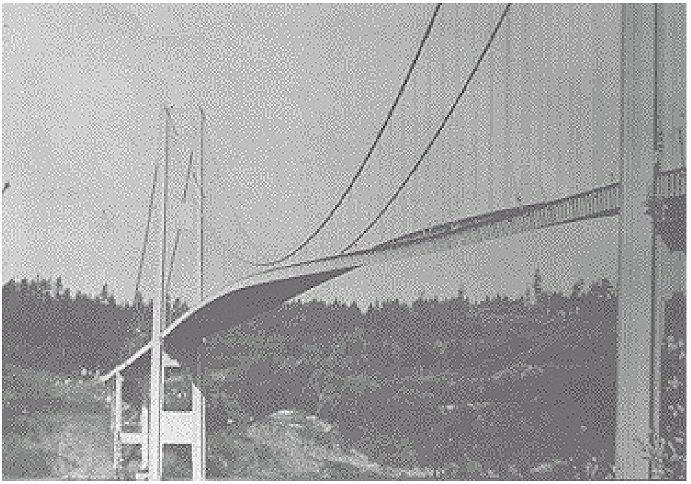
- A model is a simplified representation of the real world.
- Used gain confidence in the adequacy and validity of a proposed system
- Models selected aspects
- Removes irrelevant details

Model



”implements??”

Realization



Models

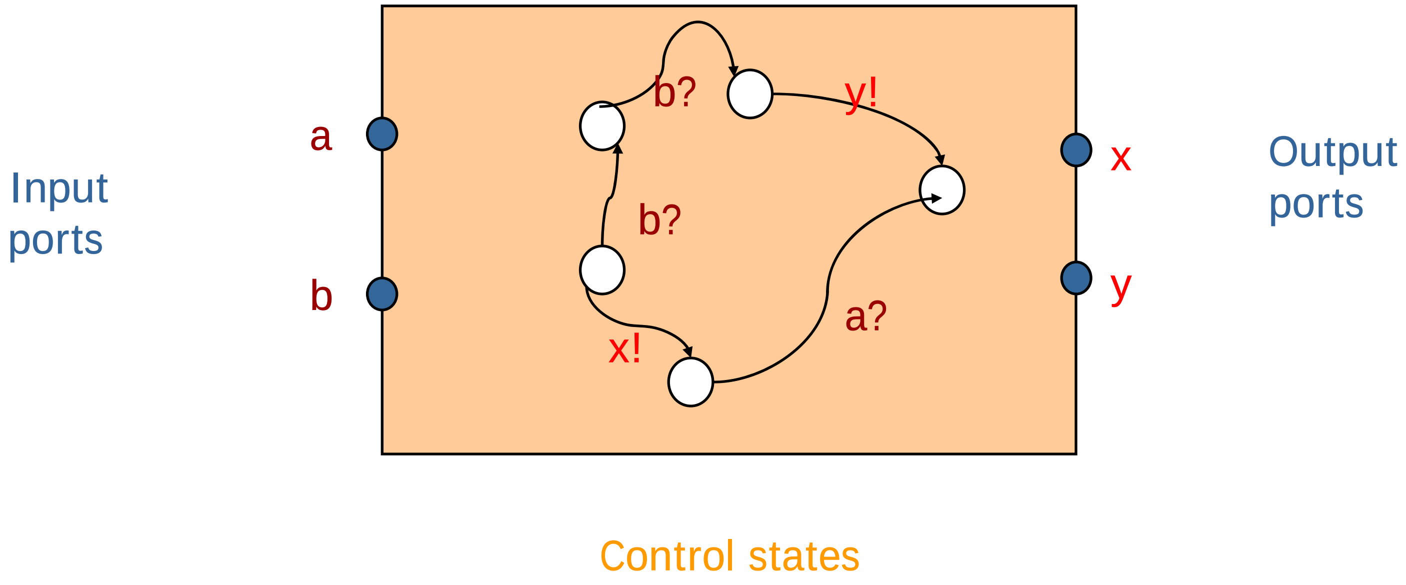
- Abstractions of the problem-space, not solution space
- Domain Specific Modeling Languages
  - Simulink/StateFlow
  - UML,
- Early exploration of design-alternatives
- Automatic transformation
  - Correctness-by-***construction*** vs. Correctness-by-*correction*

Model-based vs. MDD

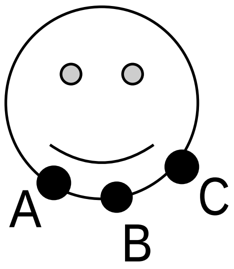
- Model Driven Development:
  - Model is the center of focus from analysis to execution
  - Model is gradually refined / transformed into solution
- Model-based Development:
  - (Unrelated) models used to support selected development activities where appropriate

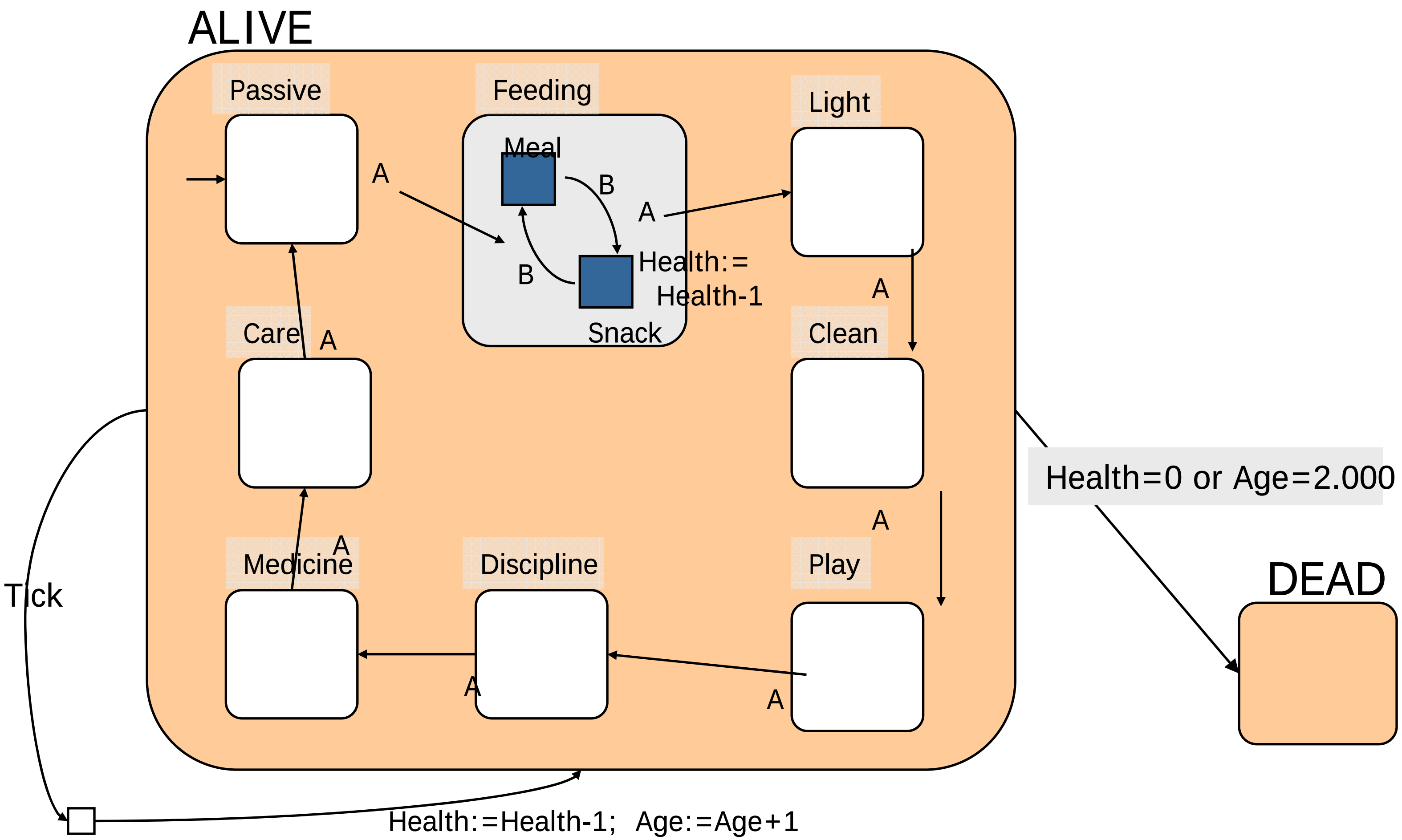
How?

Unified Model = State Machine!



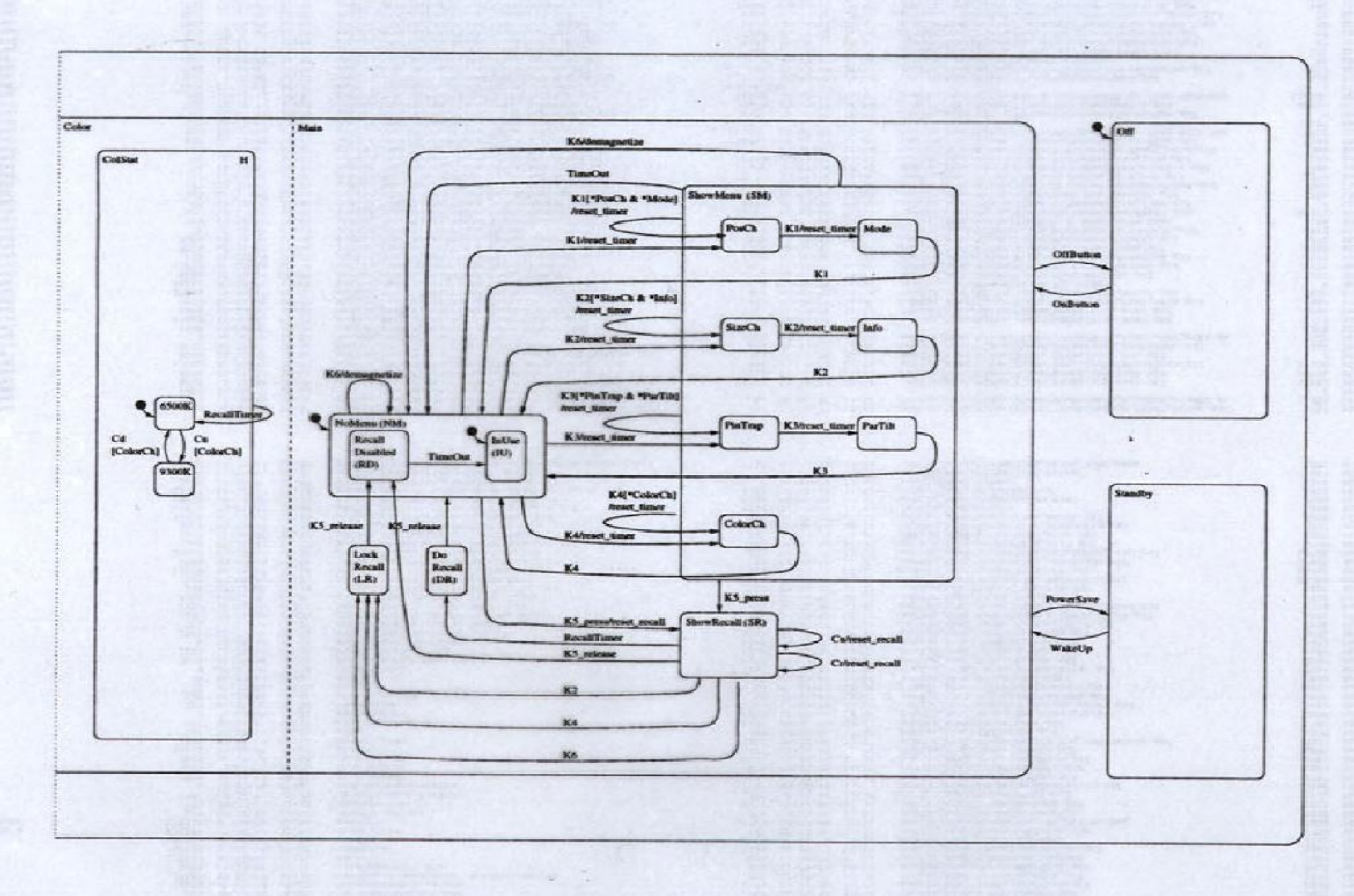
Tamagotchi





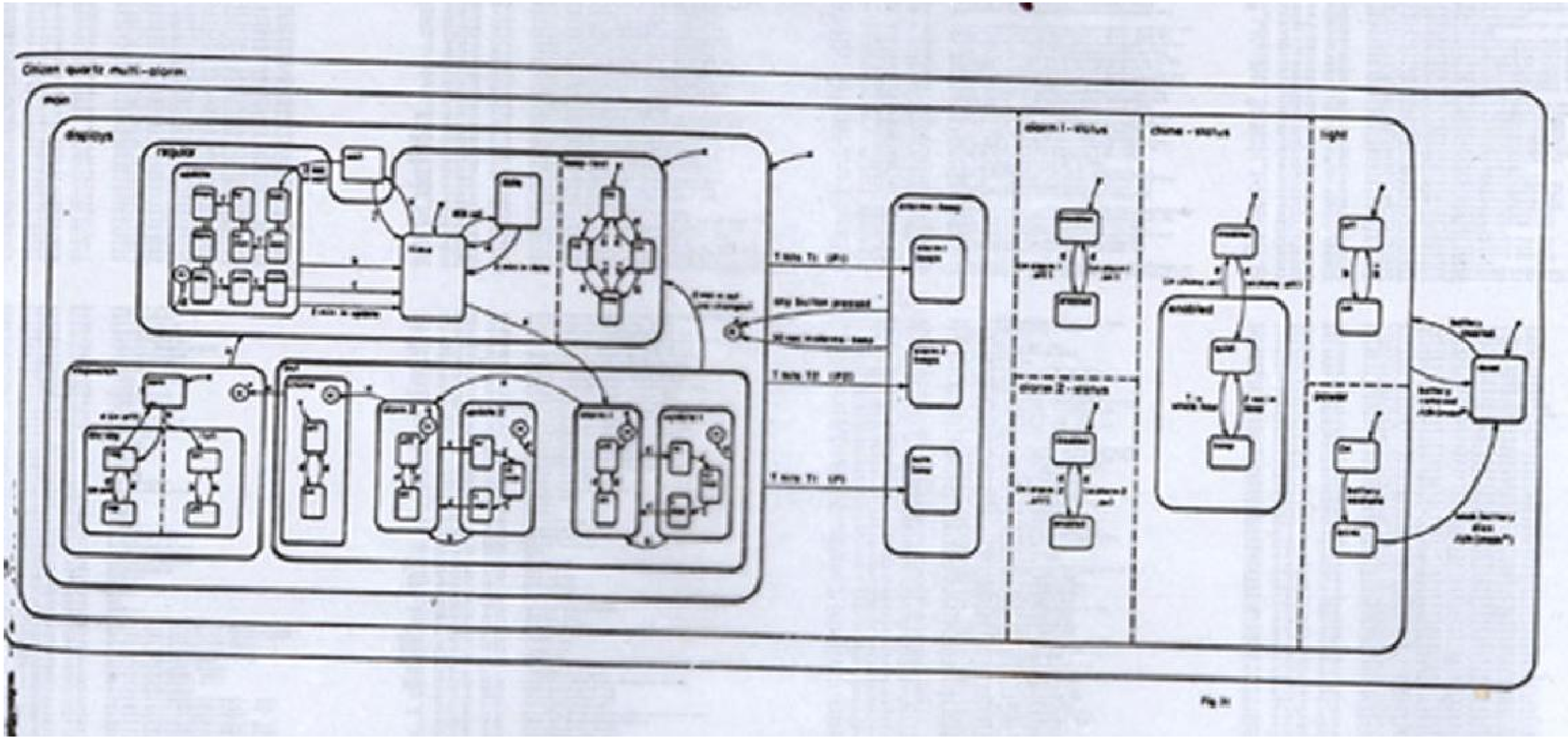
CISS

SYNCmaster

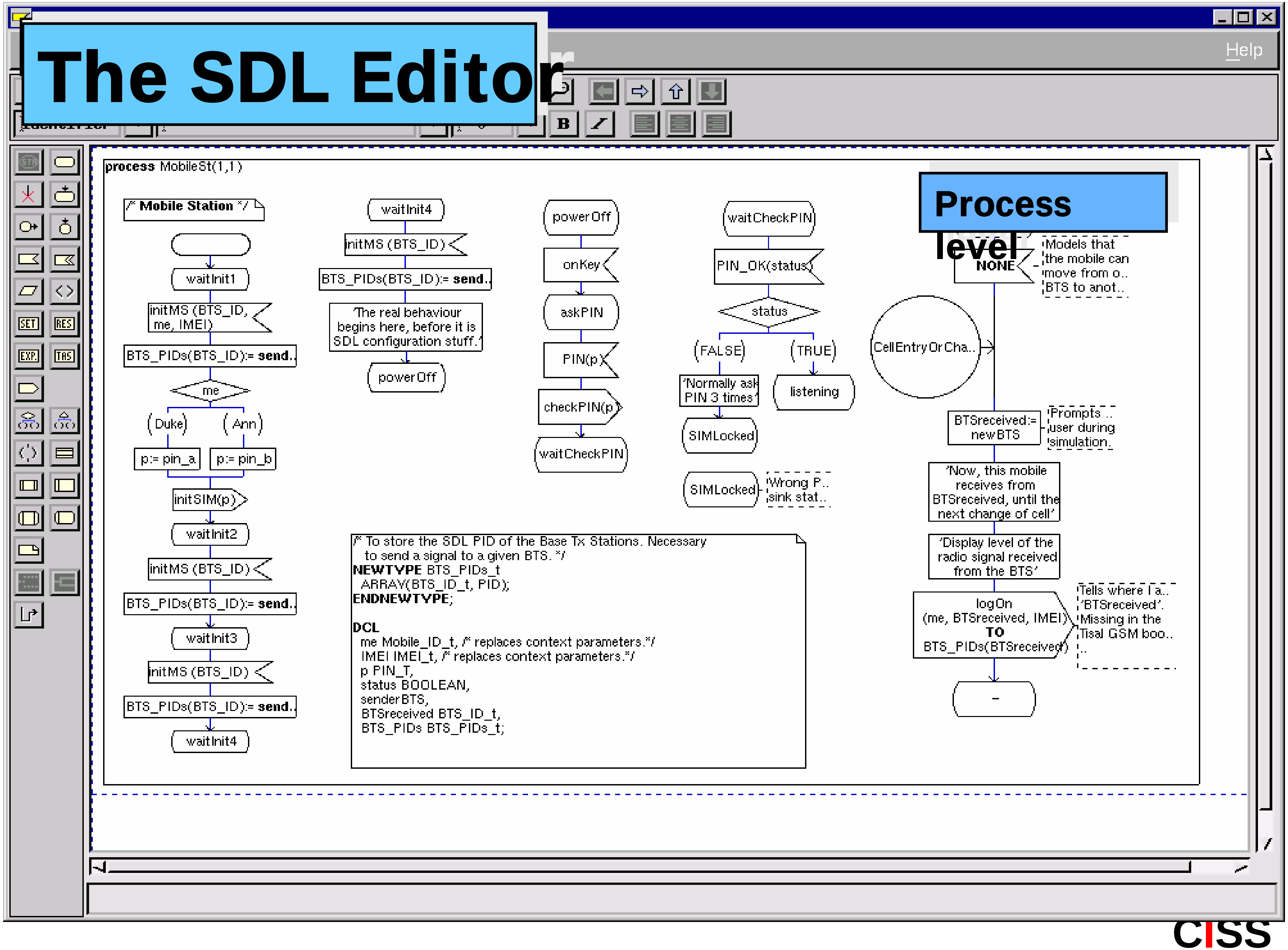


CISS

Digital Watch



The SDL Editor

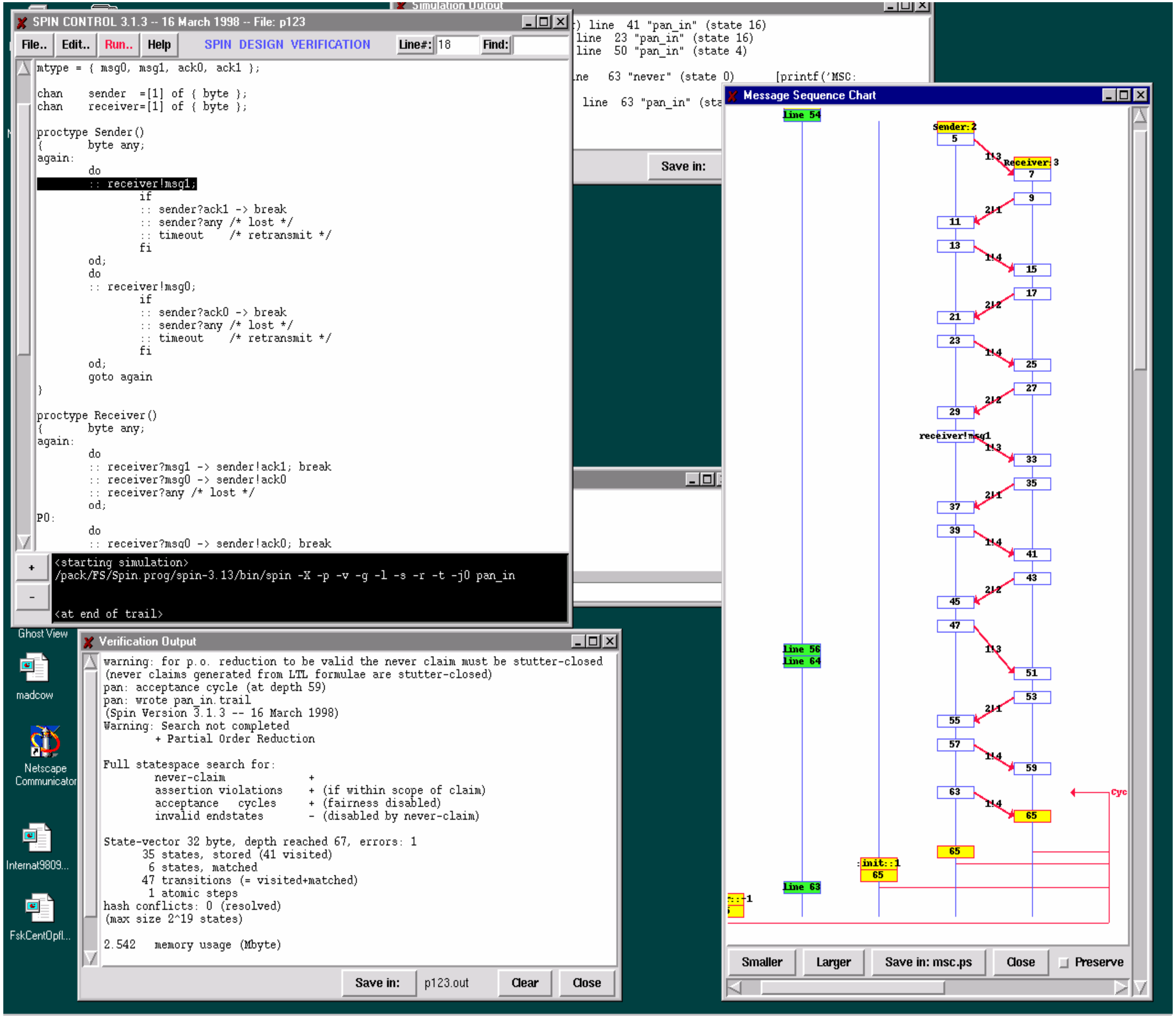


Process level

```
/* To store the SDL PID of the Base Tx Stations. Necessary
  to send a signal to a given BTS. */
NEWTYPE BTS_PIDs_t
  ARRAY(BTS_ID_t, PID);
ENDNEWTYPE;

DCL
  me Mobile_ID_t, /* replaces context parameters.*/
  IMEI IMEI_t, /* replaces context parameters.*/
  p PIN_T,
  status BOOLEAN,
  senderBTS,
  BTSreceived BTS_ID_t,
  BTS_PIDs BTS_PIDs_t;
```

Help
B
process MobileSt(1,1)
/* Mobile Station */
waitInit1
initMS (BTS_ID, me, IMEI)
BTS_PIDs(BTS_ID):= send..
me
Duke
Ann
p:= pin_a
p:= pin_b
initSIM(p)
waitInit2
initMS (BTS_ID)
BTS_PIDs(BTS_ID):= send..
waitInit3
initMS (BTS_ID)
BTS_PIDs(BTS_ID):= send..
waitInit4
waitInit4
initMS (BTS_ID)
BTS_PIDs(BTS_ID):= send..
'The real behaviour begins here, before it is SDL configuration stuff.'
powerOff
powerOff
onKey
askPIN
PIN(p)
checkPIN(p)
waitCheckPIN
waitCheckPIN
PIN_OK(status)
status
FALSE
TRUE
'Normally ask PIN 3 times'
SIMLocked
listening
SIMLocked
Wrong P.. sink stat..
NONE
Models that the mobile can move from o.. BTS to anot..
CellEntryOrCha..
BTSreceived:= newBTS
Prompts .. user during simulation.
'Now, this mobile receives from BTSreceived, until the next change of cell'
'Display level of the radio signal received from the BTS'
logOn (me, BTSreceived, IMEI) TO BTS_PIDs(BTSreceived)
Tells where I a.. 'BTSreceived'. Missing in the Tisal GSM boo.. ..
-

SPIN, Gerald Holzmann AT&T



```
mtype = { msg0, msg1, ack0, ack1 };

chan    sender  =[1] of { byte };
chan    receiver=[1] of { byte };

proctype Sender()
{       byte any;
again:
        do
        :: receiver!msg1;
                if
                :: sender?ack1 -> break
                :: sender?any /* lost */
                :: timeout    /* retransmit */
                fi
        od;
        do
        :: receiver!msg0;
                if
                :: sender?ack0 -> break
                :: sender?any /* lost */
                :: timeout    /* retransmit */
                fi
        od;
        goto again
}

proctype Receiver()
{       byte any;
again:
        do
        :: receiver?msg1 -> sender!ack1; break
        :: receiver?msg0 -> sender!ack0
        :: receiver?any /* lost */
        od;
P0:
        do
        :: receiver?msg0 -> sender!ack0; break
```

```
<starting simulation>
/pack/FS/Spin.prog/spin-3.13/bin/spin -X -p -v -g -l -s -r -t -j0 pan_in


<at end of trail>
```

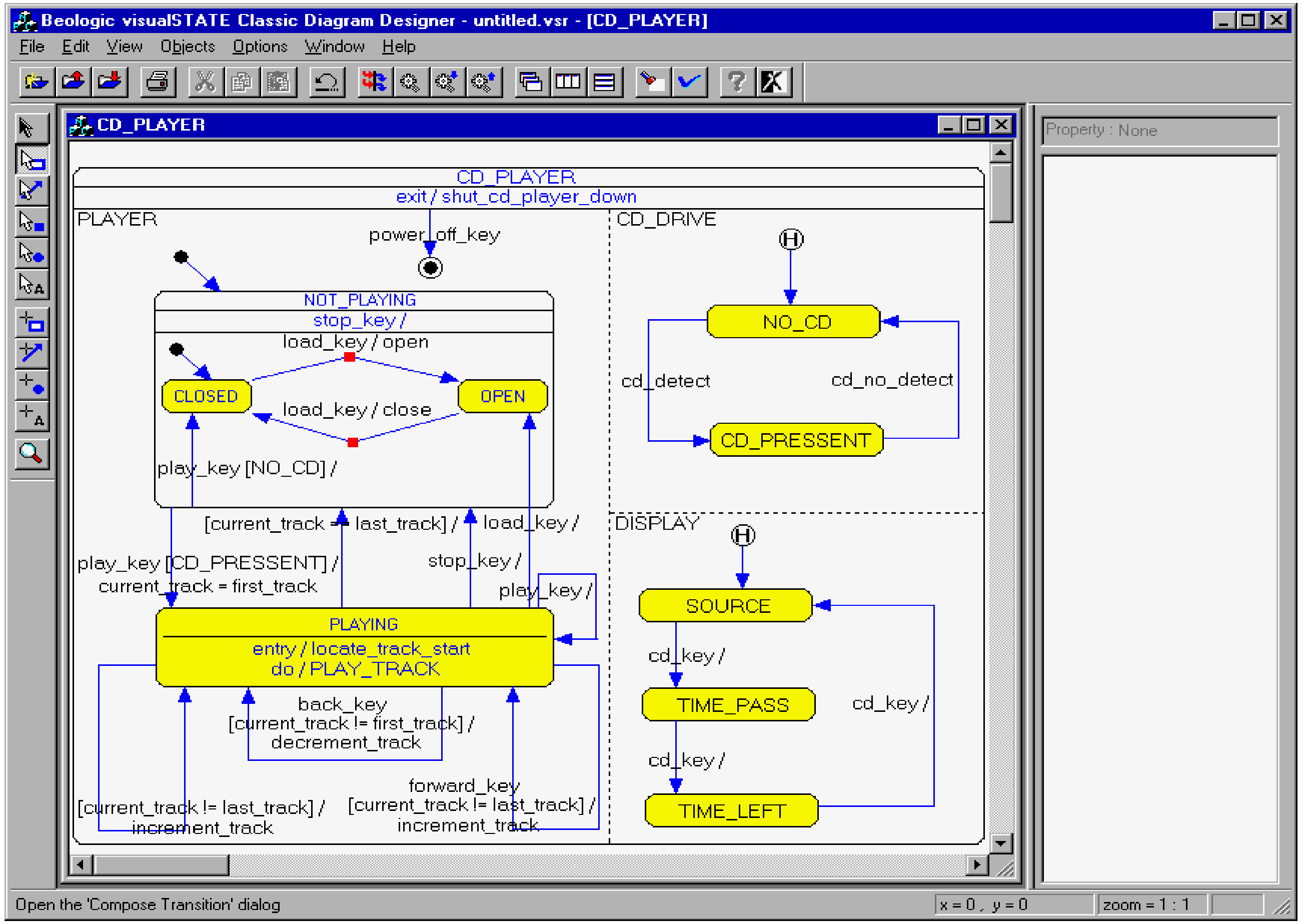
```
:) line  41 "pan_in" (state 16)
 line  23 "pan_in" (state 16)
 line  50 "pan_in" (state 4)

ne   63 "never" (state 0)       [printf('MSC:

 line  63 "pan_in" (sta
```

```
warning: for p.o. reduction to be valid the never claim must be stutter-closed
(never claims generated from LTL formulae are stutter-closed)
pan: acceptance cycle (at depth 59)
pan: wrote pan_in.trail
(Spin Version 3.1.3 -- 16 March 1998)
Warning: Search not completed
        + Partial Order Reduction

Full statespace search for:
        never-claim             +
        assertion violations    + (if within scope of claim)
        acceptance   cycles     + (fairness disabled)
        invalid endstates       - (disabled by never-claim)

State-vector 32 byte, depth reached 67, errors: 1
      35 states, stored (41 visited)
       6 states, matched
      47 transitions (= visited+matched)
       1 atomic steps
hash conflicts: 0 (resolved)
(max size 2^19 states)

2.542   memory usage (Mbyte)
```

visualSTATE VVS

w Baan Visualstate, DTU (CIT project)

- Hierarchical state systems
- Flat state systems
- Multiple and inter-related state machines
- Supports UML notation
- Device driver access

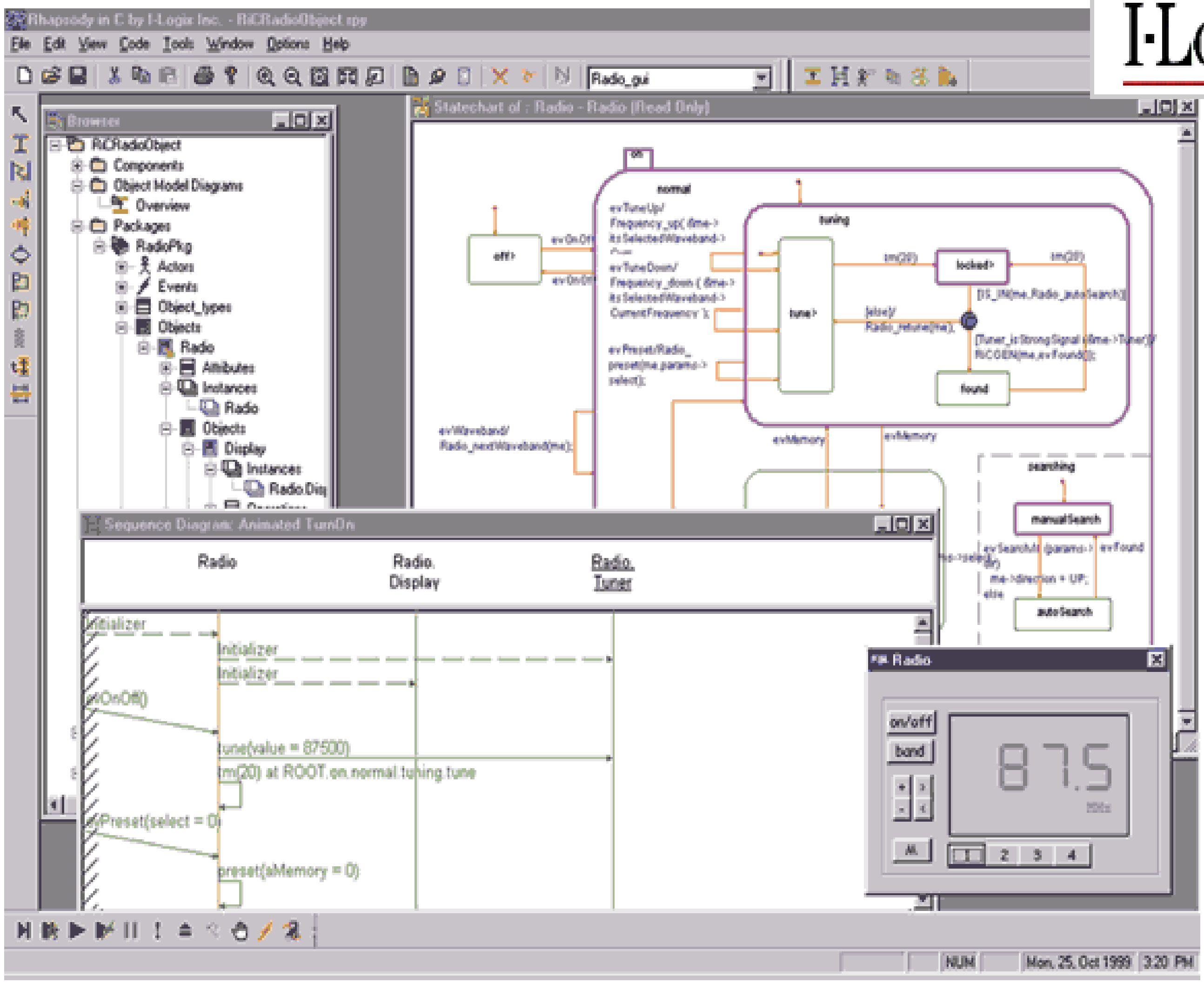


Beologic visualSTATE Classic Diagram Designer - untitled.vsr - [CD_PLAYER]

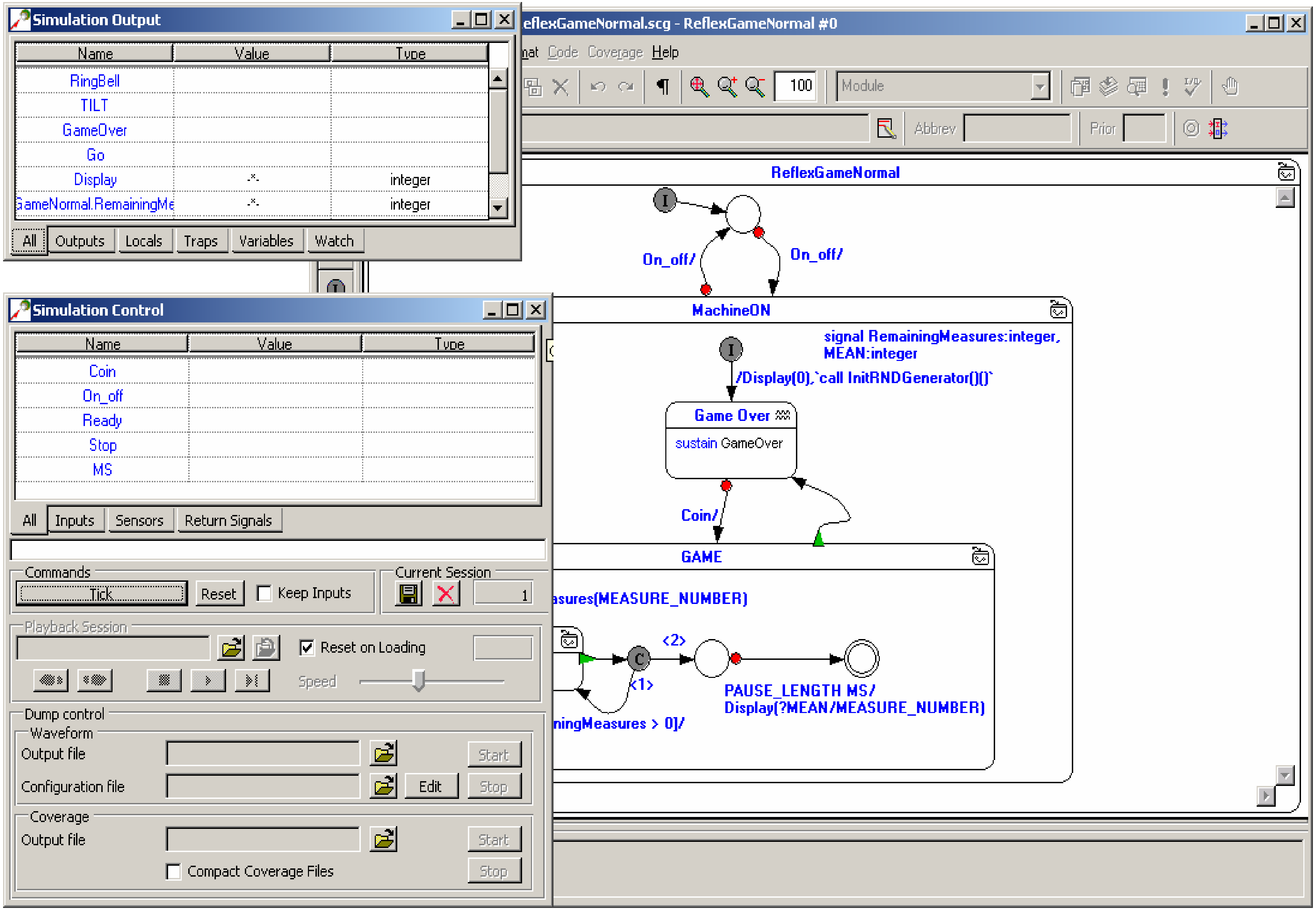
Open the 'Compose Transition' dialog

Rhapsody

Rhapsody®
I-Logix™



ESTEREL



Simulation Output

Name	Value	Type
RingBell		
TILT		
GameOver		
Go		
Display	-*-	integer
GameNormal.RemainingMe	-*-	integer

Simulation Control

Name	Value	Type
Coin		
On_off		
Ready		
Stop		
MS		

☐ Keep Inputs

Current Session: 1

☑ Reset on Loading

☐ Compact Coverage Files

eflexGameNormal.scg - ReflexGameNormal #0

Module: Module

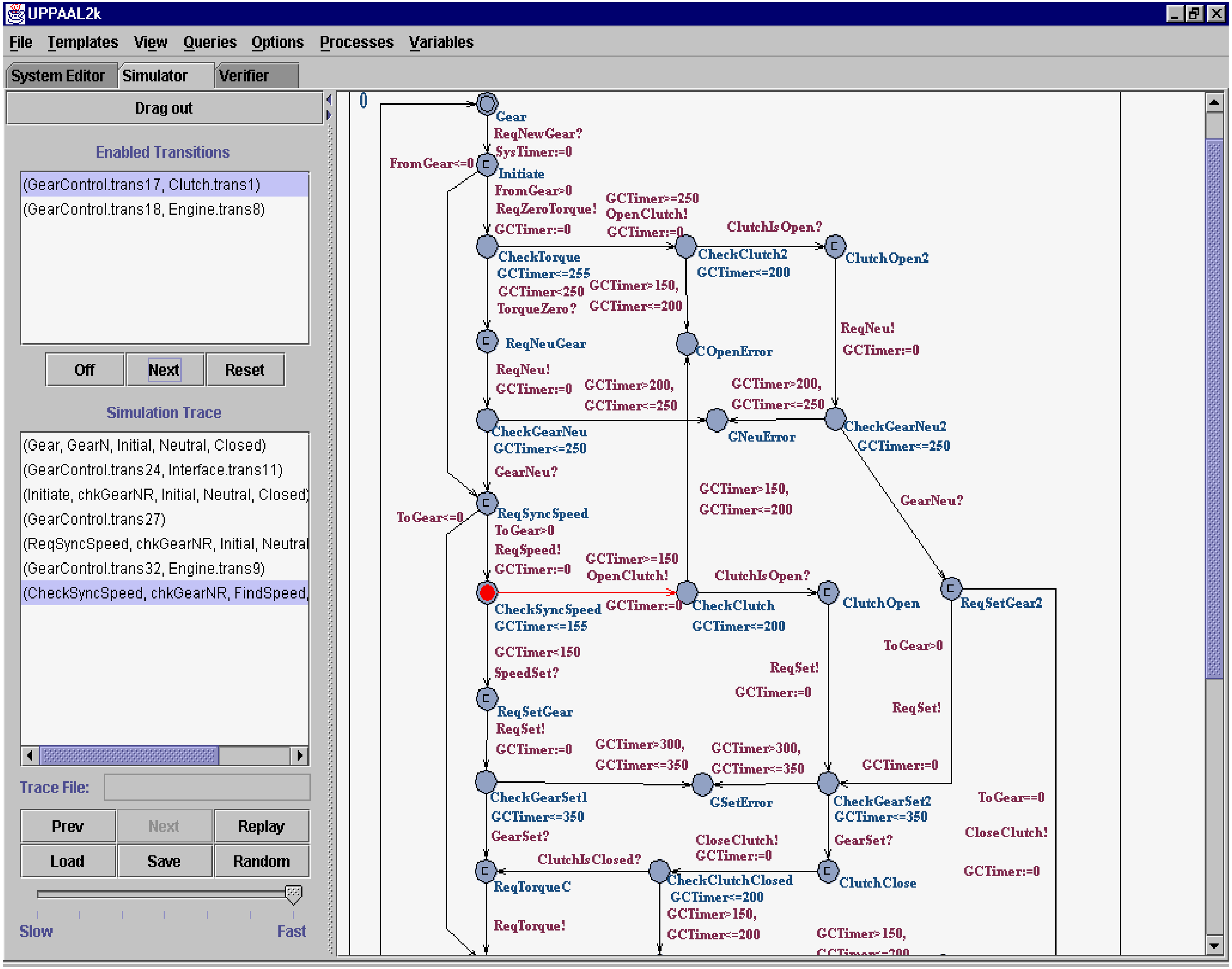
Zoom: 100

signal RemainingMeasures:integer,
MEAN:integer

/Display(0),`call InitRNDGenerator()()`

Game Over
sustain GameOver

UPPAAL



Enabled Transitions

```
(GearControl.trans17, Clutch.trans1)
(GearControl.trans18, Engine.trans8)
```

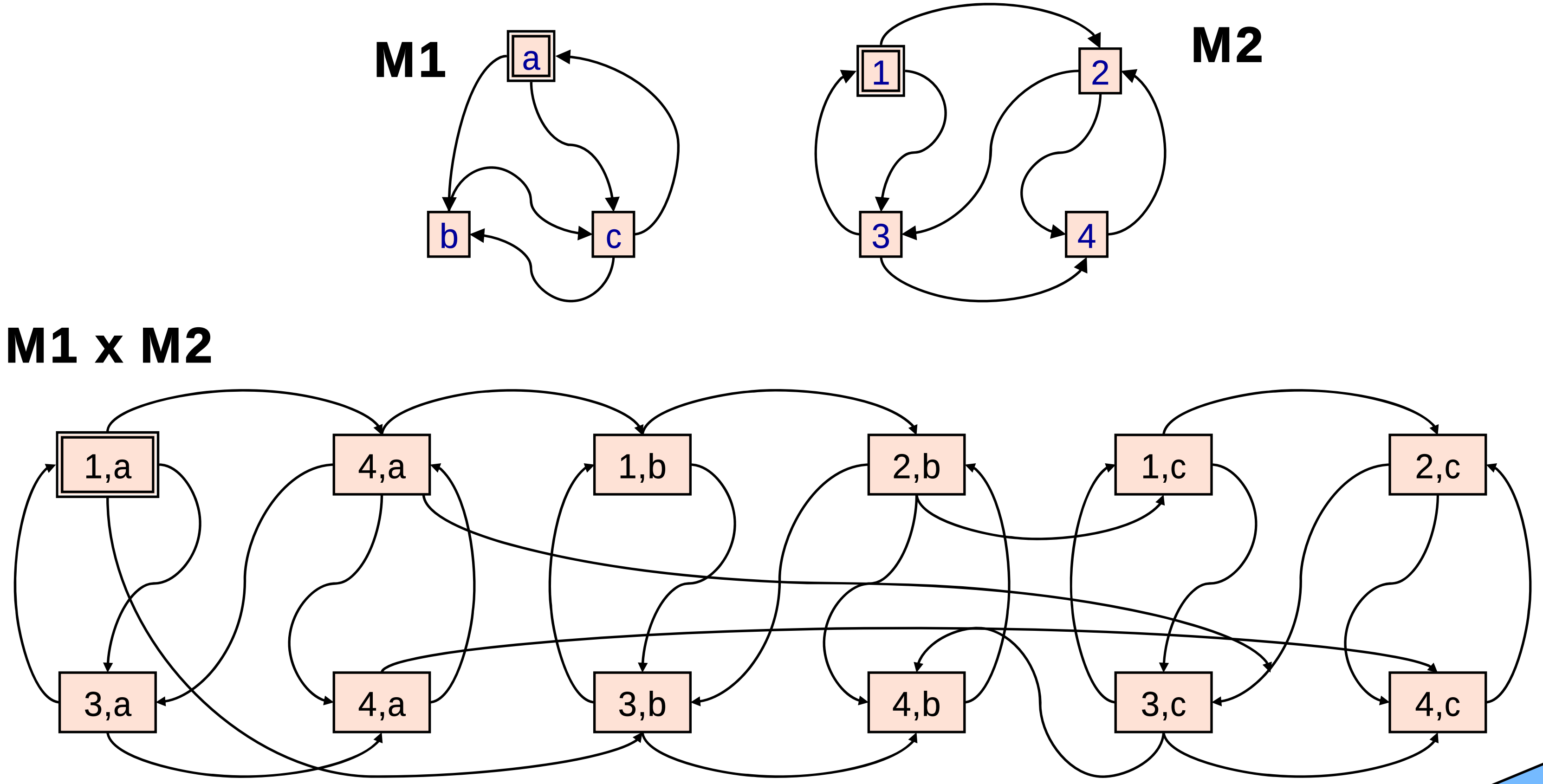
Simulation Trace

```
(Gear, GearN, Initial, Neutral, Closed)
(GearControl.trans24, Interface.trans11)
(Initiate, chkGearNR, Initial, Neutral, Closed)
(GearControl.trans27)
(ReqSyncSpeed, chkGearNR, Initial, Neutral
(GearControl.trans32, Engine.trans9)
(CheckSyncSpeed, chkGearNR, FindSpeed,
```

NModel

```
FSM(0,
     AcceptingStates(), Transitions(
     t(0,ShowTitles(),1),
     t(1,SortByFirst(),2),
     t(2,SortByMostRecent(),3),
     t(3,ShowText(),4)),
     Vocabulary("ShowTitles","ShowText",
     "SelectMessages","SelectTopics",
     "SortByFirst","SortByMostRecent")
)
```

'State Explosion' problem



All combinations = exponential in no. of compo

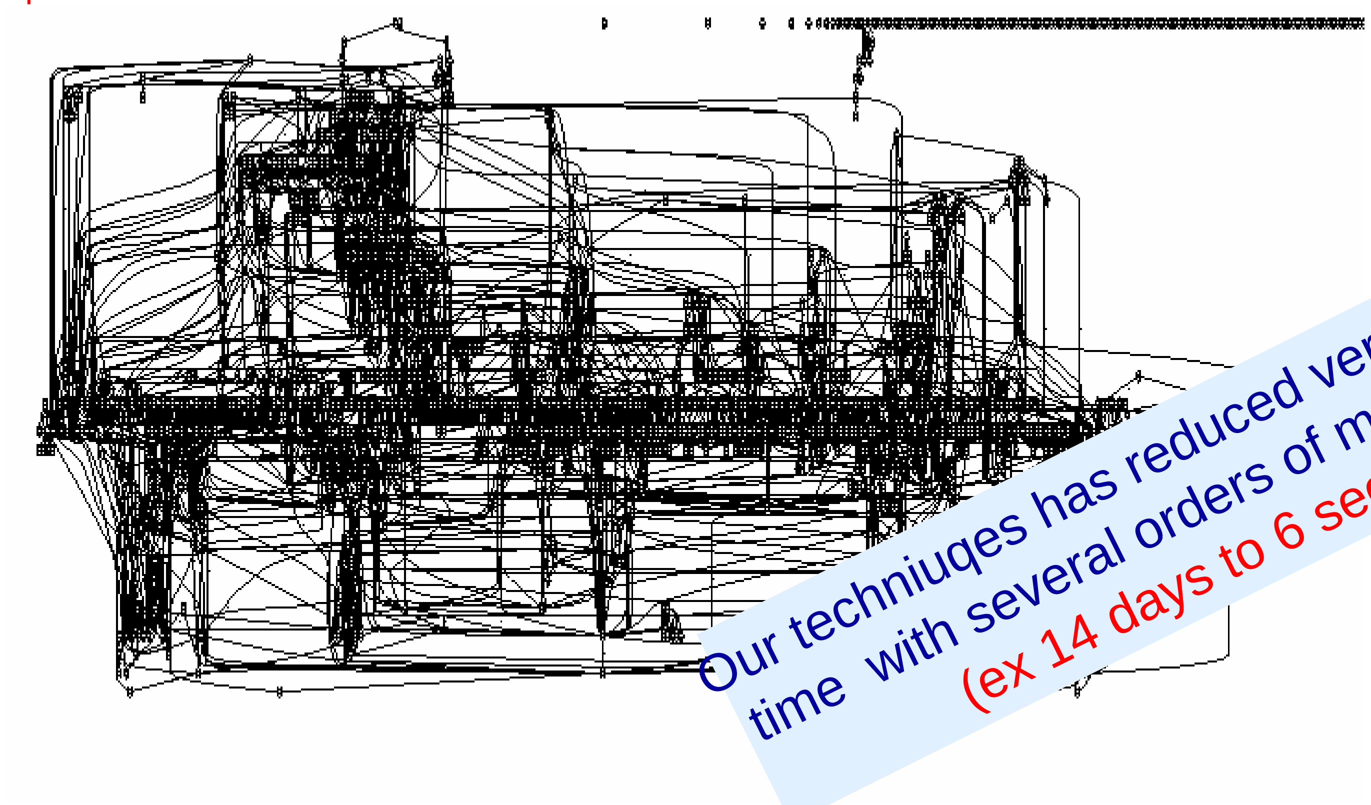
Provably theoretical intractable

Train Simulator

VVS
visualSTATE

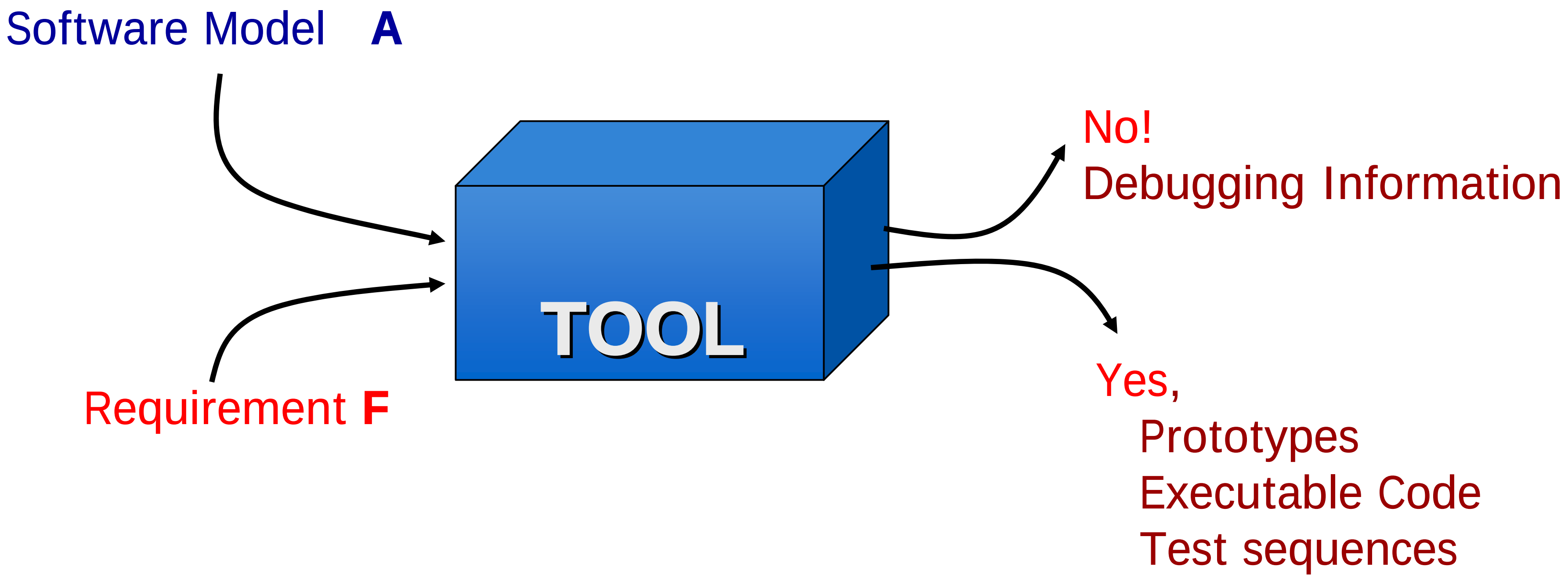
1421 machines
11102 transitions
2981 inputs
2667 outputs
3204 local states
Declare state sp.: 10^476

BUGS ?



Our techniuqes has reduced verification time with several orders of magnitude (ex 14 days to 6 sec)

Modelling and Analysis



Software Model **A**

Requirement **F**

TOOL

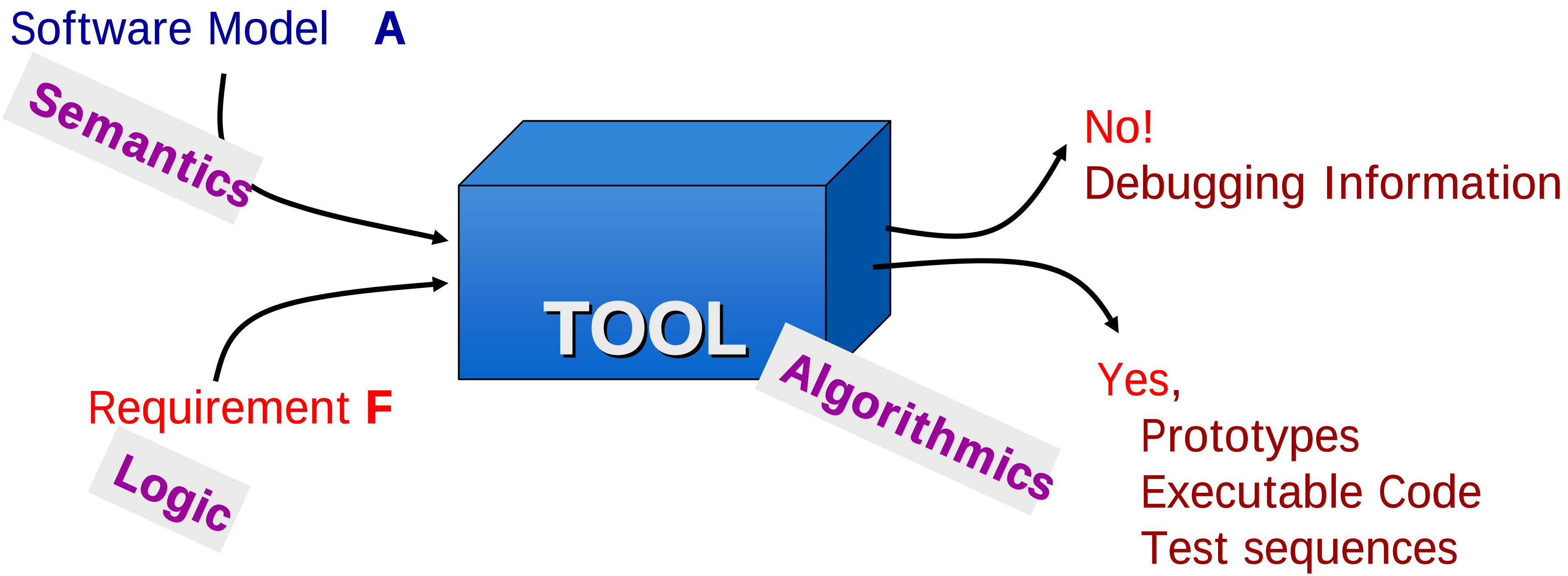
No!
Debugging Information

Yes,
Prototypes
Executable Code
Test sequences

Tools: UPPAAL, visualSTATE,
ESTEREL, SPIN, Statemate, FormalCheck,
VeriSoft, Java Pathfinder,...

Modelling and Analysis

BRICS



Software Model **A**

Semantics

Requirement **F**

Logic

TOOL

Algorithmics

No!
Debugging Information

Yes,
Prototypes
Executable Code
Test sequences

Tools: UPPAAL, visualSTATE,
ESTEREL, SPIN, Statemate, FormalCheck,
VeriSoft, Java Pathfinder,...

CISS

Most fundamentae
model in Computer Science:
Kleene og Moore

Finite State Machines

- Language versus behaviour
- Determinism versus non-determinism
- Composition and operations
- Variants of state machines
  Moore, Mealy, IO automater, UML ....

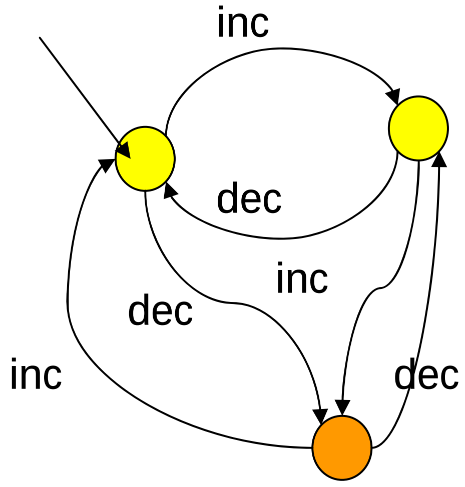
State Machines

0, 1, 2, 0, 1, 2, 0, 1,

Modulo 3 counter

Model of Computation

- Set of states
- A start state
- An input-alfabet
- A transition funktion, mapping input symbols and state to next state
- One ore more accept states.
- Computation starts from start state with a given input string (read from left to right)



inc inc dec inc inc dec inc ☹

inc inc dec inc dec inc dec inc ☺

input string

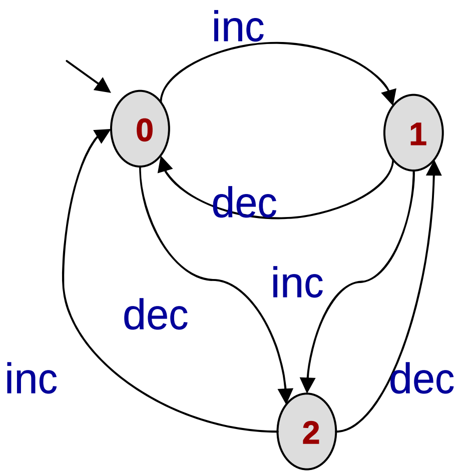
State Machines

Variants

Machines may have actions/output associated with state– Moore Machines.

inputstreng

inc inc dec inc inc dec inc



0 1 2 1 2 0 2 1

outputstreng

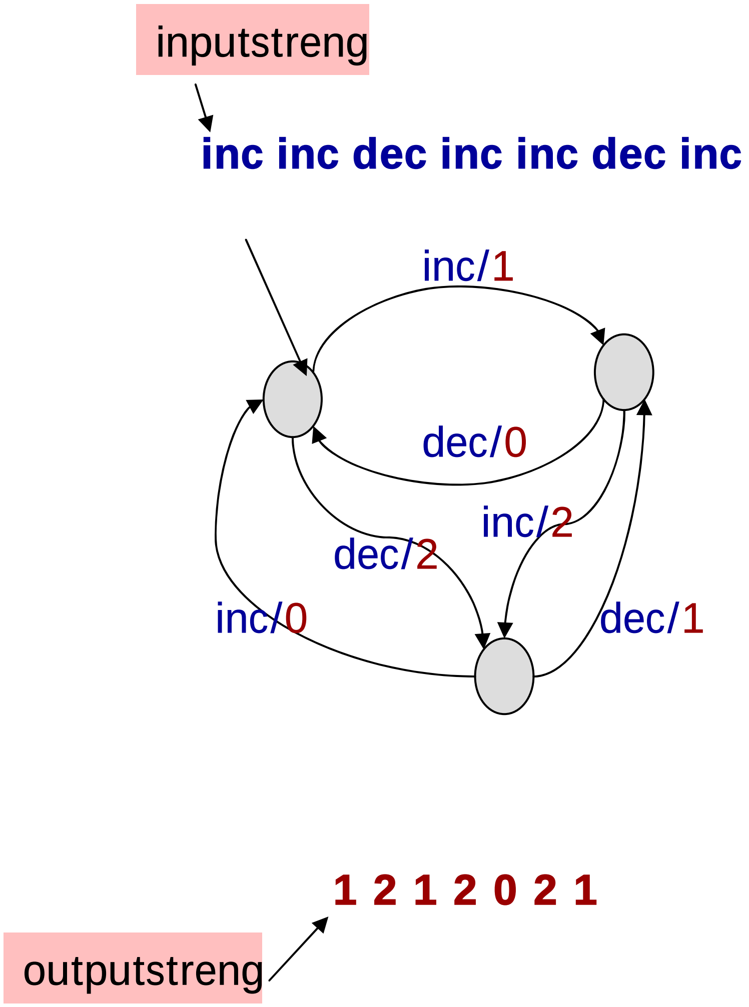
State Machines

Varianter

Machines may have actions/output associated with med transitions – Mealy Machiner.

Transitions unconditional of input (nul-transitions).

Several transitions for given for input and state
(non-determinisme).

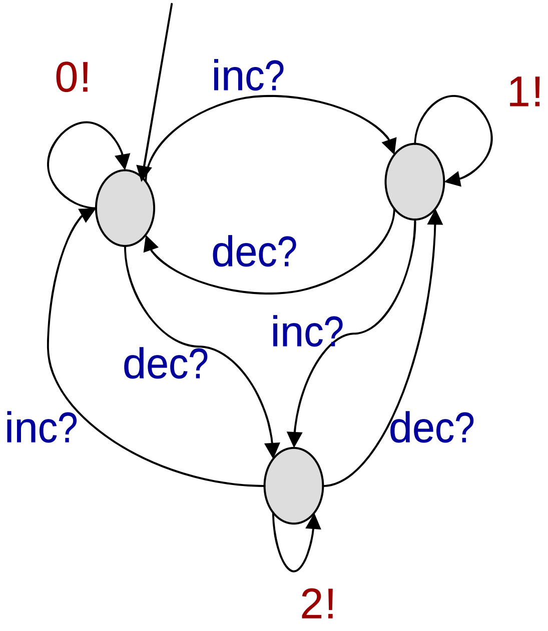


CISS

State Machines

Variants

Symbols of alphabet patitioned in input- and output-actions
(IO-automata)



0! 0! 0! inc? inc? 2! 2! dec? 1!

interaction

CISS

Interacting State Machines

AALBORG UNIVERSITY DENMARK

BRICS
Basic Research
in Computer Science

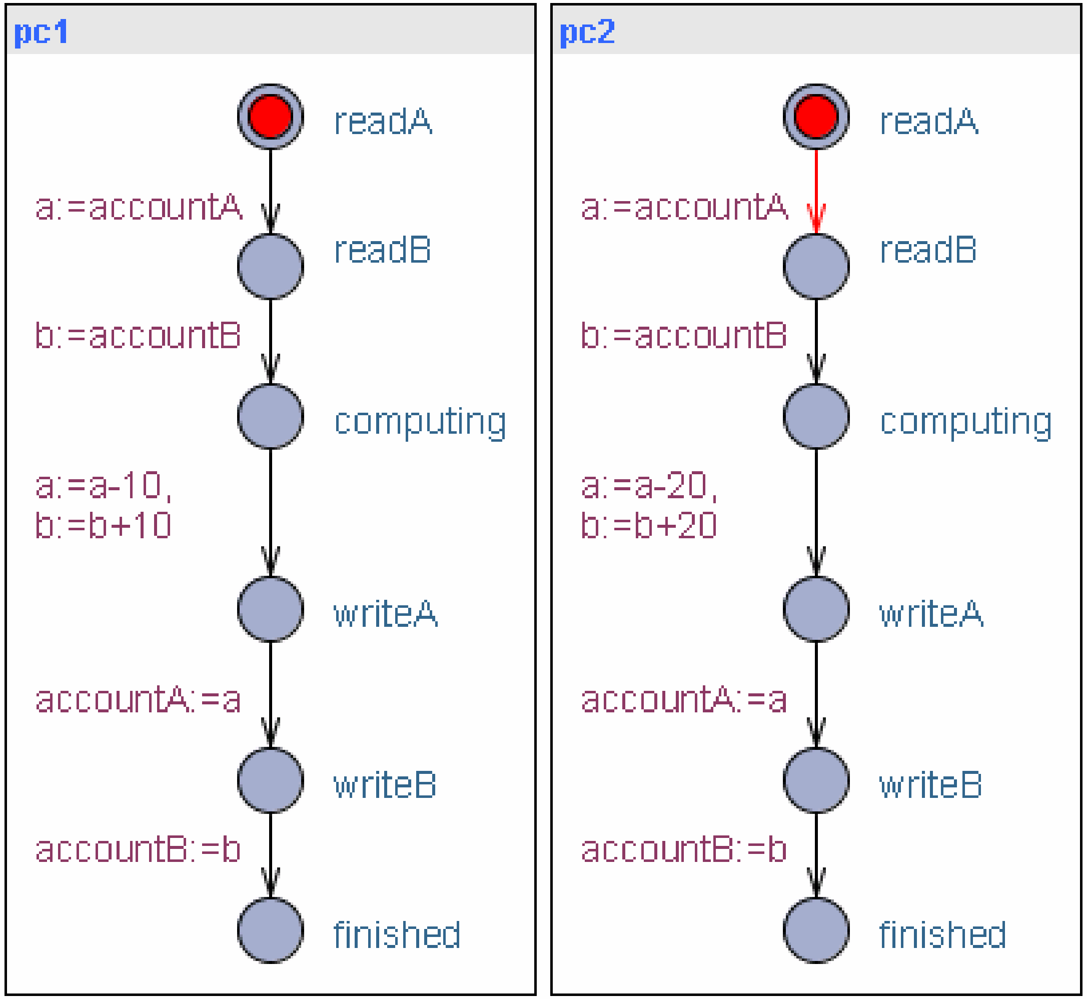
CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

Home-Banking?

```
int accountA, accountB; //Shared global variables
//Two concurrent bank costumers


Thread costumer1 () {       Thread costumer2 () {
  int a,b; //local tmp copy   int a,b;

  a=accountA;                 a=accountA;
  b=accountB;                 b=accountB;
  a=a-10;b=b+10;              a=a-20; b=b+20;
  accountA=a;                 accountA=a;
  accountB=b;                 accountB=b;
}                           }
```

- Are the accounts in balance after the transactions?

Home Banking



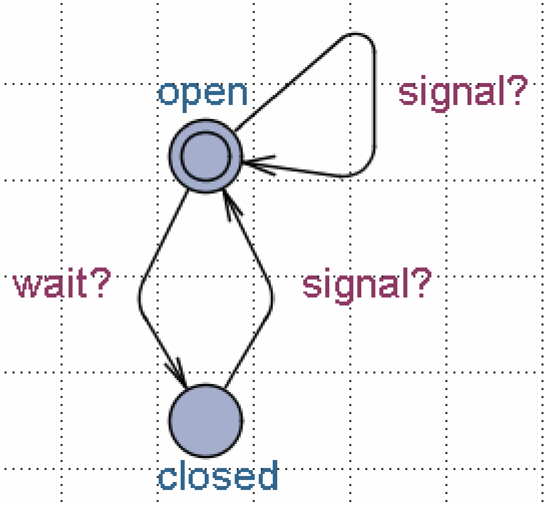
`A[] (pc1.finished and pc2.finished) imply (accountA+accountB==200)?`

Home Banking

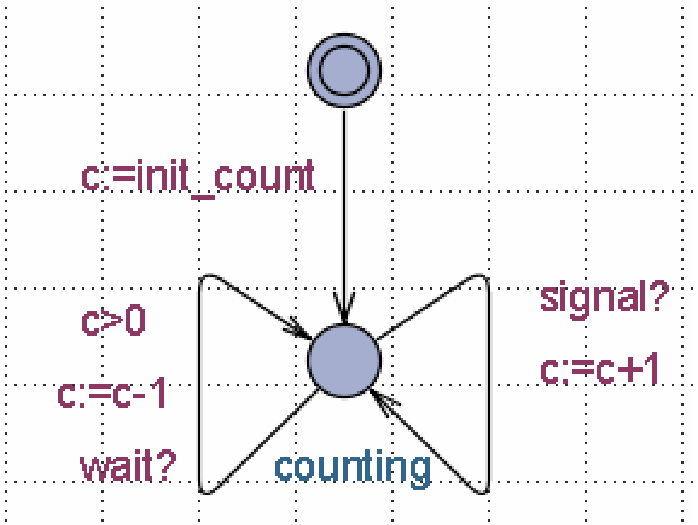
```
int accountA, accountB; //Shared global variables
Semaphore A,B;          //Protected by sem A,B
//Two concurrent bank costumers

Thread costumer1 () {       Thread costumer2 () {
  int a,b; //local tmp copy   int a,b;

  wait(A);                    wait(B);
  wait(B);                    wait(A);
  a=accountA;                 a=accountA;
  b=accountB;                 b=accountB;
  a=a-10;b=b+10;              a=a-20; b=b+20;
  accountA=a;                 accountA=a;
  accountB=b;                 accountB=b;
  signal(A);                  signal(B);
  signal(B);                  signal(A);
}                           }
```

Semaphore FSM Model

Binary Semaphore

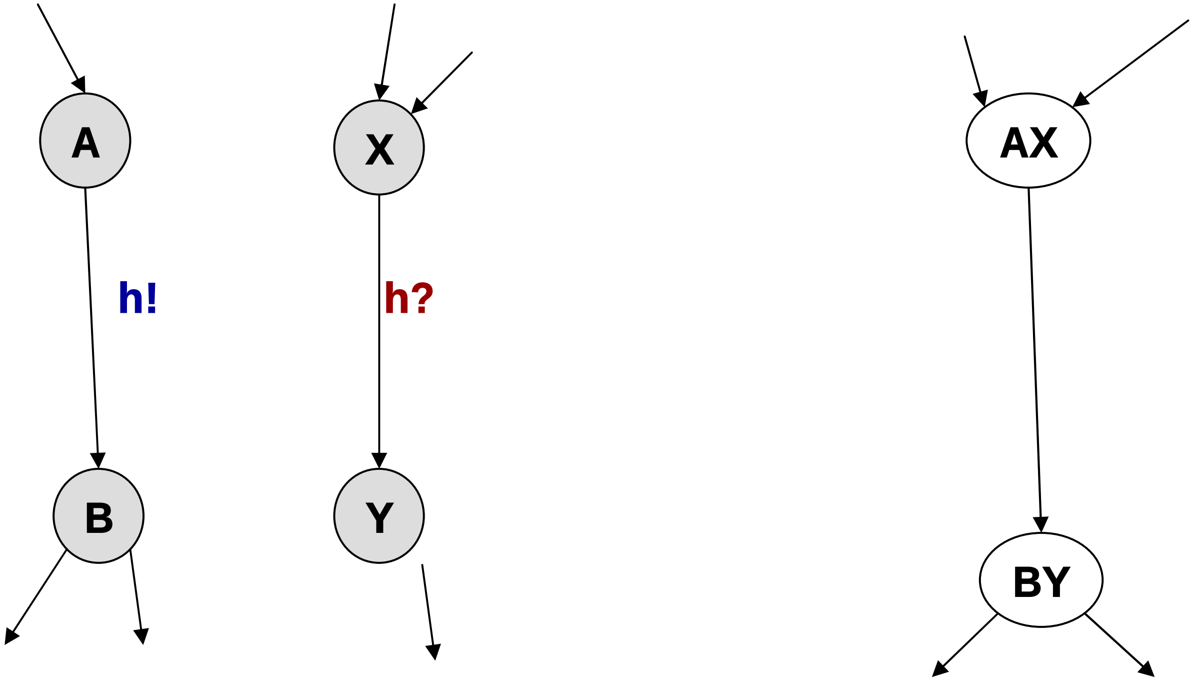


Counting Semaphore



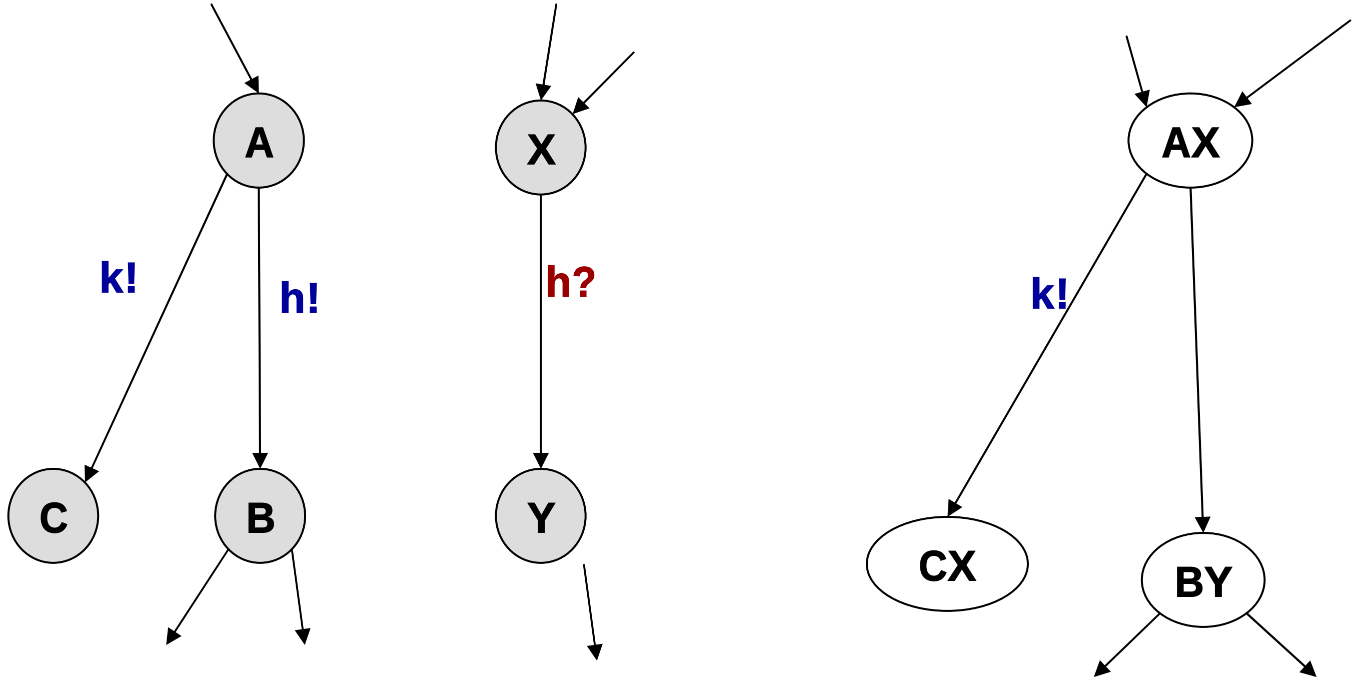
Composition

IO Automater (2-vejs synkronisering)



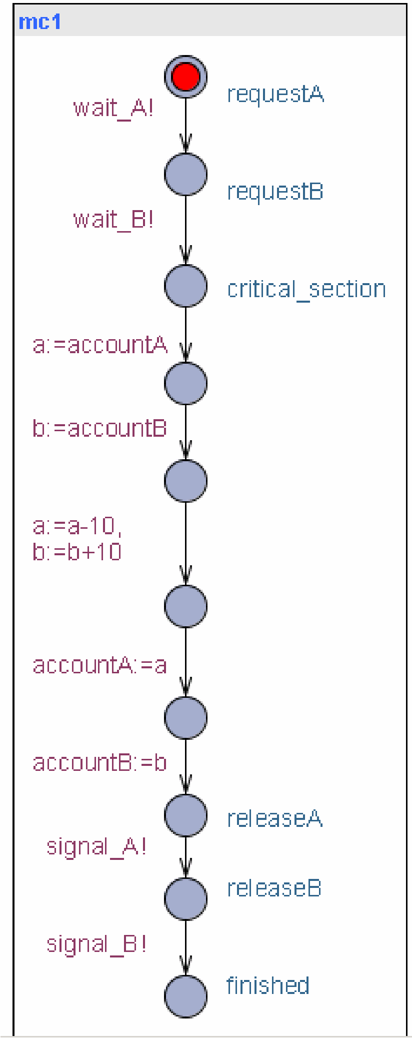
Composition

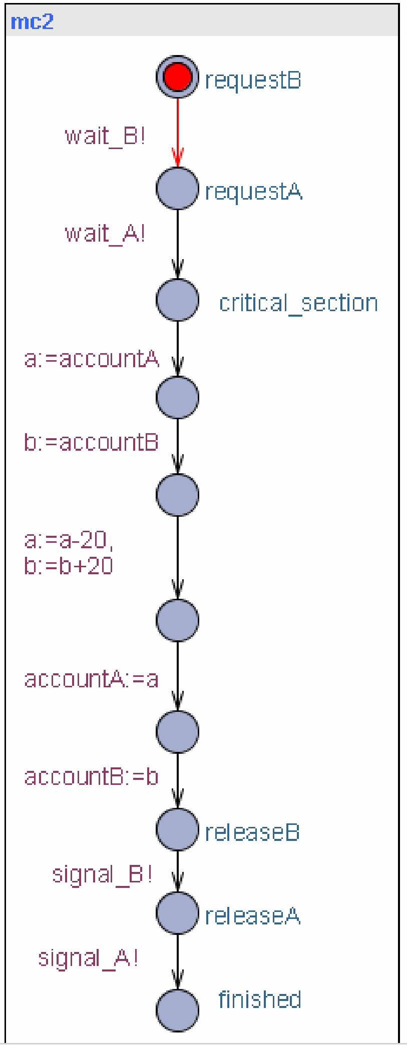
IO Automater (2-vejs synkronisering)

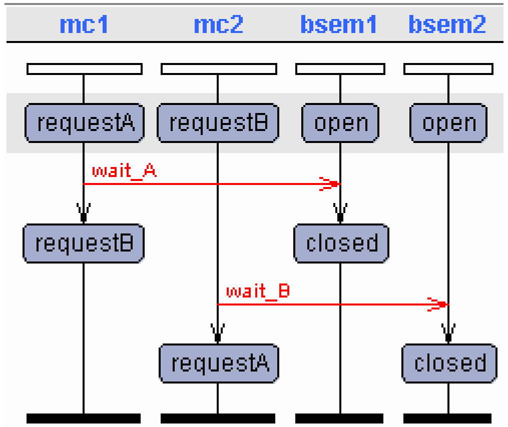


CISS

Semaphore Solution?







1. Consistency? (Balance)
2. Race conditions?
3. Deadlock?

1. A[] (mc1.finished and mc2.finished) imply (accountA+accountB==200) ✓
2. E<> mc1.critical_section and mc2.critical_section ✓
3. A[] not (mc1.finished and mc2.finished) imply not deadlock ÷