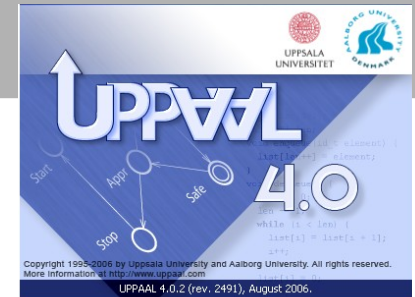


UPPAAL

Model Checking, Performance Analysis
and Testing of
Real Time Systems



Kim G. Larsen

CISS – Aalborg University

DENMARK



Regional ICT Center (2002-)

- 3 research groups
 - Computer Science
 - Control Theory
 - Hardware
 - Wireless Communication
- 20 Employed
- 25 Associated
- 20 PhD Students
- 50 Industrial projects
- 10 Elite-students
- 140+ MDKK
- ARTIST Design
- ARTEMIS
-

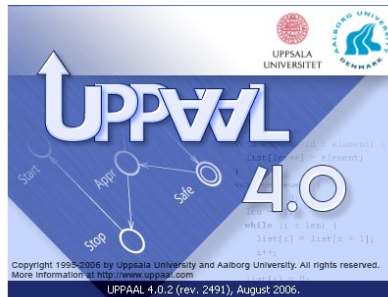


Characteristica:

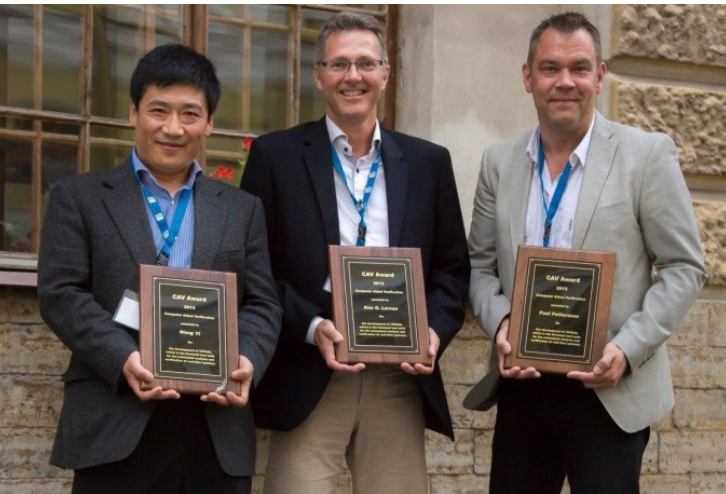
- Dedicated function
- Complex environment
- SW/HW/Mechanics
- Networked
- Autonomous
- Ressource constrained
 - : Energy
 - : Bandwidth
 - : Memory
 - : ...
- Timing constraints**



Model Checking & Performance Analysis



Origin of UPPAAL



TAU
CCS & Modal Transition Systems
Refinements
Modal Mu-Calculus
Explicit State Representation
Prolog

UPPAAL
Timed Automata
TCTL
Zones
C++ & Java

EPSILON
TCCS
Timed Refinements
Timed Mu-Calculus
Regions
Prolog<

UP4ALL

CAV Award

1995

2007

2013

Contributors



@UPPsala



- Wang Yi
- Paul Pettersson
- John Håkansson
- Anders Hessel
- Pavel Krcal
- Leonid Mokrushin
- Shi Xiaochun

@AALborg



- Kim G Larsen
- Alexandre Davic
- Gerd Behrman
- Arne Skou
- Brian Nielsen
- Jacob I. Rasmussen
- Marius Mikucionis
- Thomas Chatain

@Elsewhere

- Emmanuel Fleury, Didier Lime, Johan Bengtsson, Fredrik Larsson, Kåre J Kristoffersen, Tobias Amnell, Thomas Hune, Oliver Möller, Elena Fersman, Carsten Weise, David Griffioen, Ansgar Fehnker, Frits Vandraager, Theo Ruys, Pedro D'Argenio, J-P Katoen, Jan Tretmans, Judi Romijn, Ed Brinksma, Martijn Hendriks, Klaus Havelund, Franck Cassez, Magnus Lindahl, Francois Laroussinie, Patricia Bouyer, Augusto Burgueno, H. Bowmann, D. Latella, M. Massink, G. Faconti, Kristina Lundqvist, Lars Asplund, Justin Pearson...

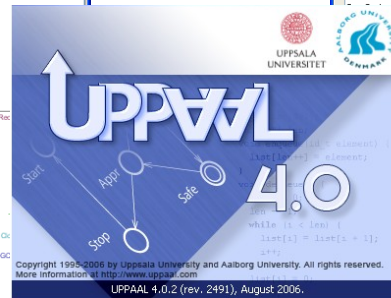
UPPAAL Model Checker



Editor

Discrete Control
Concurrency
Continuous Aspects
Stochasticity
Timing Constraints
Resources

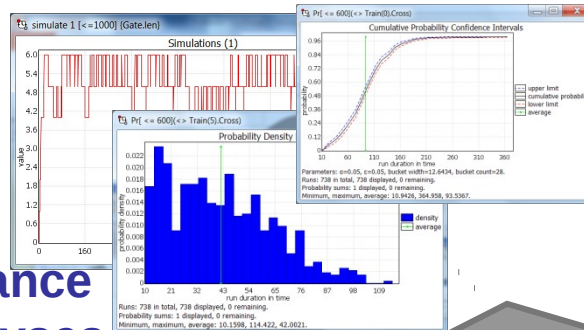
Simulator



Verifier

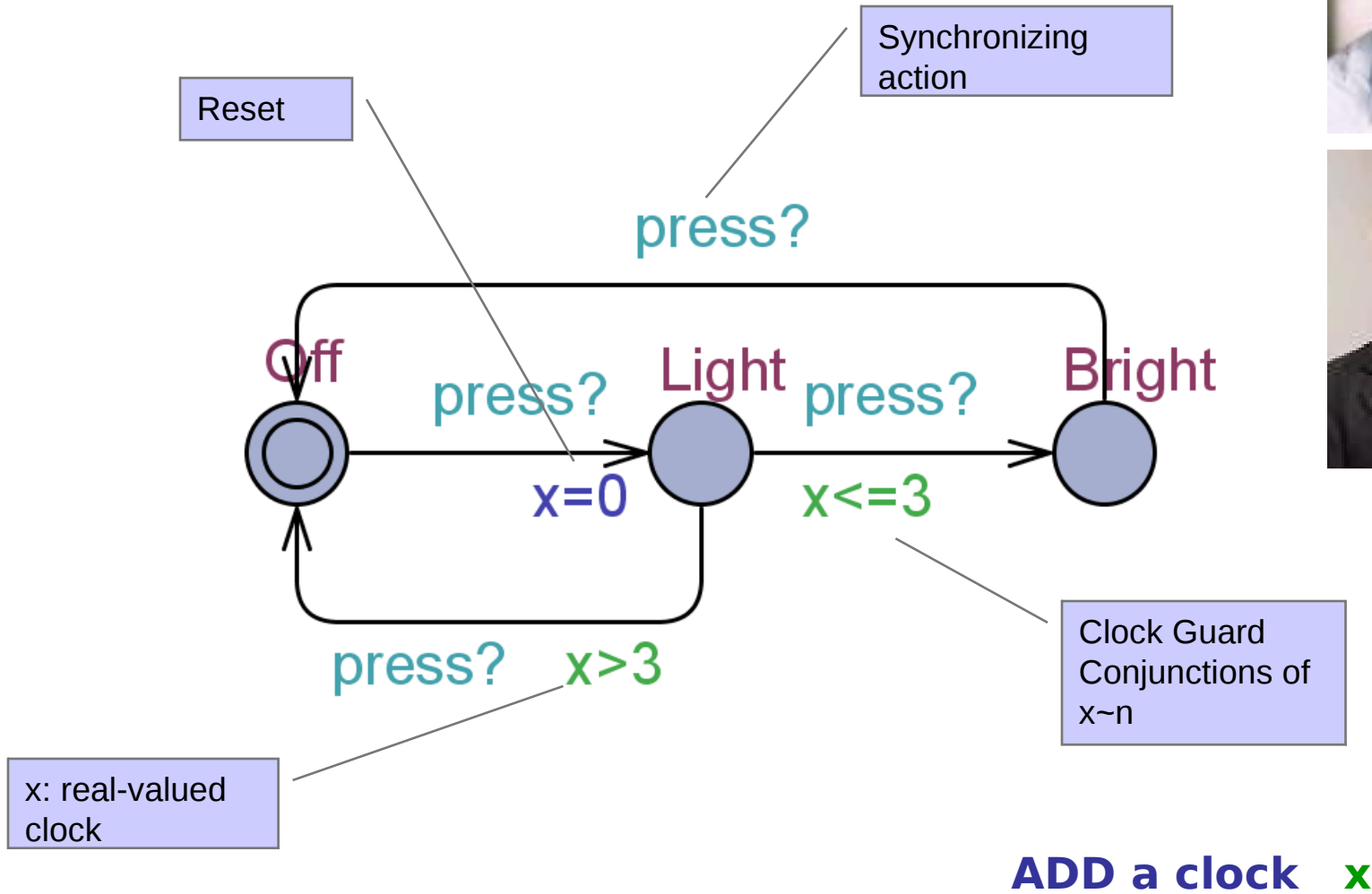
```

E<> GearControl.GearChanged
E<> ( Interface.Gear5
E<> ( Interface.GearR
E<> ( GearControl.GearChanged and ( SysTimer<=100 )
A[] not ( GearBox.Neutral and ( Interface.Gear1 or I...
Query
E<> GearControl.GearChanged
Comment
P1. It is possible to change gear.
Status
Property is satisfied.
A[] ( Clutch.Closed imply ( GearControl.ReqTorqueC or GearControl.GearChanged or GearControl.Gear or GearC...
Property is satisfied.
A[] ( GearBox.Idle imply ( GearControl.ClutchClose or GearControl.CheckClutchClosed or GearControl.CClose...
Property is satisfied.
A[] ( GearBox.Neutral imply ( GearControl.ReqSetGear or GearControl.ChkClutchClosed or GearControl.C...
Property is satisfied.
    
```

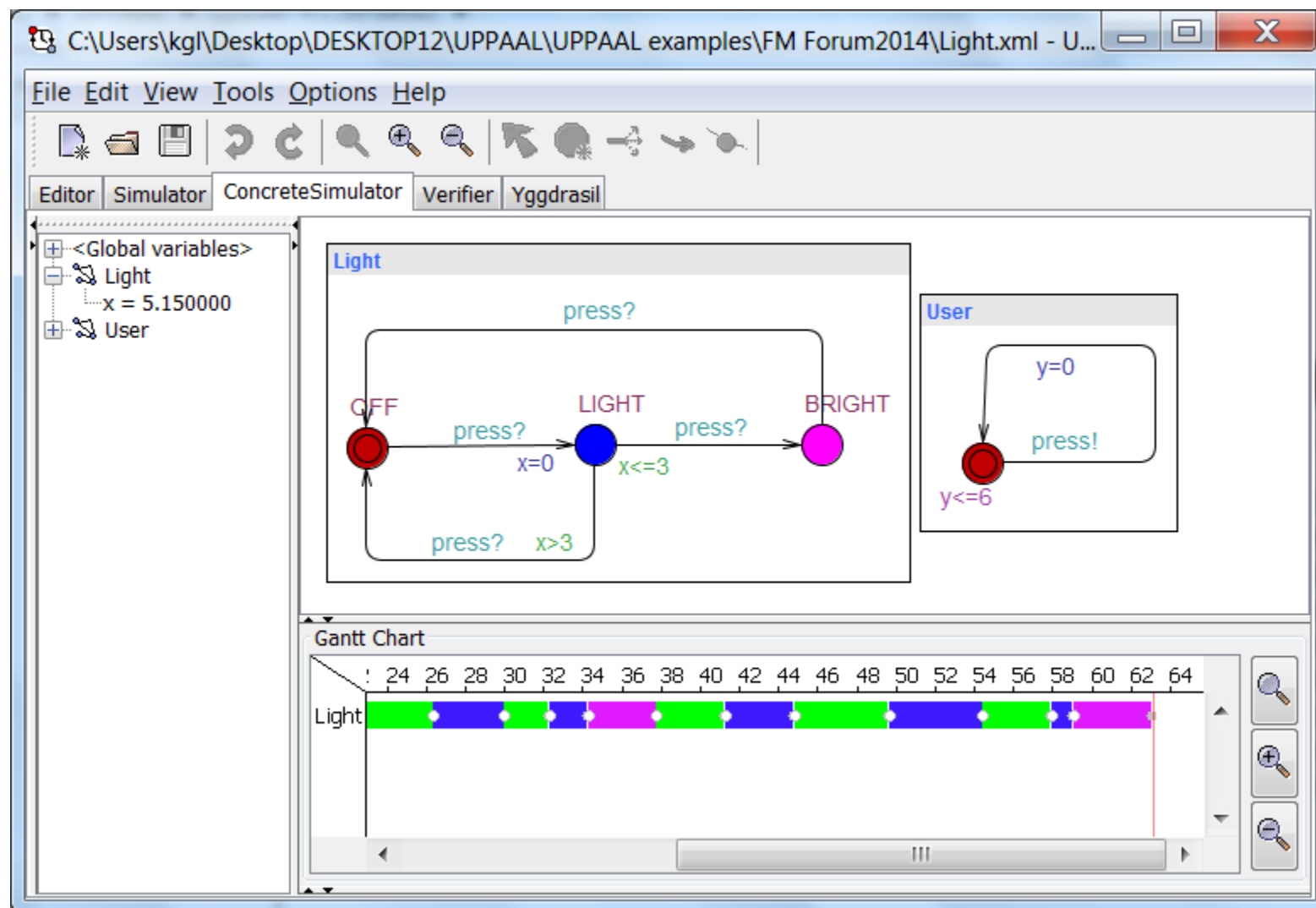


Timed Automata

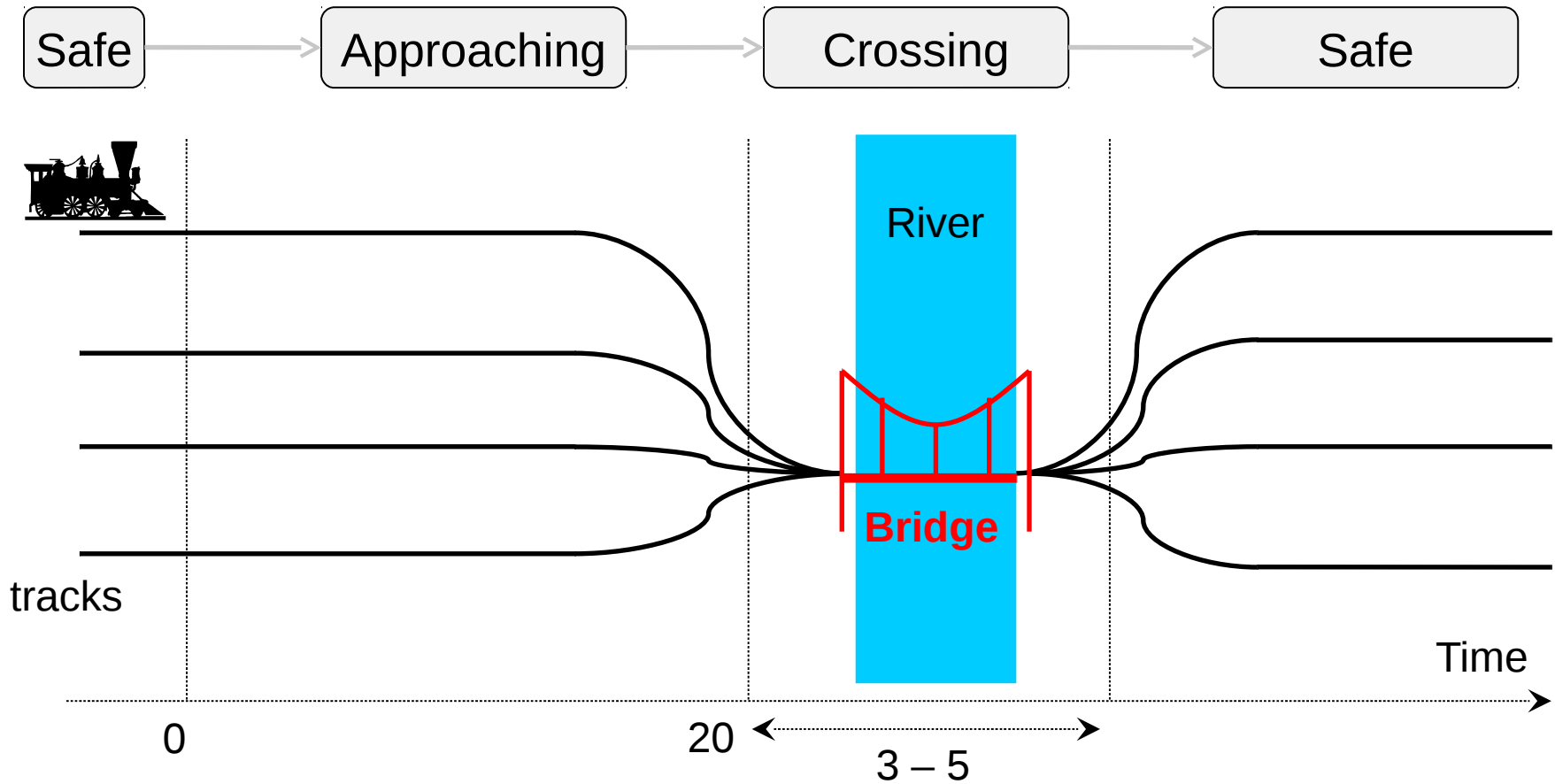
[Alur & Dill'89]



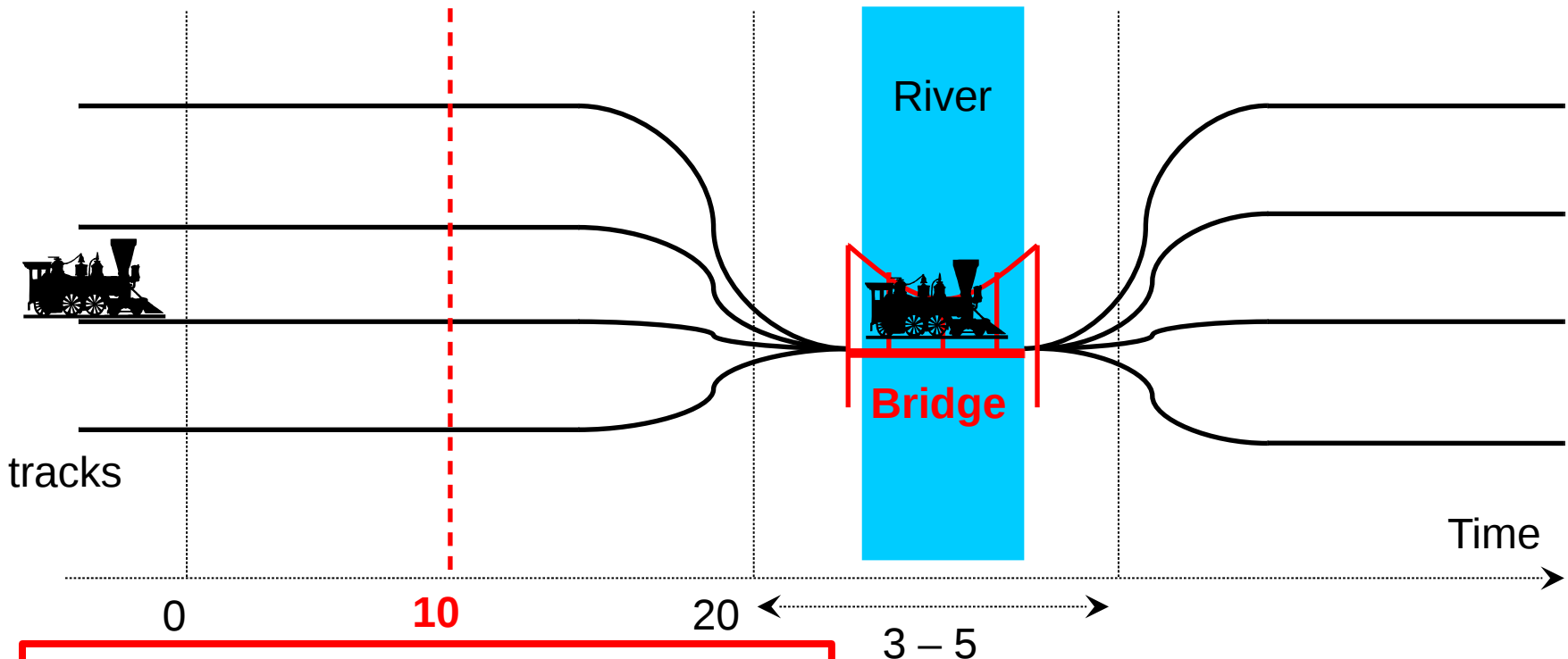
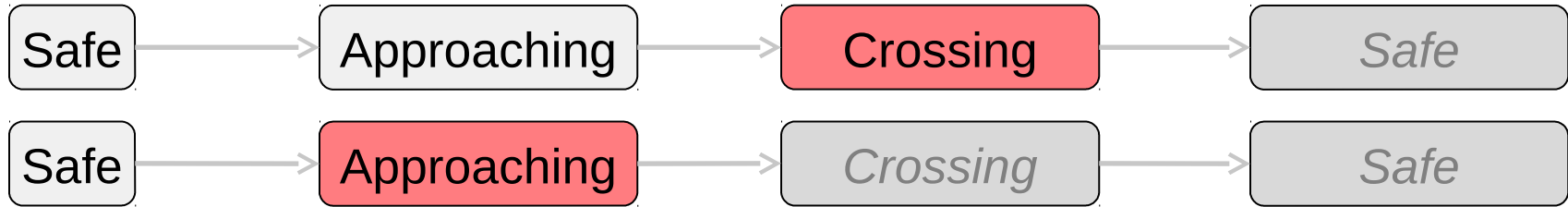
Semantics in UPPAAL



Train Crossing

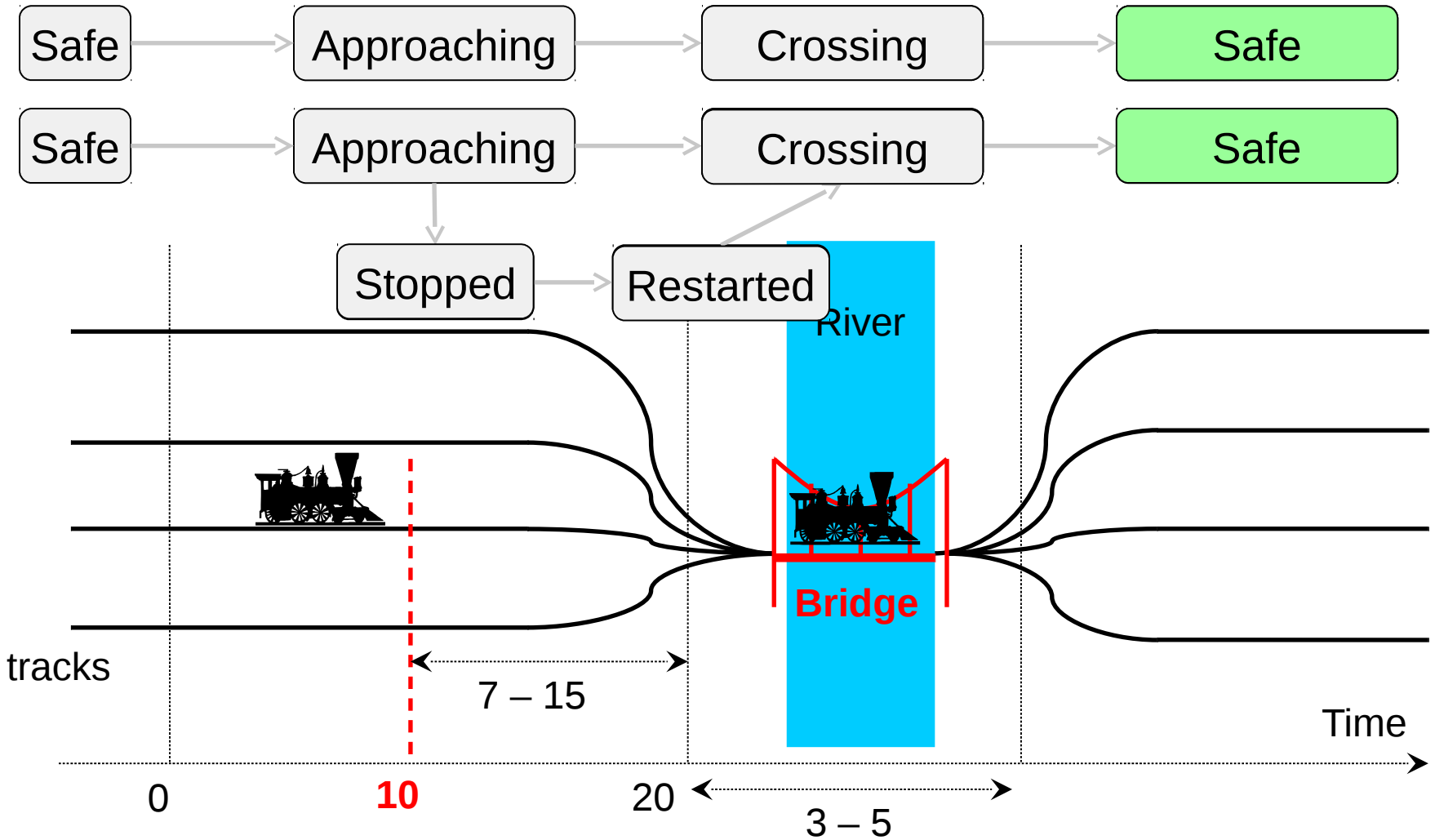


Train Crossing

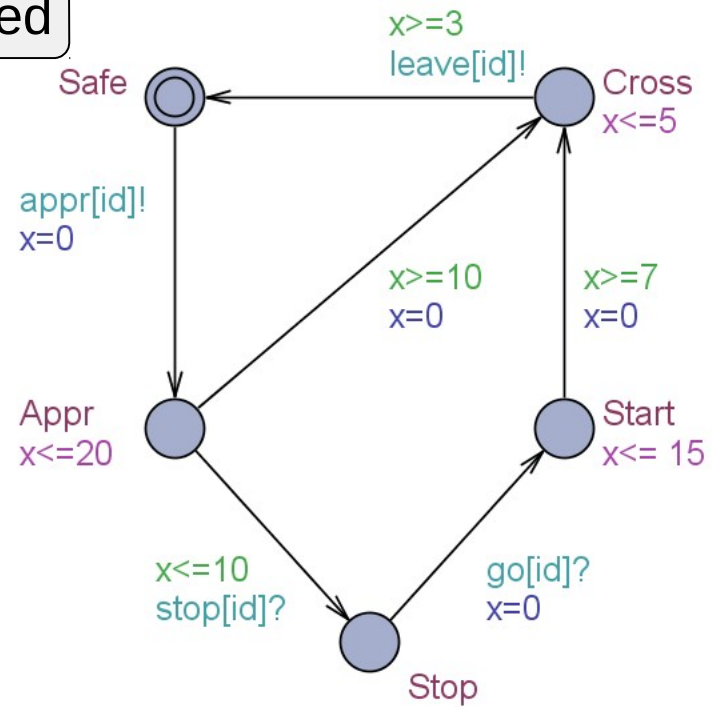
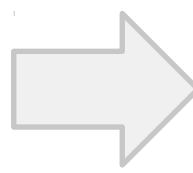
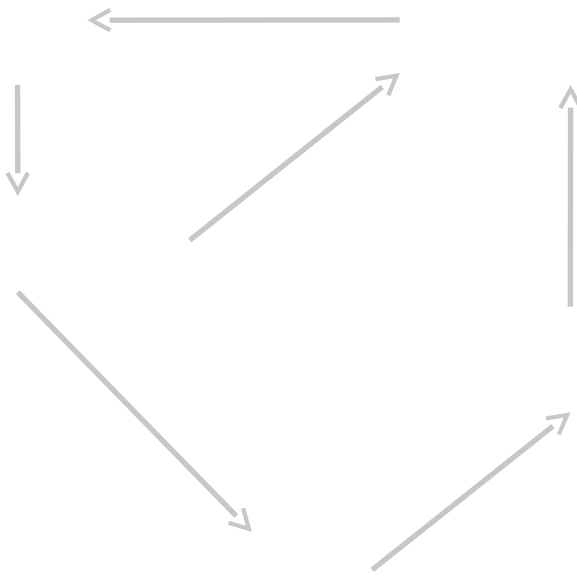
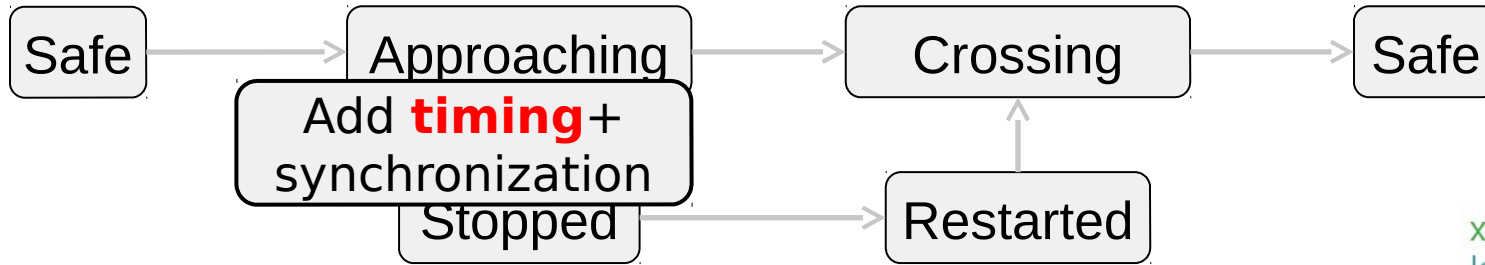


Stop the train while it still stoppable!

Train Crossing



Train Crossing



```
id_t list[N+1];
int[0,N] len;

// Put an element at the end of the queue
void enqueue(id_t element)
{
    list[len++] = element;
}

// Remove the front element of the queue
void dequeue()
{
    int i = 0;
    len -= 1;
    while (i < len)
    {
        list[i] = list[i + 1];
        i++;
    }
    list[i] = 0;
}

// Returns the front element of the queue
id_t front()
```

Language

- User defined functions (C-like)
- New types (records, type declarations, meta variables, scalars)
- Partial instantiation of templates
- Select clauses on edges
- Forall and exist quantifiers

Concrete Simulator



- ### Graphical Simulator
- visualization and recording
 - inexpensive fault detection
 - inspection of error traces
 - Message Sequence Charts
 - Gantt Charts

The screenshot displays the Concrete Simulator interface with the following components:

- Transition chooser:** A table with columns 0.0, 3.0, 6.0, 9.0, 12.0, 15.0. The selected transition is `appr[5]: Train(5) → Gate[5]`. Below it are controls for Delay (10,898) and buttons for Take transition and Reset.
- Simulation Trace:** Buttons for First, Prev, Play, Next, Last, and a Speeder slider ranging from Slow to Fast. A Random button is also present.
- Global variables:** A tree view showing variables like `t(0) = 0`, `time = 457.299845`, and an array `list = {4,1,2,0,3,0,0}`.
- State transition diagrams:** Four diagrams for Train(0), Train(1), Train(2), and Train(3). Each diagram shows states (Safe, Appr, Start, Stop, Cross) and transitions with guards like `x >= 10` and `x <= 10`.
- Gantt Chart:** A horizontal timeline from 0 to 19015 showing the execution of Train(0) through Train(5) with colored bars representing their activity.

Symbolic Simulator



Graphical Simulator

- visualization and recording
- inexpensive fault detection
- inspection of error traces
- Message Sequence Charts
- Gantt Charts

The screenshot displays the UPPAAL Symbolic Simulator interface. The main window is titled "C:\Users\kgl\Desktop\DESKTOP12\UPPAAL\UPPAAL examples\LCCC2013\SMC\TrainGa". The interface includes a menu bar (File, Edit, View, Tools, Options, Help), a toolbar, and a tabbed interface with "Editor", "Simulator", "ConcreteSimulator", "Verifier", and "Yggdrasil" tabs.

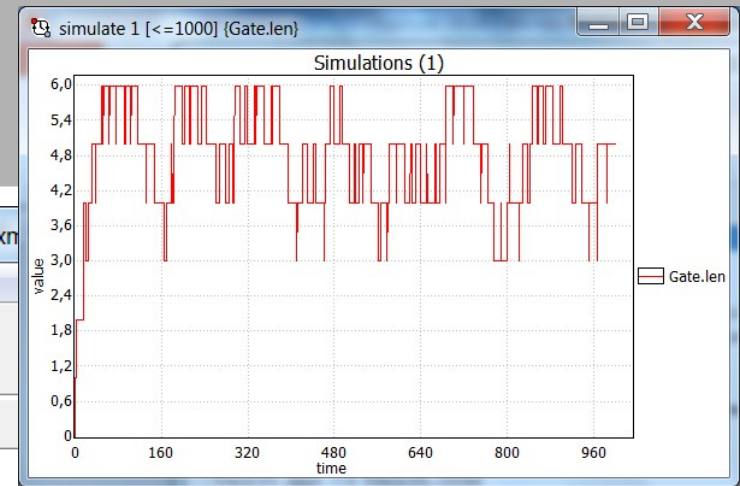
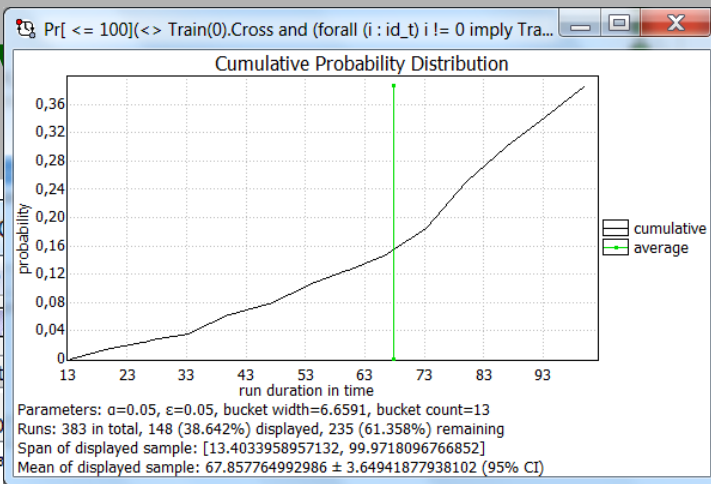
Enabled Transitions: go[front()]: Gate → Train(5). Buttons: Next, Reset.

Simulation Trace:
Train(1)
(Safe, Cross, Stop, Stop, Stop, Stop, Occ)
leave[1]: Train(1) → Gate[1]
(Safe, Safe, Stop, Stop, Stop, Stop, Free)
go[front()]: Gate → Train(5)
(Safe, Safe, Stop, Stop, Stop, Start, Occ)
appr[0]: Train(0) → Gate[0]

Gate
-list = {5,3,4,2,0,0,0}
[0] = 5
[1] = 3
[2] = 4
[3] = 2
[4] = 0
[5] = 0
[6] = 0
len = 4

Constraints:
time ≥ 63
Train(0).x ∈ [13,25]
Train(1).x ∈ [3,5]
Train(2).x ∈ [10,25]
Train(3).x ∈ [30,65]
Train(4).x ∈ [23,60]
Train(5).x ∈ [30,65]
Train(0).x - time ≤ -50
Train(0).x - Train(1).x ∈ [10,20]
Train(0).x - Train(2).x ∈ [0,5]
Train(3).x - Train(0).x ∈ [17,40]
Train(4).x - Train(0).x ∈ [10,35]
Train(2).x - Train(1).x ∈ [7,20]
Train(3).x - Train(5).x ∈ [-5,0]
Train(4).x - time ≤ -33
Train(4).x - Train(3).x ∈ [-20,0]
Train(5).x - time ≤ -30
Train(5).x - Train(0).x ∈ [17,40]
Train(5).x - Train(4).x ∈ [0,20]

Graphical Views:
- **State Transition Diagrams:** Four diagrams showing state transitions for Train(0), Train(1), Train(2), and Train(3) with variables x and transitions like stop, go, leave, and appr.
- **Message Sequence Chart (MSC):** Shows interactions between Train(0) through Train(5) and Gate. Messages include Start, Cross, Safe, Stop, Free, Occ, and go[front()].
- **Gantt Chart:** A timeline view showing the execution of Train(0) through Train(5) and Gate. Events include Start, Cross, Safe, Stop, Free, and Occ. Red arrows indicate message passing between components.



013\SMC\TrainGateCPS14.xm

```

E<> Train(0).Cross
E<> Train(1).Cross
E<> Train(0).Cross and (forall (i : id_t) i != 0 imply Train(i).Stop)
A[] forall (i : id_t) forall (j : id_t) Train(i).Cross && Train(j).Cross imply i == j
A[] Gate.list[N] == 0

Pr[ <= 100](<> Train(0).Cross and (forall (i : id_t) i != 0 imply Train(i).Stop))
E[ <= 100; 2000](max: sum(i:id_t) Train(i).Stop)
  
```

Train(0).Appr --> Train(0).Cross

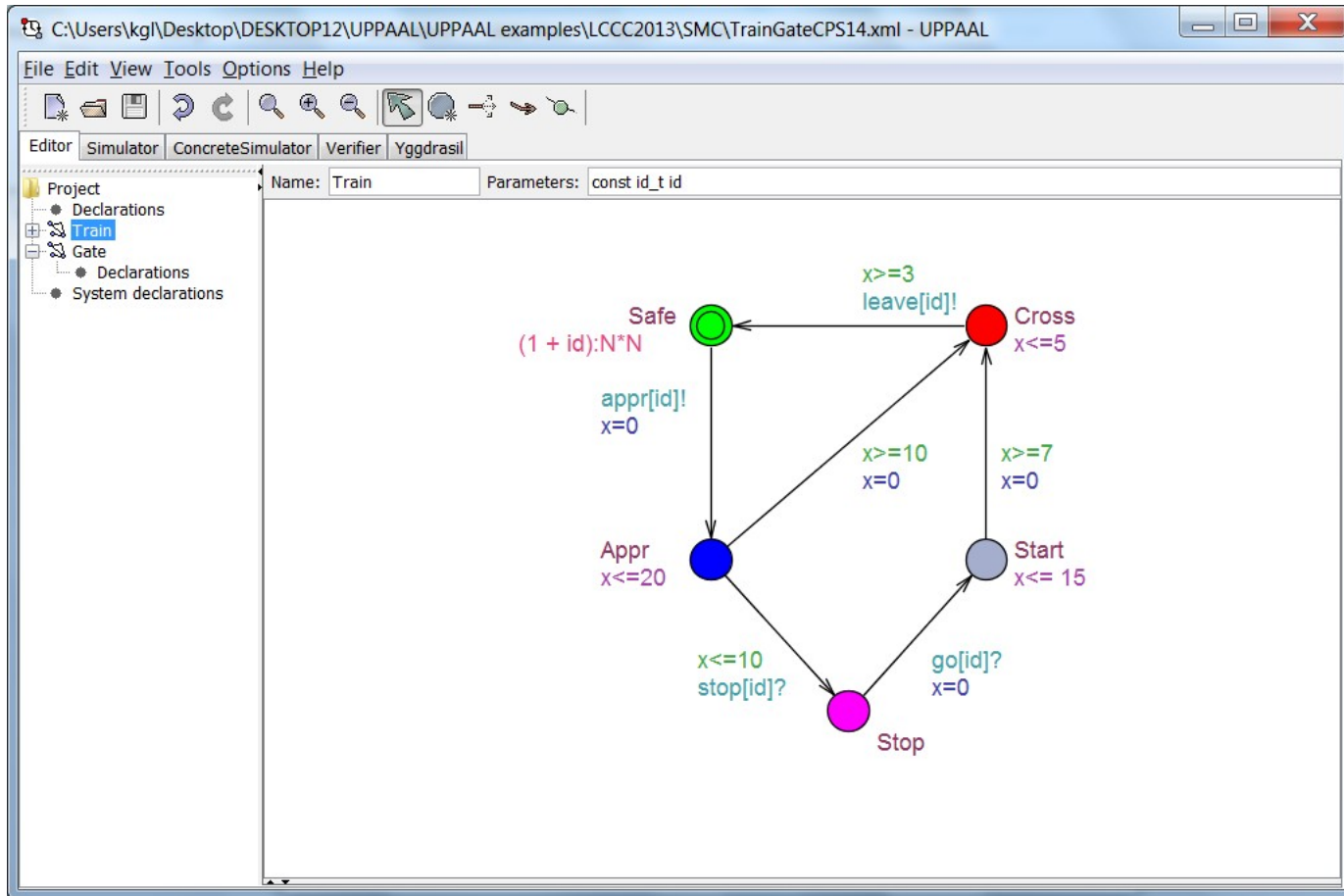
Query
Train(0).Appr --> Train(0).Cross

Comment
Whenever a train approaches the bridge, it will eventually cross.

Verifier

- Exhaustive & automatic checking of requirements
- .. including validating, safety, liveness, bounded liveness and response properties
- .. performance properties, e.g probabilistic and expectation.
- .. generation of debugging information for visualisation in simulator.
- .. plot composer

Demo

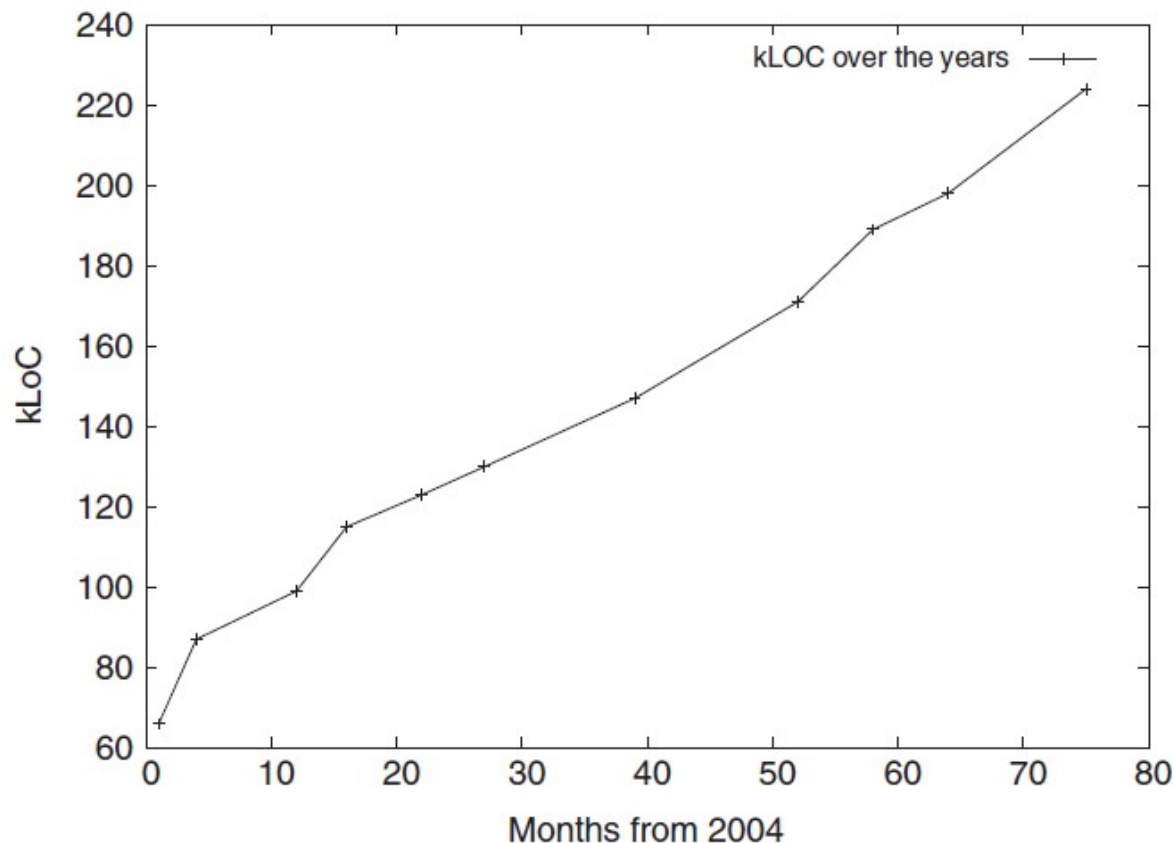


Evolution of Performance



Version	CSMA5	CSMA7	CSMA12	Fischer5	Fischer7	Fischer12	HDDI7	HDDI12
3.0.39	8.4 s 7.2 MB	— —	— —	4.2 s 10.6 MB	— —	— —	36.3 s 20.1 MB	— —
3.2.12	0.3 s 3.8 MB	417 s 145 MB	— —	1.6 s 6.8 MB	— —	— —	7.2 s 11.9 MB	— —
3.3.25	0.2 s 3.4 MB	198 s 113 MB	— —	1.1 s 6 MB	— —	— —	3.2 s 8.4 MB	— —
3.4.6	<0.1 s 3.1 MB	40.7 s 34.5 MB	— —	0.3 s 4.9 MB	4706 s 267 MB	— —	0.1 s 1.6 MB	5.3 s 14.1 MB
4.0.11	<0.1 s 1.6 MB	0.2 s 38 MB	33.8 s 115 MB	<0.1 s 1.6 MB	0.4 s 38.1 MB	418 s 300 MB	<0.1 s 1.6 MB	0.4 s 38 MB
4.1.2	<0.1 s 1.6 MB	0.2 s 21.6 MB	41.9 s 99 MB	<0.1 s 21 MB	0.3 s 21.6 MB	341 s 248 MB	0.05 s 1.6 MB	0.2 s 22.9 MB

Evolution of Code Base



Client-Server Architecture

GUI: Java

Engine: C++

Platforms:

Linux, MacOS, Solaris,
Windows

3 major cycles.

THE "secret" of UPPAAL



The screenshot displays the UPPAAL simulator interface. The main window shows a simulation trace and a state transition diagram. A central text box highlights the following simulation trace:

```
Train(2).x ∈ [23,60]
Train(5).x ∈ [30,65]
Train(0).x - time ≤ -50
Train(0).x - Train(1).x ∈ [10,20]
Train(0).x - Train(2).x ∈ [0,5]
Train(3).x - Train(0).x ∈ [17,40]
Train(4).x - Train(0).x ∈ [10,35]
Train(2).x - Train(1).x ∈ [7,20]
```

The interface includes a menu bar (File, Edit, View, Tools, Options, Help), a toolbar, and a tabbed interface with 'Editor', 'Simulator', 'ConcreteSimulator', 'Verifier', and 'Yggdrasil'. The 'Simulator' tab is active, showing the simulation trace and a state transition diagram. The trace lists events such as 'go[front(): Gate → Train(5)', 'leave[1]: Train(1) → Gate[1]', and 'appr[0]: Train(0) → Gate[0]'. The state transition diagram shows nodes for 'Cross', 'Stop', and 'Safe' with transitions labeled 'leave[1]' and 'go[front()]'. The diagram also shows a 'Gate' component with a 'list = {5,3,4,2,0,0}' and a 'Train(1)' component with a 'Cross' node and a 'Stop' node.

Zones & DBMs

THE "secret" UPPAAL



RELATED SITES: UPPAAL

UPPAAL DBM Library

The library used to manipulate DBMs in UPPAAL

[Main Page](#) | [Download](#) | [Ruby Binding](#) | [Help](#) | [Contact us](#)

Welcome!

DBMs [dill89, rokicki93, lpw:fct95, bengtsson02] are efficient data structures to represent clock constraints in timed automata [ad90]. They are used in UPPAAL [lpy97, by04, bdl04] as the core data structure to represent time. The library features all the common operations such as up (delay, or future), down (past), general updates, different extrapolation functions, etc.. on DBMs and federations. The library also supports subtractions. The API is in C and C++. The C++ part uses active clocks and hides (to some extent) memory management.

References

- [dill89] David L. Dill. *Timing Assumptions and Verification of Finite-State Concurrent Systems*. LNCS 407. Springer Berlin 1989, pp 197-212.
- [rokicki93] Tomas Gerhard Rokicki. *Representing and Modeling Digital Circuits*. Ph.D. thesis, Stanford University 1993.
- [lpw:fct95] Kim G. Larsen, Paul Pettersson, and Wang Yi. *Model-Checking for Real-Time Systems*. Fundamentals of Computation Theory 1995, LNCS 965 pages 62-88.
- [bengtsson02] Johan Bengtsson. *Clocks, DBM, and States in Timed Systems*. Ph.D. thesis, Uppsala University 2002.
- [ad90] Rajeev Alur and David L. Dill. *Automata for Modeling Real-Time Systems*. International Colloquium on Algorithms, Languages, and Programming 1990, LNCS 443 pages 322-335.
- [lpy97] Kim G. Larsen, Paul Pettersson, and Wang Yi. *UPPAAL in a Nutshell*. International Journal on Software Tools for Technology Transfer, October 1997, number 1-2 pages 134-152.
- [by04] Johan Bengtsson and Wang Yi. *Timed Automata: Semantics, Algorithms and Tools*. Concurrency and Petri Nets 2004, LNCS 3008

Latest News

Draft manual available.

23 Oct 2006

The current work-in-progress manual of the DBM library is now available for download. It is still incomplete but it may be useful so we release it.

Ruby binding version 0.7 released

30 Jun 2006

Added displaying of points in the viewer.
Re-compiled against 2.0.5.

Version 2.0.5 released

30 Jun 2006

Fixed bug in getValuation, added hooks to mingraph from the C++ API, drastically improved partition_t, improved subtractions, added a hasZero method, new print format of DBMs and federations to be more compatible with the Ruby binding, fixed doxygen

True

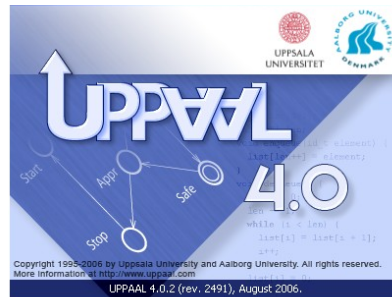
UPPAAL as a back-end



- Voodoo: verification of object-oriented designs using Uppaal, 2004.
- Moby/RT: A Tool for Specification and Verification of Real-Time Systems, 2000.
- Formalising the ARTS MPSOC Model in UPPAAL, 2007
- Marte UML ✉ UPPAAL , 2003.
- Yggdrasil: Statechart ✉ UPPAAL, 2003
- Component-Based Design and Analysis of Embedded Systems with UPPAAL PORT, 2008
- Verification of COMDES-II Systems Using UPPAAL with Model Transformation, 2008
- METAMOC: Modular WCET Analysis Using UPPAAL, 2010.
-

Industrial Usage

some examples



Bang & Olufsen (1997)



Arne Skou, Klaus Havelund



- Bug known to exist for 10 years
- Ill-described:
 - 2.800 loc +
 - 3 flowchart +
 - 1 B&O eng.
- 3 months for modeling.
- UPPAAL detects error with 1.998 transition steps (shortest)
- Error trace was confirmed in B&O laboratory.
- Error corrected and verified in UPPAAL.
- Follow-up project.



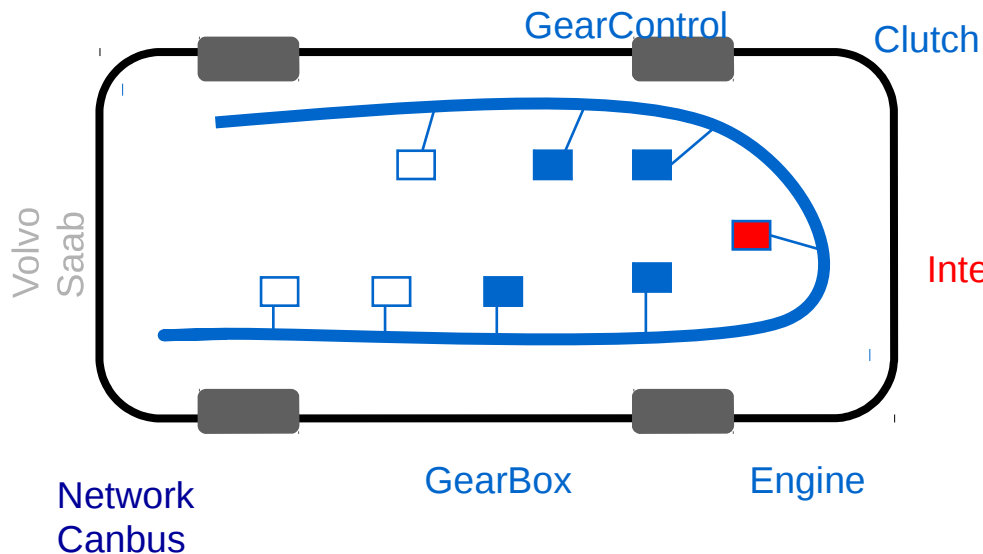
Conclusions

- It *is* possible to make a feasible abstraction of the existing system
- Bugs were found during model development and simulation
- A timing problem was identified during model checking. B&O changed their desing to remove the problem
- Time slicing and interrupt priorities can be modelled by timed automatons
- B&O obtained more confidence in the design before starting their implementation work. The design was robust in the sense that it did not have to be changed during the implementation phase

MECEL AB (1998)

Gear Controller

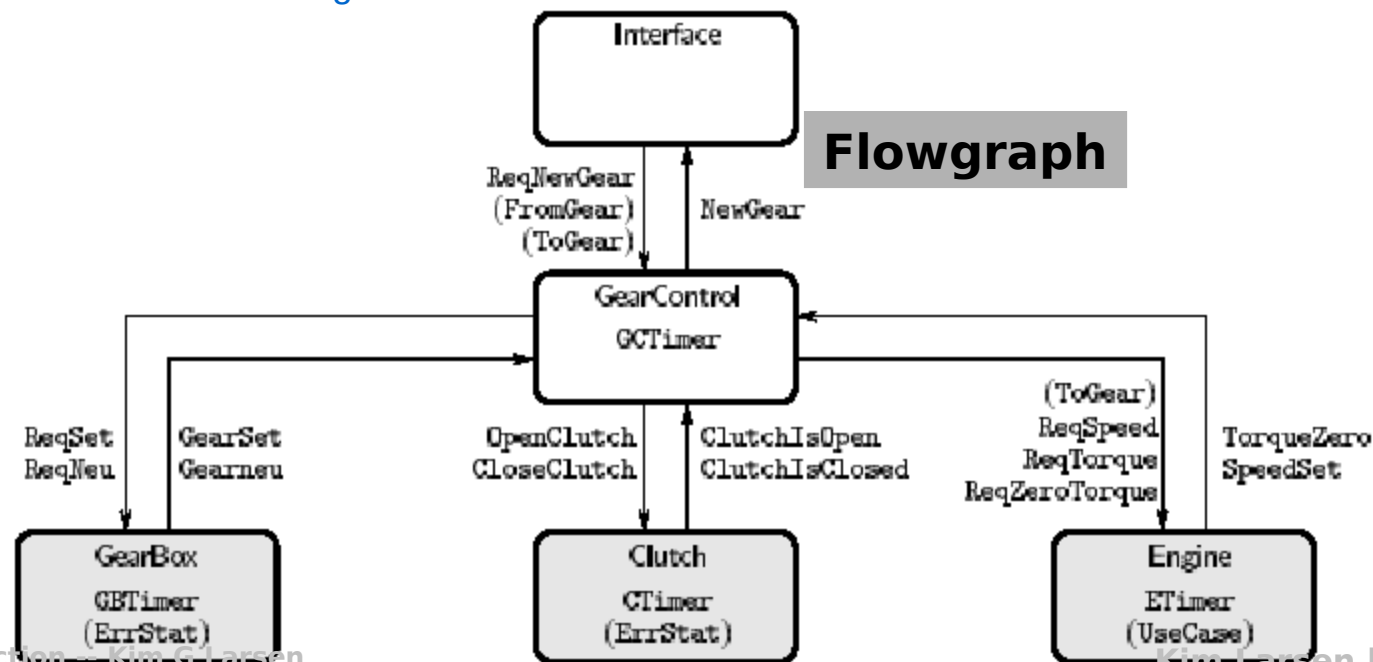
Lindahl, Pettersson, Yi 1998



Paul Pettersson

Interface

Flowgraph

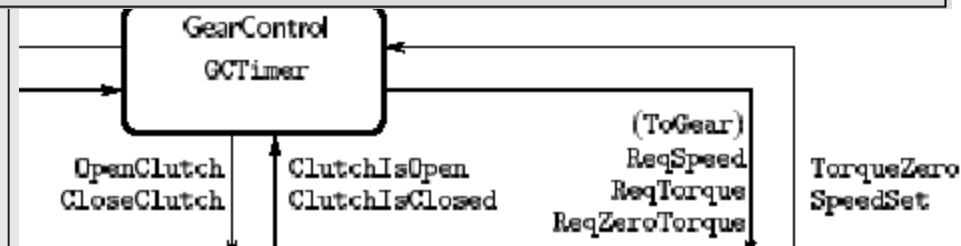
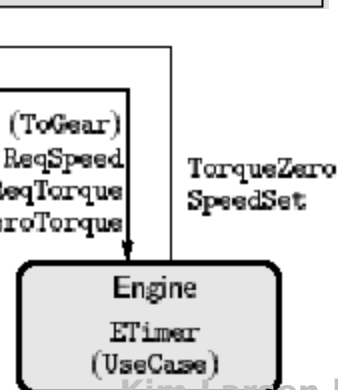
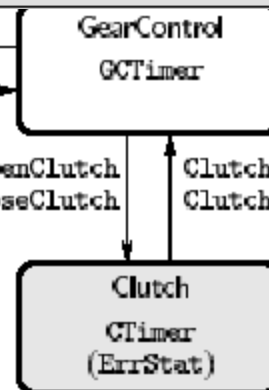
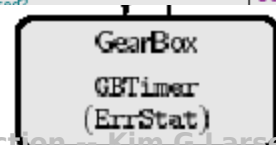
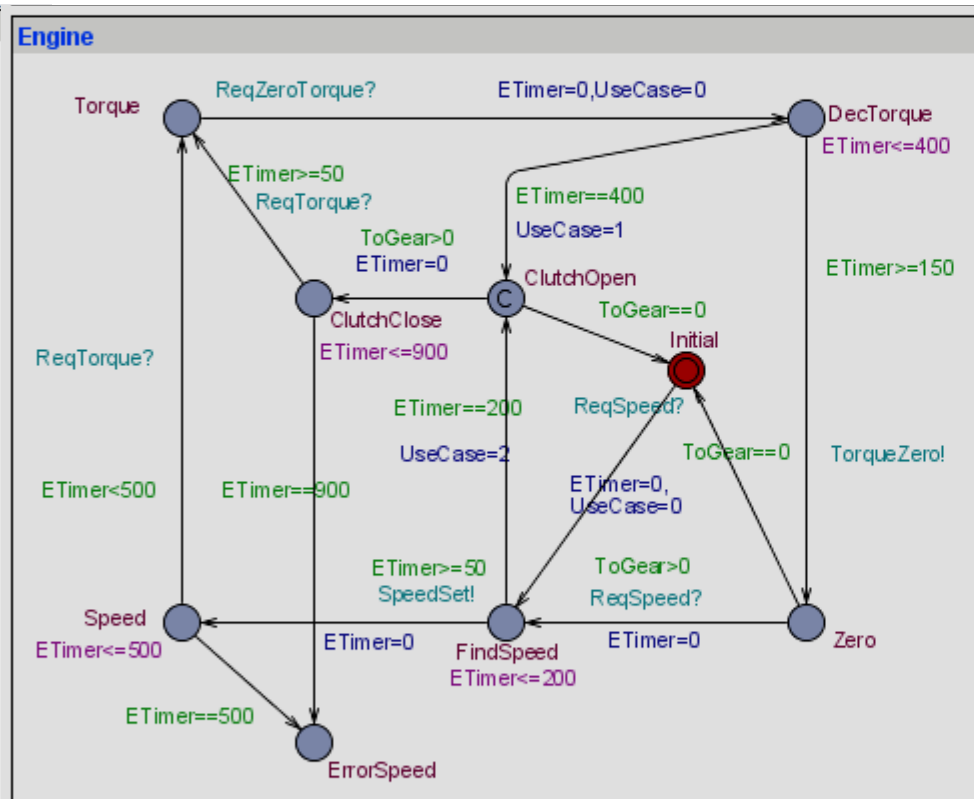
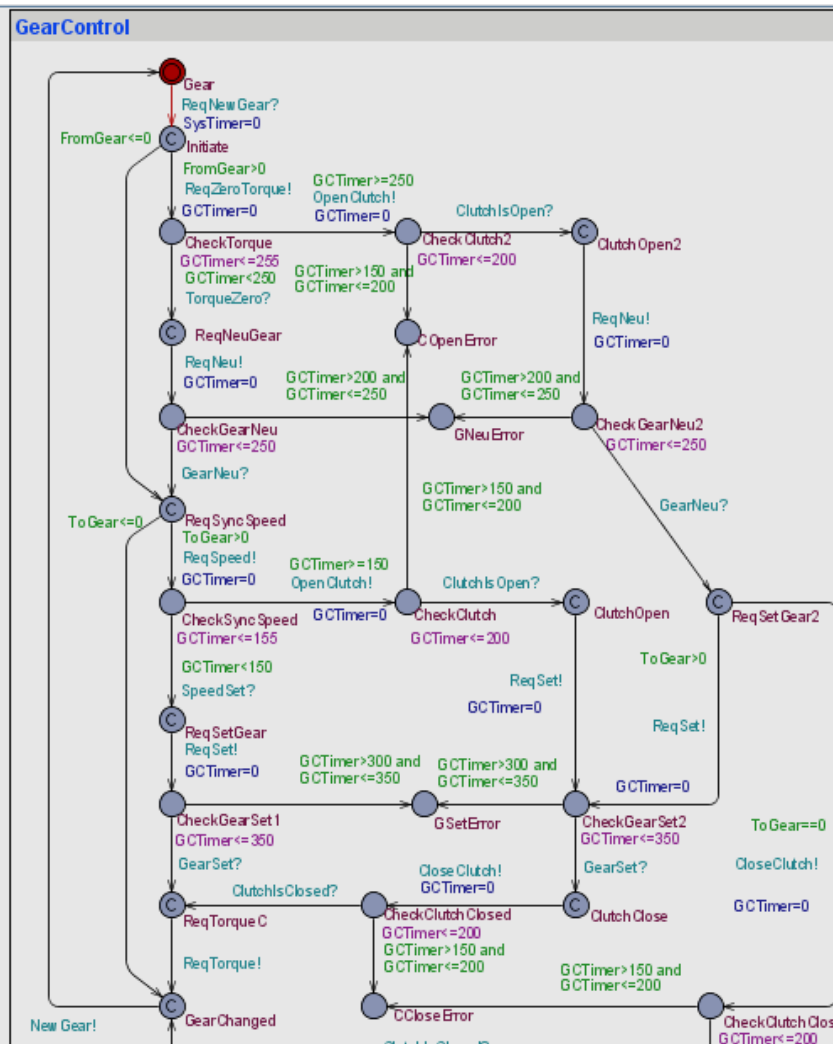


MECEL AB (1998)

Gear Controller



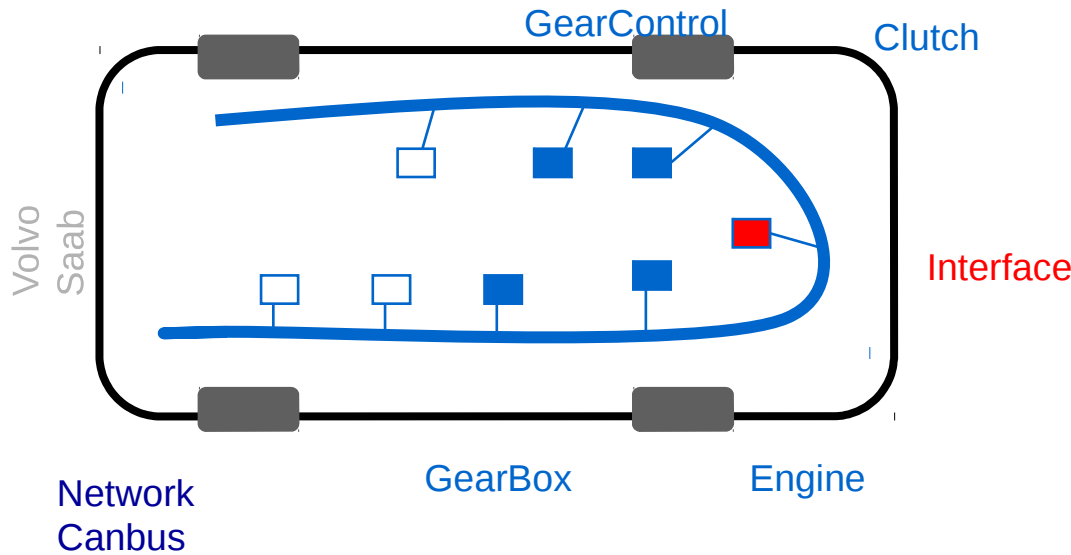
Lindah, Pettersson, Yi 1998



MECEL AB (1998)

Gear Controller

Lindahl, Pettersson, Yi 1998



Paul Pettersson

$$\text{GearControl@Initiate} \rightsquigarrow_{\leq 1500} ((\text{ErrStat} = 0) \Rightarrow \text{GearControl@GearChanged}) \quad (1)$$

$$\begin{aligned} \text{GearControl@Initiate} &\rightsquigarrow_{\leq 1000} \\ & ((\text{ErrStat} = 0 \wedge \text{UseCase} = 0) \Rightarrow \text{GearControl@GearChanged}) \end{aligned} \quad (2)$$

$$\text{Clutch@ErrorClose} \rightsquigarrow_{\leq 200} \text{GearControl@CCloseError} \quad (3)$$

$$\text{Clutch@ErrorOpen} \rightsquigarrow_{\leq 200} \text{GearControl@COpenError} \quad (4)$$

$$\text{GearBox@ErrorIdle} \rightsquigarrow_{\leq 350} \text{GearControl@GSetError} \quad (5)$$

$$\text{GearBox@ErrorNeu} \rightsquigarrow_{\leq 200} \text{GearControl@GNeuError} \quad (6)$$

$$\text{Inv} (\text{GearControl@CCloseError} \Rightarrow \text{Clutch@ErrorClose}) \quad (7)$$

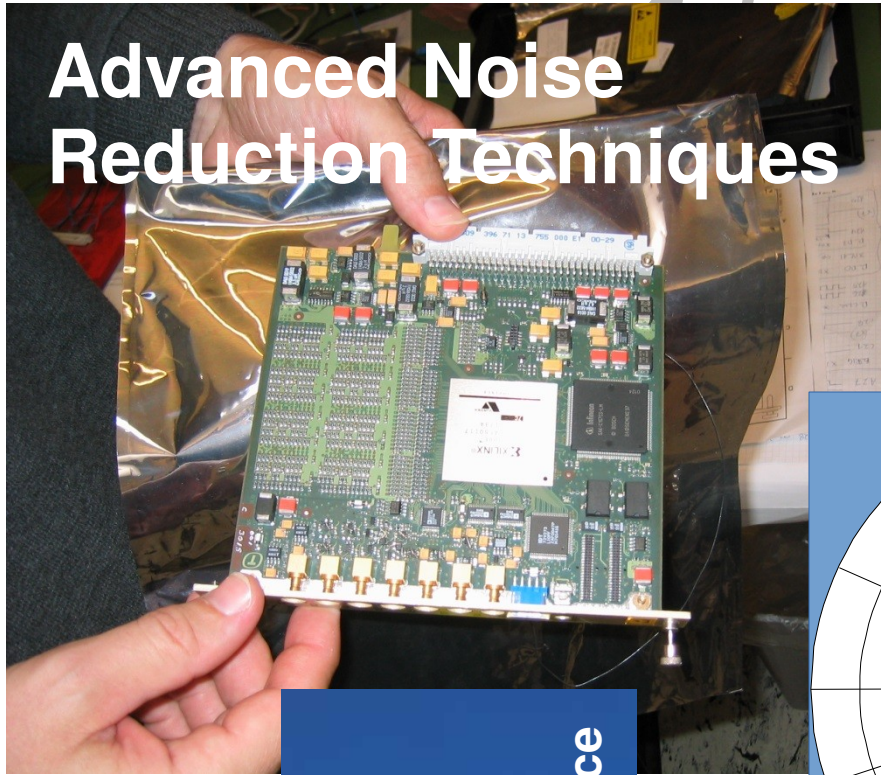
$$\text{Inv} (\text{GearControl@COpenError} \Rightarrow \text{Clutch@ErrorOpen}) \quad (8)$$

TERMA A/S (2004)

Memory Management for Radars

TERMA[®]

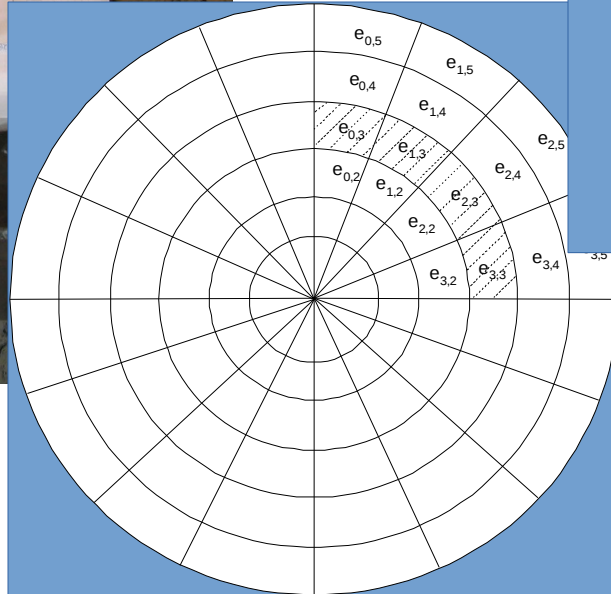
Advanced Noise Reduction Techniques



Costal Surveillance



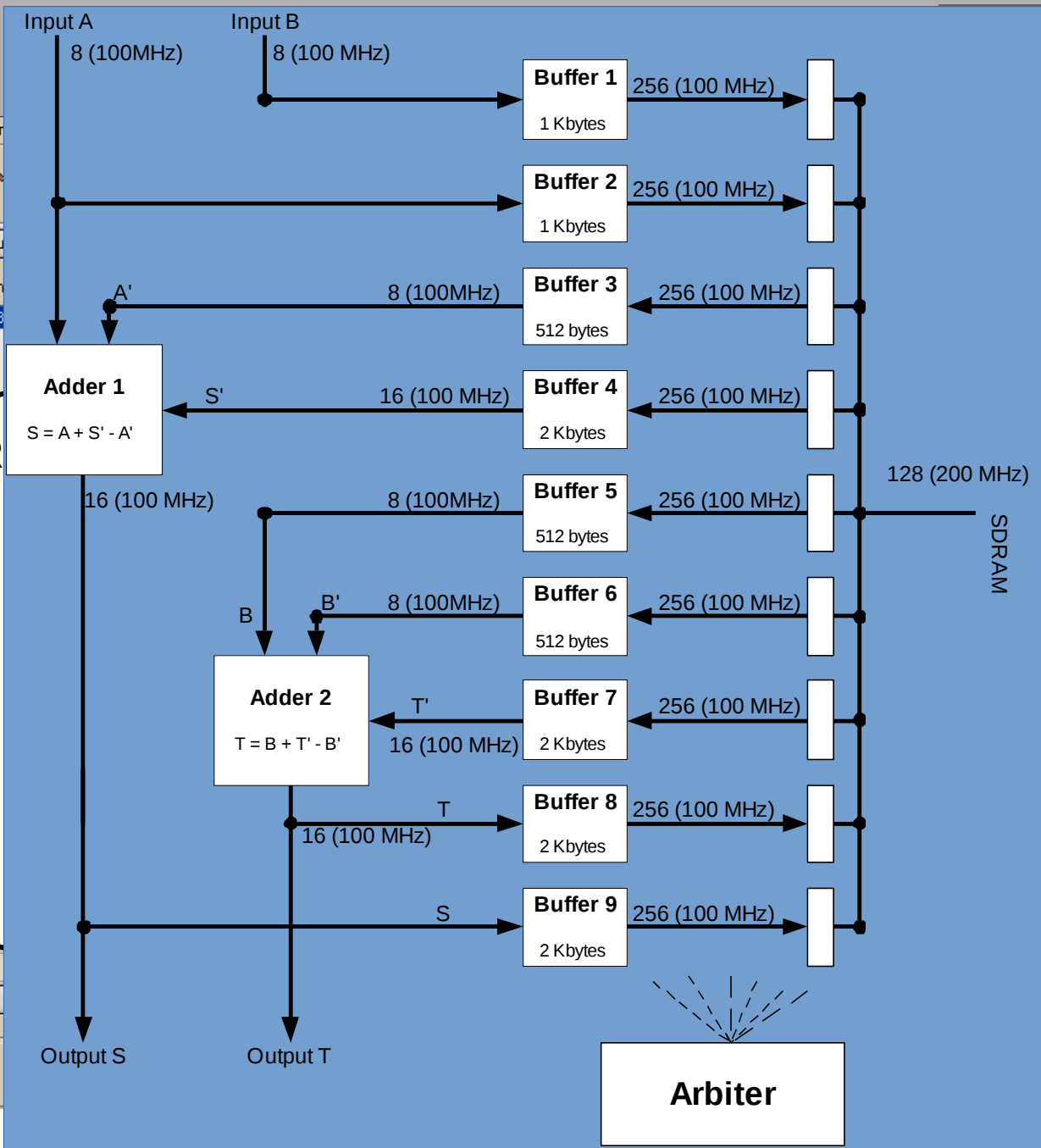
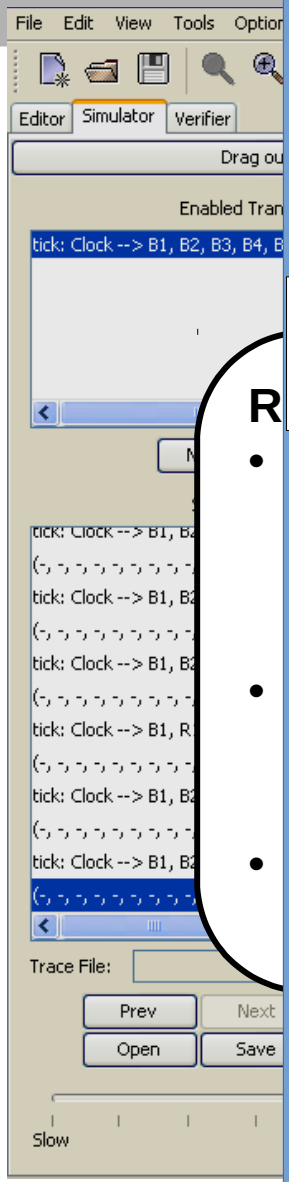
Frequency Diversity

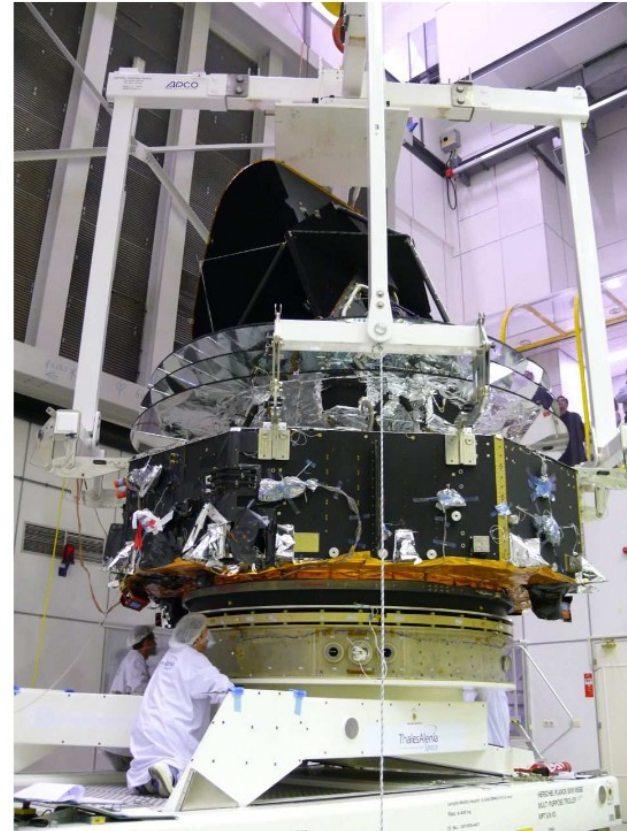


Airport Surveillance

AMETIST

advanced methods for timed systems





Attitude and Orbit Control Software
TERMA A/S Steen Ulrik Palm, Jan Storbank Pedersen, Poul Hougaard

- **Application software (ASW)**
 - built and tested by Terma:
 - does attitude and orbit control, tele-commanding, fault detection isolation and recovery.
- **Basic software (BSW)**
 - low level communication and scheduling periodic events.
- **Real-time operating system (RTEMS)**
 - Priority Ceiling for ASW,
 - Priority Inheritance for BSW
- **Hardware**
 - single processor, a few buses, sensors and actuators

Application Software (ASW)

Basic Software (BSW)

Hardware

Requirements:

Software tasks should be schedulable.

CPU utilization should not exceed 50% load

File Edit View Tools Options Help

Editor Simulator Verifier

Drag out

Transition chooser

0,0	15,0	30,0	45,0	60,0	7
-----	------	------	------	------	---

Scheduler

Delay: 13.5 Reset

Take transition

Trace controls

First 353.5 Last

Prev Play Next

Speeder

Slow Fast

Random

Simulation Trace

(-, starting, Idle, Idle, starting, starting, st
initialize: Scheduler --> Bkgnd_P, NominalE
(Running, Idle, Idle, Idle, Idle, Idle, Idle, I
enqueue: RTEMS_RTC --> Scheduler
(Schedule, Idle, Idle, Idle, Idle, Idle, Idle, I
preempt[ctask]: Scheduler --> IdleTask
(Preempt, Idle, Idle, Idle, Idle, Idle, Idle, I

Drag out

```
cycleCount = 0
ctask = 7
taskqueue[0] = 8
taskqueue[1] = 9
taskqueue[2] = 10
taskqueue[3] = 11
taskqueue[4] = 12
taskqueue[5] = 13
taskqueue[6] = 14
taskqueue[7] = 16
taskqueue[8] = 17
taskqueue[9] = 23
taskqueue[10] = 24
taskqueue[11] = 25
taskqueue[12] = 26
taskqueue[13] = 27
taskqueue[14] = 28
taskqueue[15] = 29
taskqueue[16] = 30
taskqueue[17] = 33
taskqueue[18] = 0
taskqueue[19] = 0
taskqueue[20] = 0
taskqueue[21] = 0
taskqueue[22] = 0
taskqueue[23] = 0
taskqueue[24] = 0
taskqueue[25] = 0
taskqueue[26] = 0
taskqueue[27] = 0
taskqueue[28] = 0
taskqueue[29] = 0
taskqueue[30] = 0
taskqueue[31] = 0
taskqueue[32] = 0
taskqueue[33] = 0
running[0] = 0
running[1] = 0
running[2] = 0
```

Scheduler

Bkgnd_P

secondF_2

secondF_1

UPPAAL 4.1 Framework ISoLA 2010

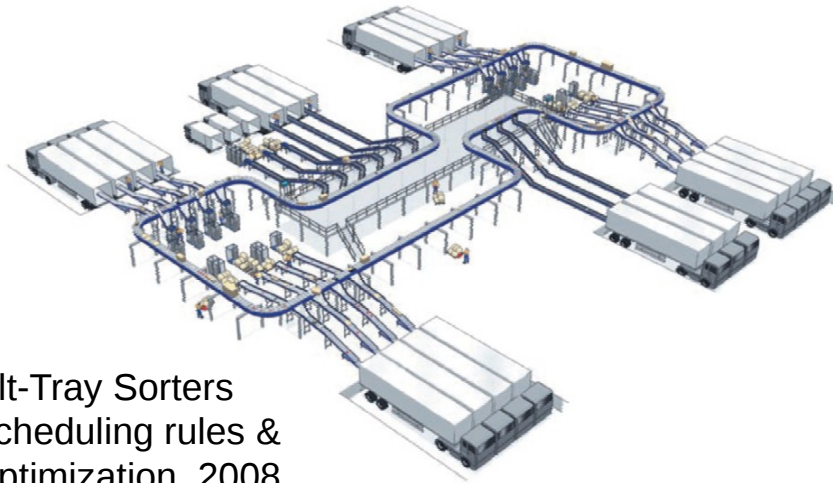
ID	Task	Specification			Blocking times			WCRT		
		Period	WCET	Deadline	Terma	UPPAAL	Diff	Terma	UPPAAL	Diff
1	RTEMS_RTC	10.000	0.013	1.000	0.035	0	0.035	0.050	0.013	0.037
2	AswSync_SyncPulseIsr	250.000	0.070	1.000	0.035	0	0.035	0.120	0.083	0.037
3	Hk_SamplerIsr	125.000	0.070	1.000	0.035	0	0.035	0.120	0.070	0.050
4	SwCyc_CycStartIsr	250.000	0.200	1.000	0.035	0	0.035	0.320	0.103	0.217
5	SwCyc_CycEndIsr	250.000	0.100	1.000	0.035	0	0.035	0.220	0.113	0.107
6	Rt1553_Isr	15.625	0.070	1.000	0.035	0	0.035	0.290	0.173	0.117
7	Bc1553_Isr	20.000	0.070	1.000	0.035	0	0.035	0.360	0.243	0.117
8	Spw_Isr	39.000	0.070	2.000	0.035	0	0.035	0.430	0.313	0.117
9	Obdh_Isr	250.000	0.070	2.000	0.035	0	0.035	0.500	0.383	0.117
10	RtSdb_P_1	15.625	0.150	15.625	3.650	0	3.650	4.330	0.533	3.797
11	RtSdb_P_2	125.000	0.400	15.625	3.650	0	3.650	4.870	0.933	3.937
12	RtSdb_P_3	250.000	0.170	15.625	3.650	0	3.650	5.110	1.103	4.007
14	FdirEvents	250.000	5.000	230.220	0.720	0	0.720	7.180	5.153	2.027
15	NominalEvents_1	250.000	0.720	230.220	0.720	0	0.720	7.900	5.873	2.027
16	MainCycle	250.000	0.400	230.220	0.720	0	0.720	8.370	6.273	2.097
17	HkSampler_P_2	125.000	0.500	62.500	3.650	0	3.650	11.960	5.380	6.580
18	HkSampler_P_1	250.000	6.000	62.500	3.650	0	3.650	18.460	11.615	6.845
19	Acb_P	250.000	6.000	50.000	3.650	0	3.650	24.680	6.473	18.207
20	IoCyc_P	250.000	3.000	50.000	3.650	0	3.650	27.820	9.473	18.347
21	PrimaryF	250.000	34.050	59.600	5.770	0.966	4.804	65.470	54.115	11.355
22	RCSControlF	250.000	4.070	239.600	12.120	0	12.120	76.040	53.994	22.046
23	Obt_P	1000.000	1.100	100.000	9.630	0	9.630	74.720	2.503	72.217
24	Hk_P	250.000	2.750	250.000	1.035	0	1.035	6.800	4.953	1.847
25	StsMon_P	250.000	3.300	125.000	16.070	0.822	15.248	85.050	17.863	67.187
26	TmGen_P	250.000	4.860	250.000	4.260	0	4.260	77.650	9.813	67.837
27	Sgm_P	250.000	4.020	250.000	1.040	0	1.040	18.680	14.796	3.884
28	TcRouter_P	250.000	0.500	250.000	1.035	0	1.035	19.310	11.896	7.414
29	Cmd_P	250.000	14.000	250.000	26.110	1.262	24.848	114.920	94.346	20.574
30	NominalEvents_2	250.000	1.780	230.220	12.480	0	12.480	102.760	65.177	37.583
31	SecondaryF_1	250.000	20.960	189.600	27.650	0	27.650	141.550	110.666	30.884
32	SecondaryF_2	250.000	39.690	230.220	48.450	0	48.450	204.050	154.556	49.494
33	Bkgnd_P	250.000	0.200	250.000	0.000	0	0.000	154.090	15.046	139.044



Marius Micusionis

CONCLUSION

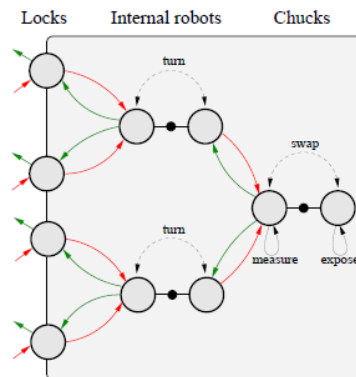
- Schedulability framework made available in UPPAAL
- Provides more exact analysis than classical methods
 - Depending on WCET information the task set is schedulable or not.
- Performance:
 - 1-2 minutes: $BCET=WCET$ or $BCET/WCET < 0.5$
 - 1 day: $0.5 < BCET/WCET < 0.8$
- Work on domain specific notation in order to be fully taken up by company.



Tilt-Tray Sorters
Scheduling rules &
Optimization, 2008



Océ Datapath, 2012



ASML, 2004:
Wafer Scanners
Optimization of
Throughput

Philips: Indoor Lighting systems, 2014

Within the Prisma project of TNO-ESI and Philips Lighting, research is done into the robustness and reliability of large-scale indoor lighting systems. The focus is on the robustness of the lighting control system. To analyse control system robustness, model checking is used. Timed automata models of lighting control systems have been created and checked with the model checker Uppaal. To validate

UPPAAL Outside Europe



Communication Art Technology Systems

HOME | 会社地図 | サイトマップ | English

製品・サービス | サポート | イベント・セミナー | 企業情報 | 採用情報 | お問い合わせ

すべての製品・サービス

ソフトウェア関連製品

- ZIPC
- ZIPC++
- ZIPC SPLM
- ZIPC Feature
- ZIPC AUTOSAR
- Perfect Pass
- Drawial
- ASN1 Tool
- XModelink
- EFMツール
- 教育支援サービス
- UPPAAL
- 代理店製品
- 取扱代理店製品一覧
- 要件・構成管理ツール

HOME > 製品・サービス > UPPAAL

リアルタイムシステムの安心・安全設計への処方箋!

UPPAAL日本国内で発売開始!

いまや、「リアルタイムシステムの安全性」は開発者だけでなく、高度に電子化されつつある現代社会のもっとも高い関心事のひとつです。しかし、リアルタイムでかつ複数のプロセスが同時に動作するようなシステムの安全性を確保するのは難しく、通常の方法(テスト)では非常にコストがかかります。このツールは、「モデル検査」という新しい技術によって、その様な安心・安全なリアルタイム・システムの開発を支援するビジュアルな統合モデル検証ツールです。

Train(0), Train(1), Gate

C:\Documents and Settings\Agj\Desktop\DESKTOP FEB 2007\UPPAAL\Moredemo\engine-class-xta - UPPAAL

File Edit View Tools Options Help

Simulator

北京奥亚锐通科技有限公司
BEIJING AOYARUITONGKEJI CO.,LTD

网站首页 | 关于我们 | 新闻中心 | 产品展示 | 下载中心 | 联系我们

系統验证解决方案

公司简介 - COMPANY INTRODUCTION

公司新闻 - NEWS

北京奥亚锐通科技有限公司是一家为客户提供专业应用软件和技术方案的高科技公司。

公司总部在北京，在成都设有研发中心，配备有20人的技术研发团队和高性能开发实验室。

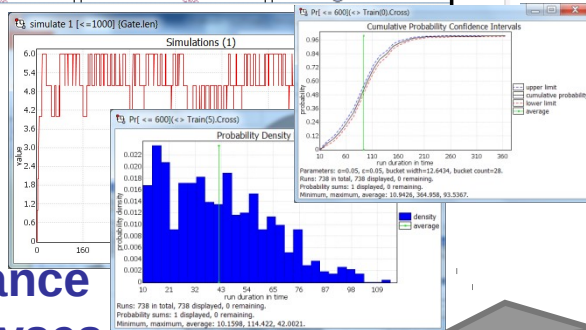
公司致力于在系统设计仿真、电子设计仿真、软件工程.....

2012-3-13 北京奥亚锐通科技有限公司网站正式上线

2012-4-13 北京奥亚锐通邮件系统正式上线...

2012-4-13 北京奥亚锐通办公平台正式上线...

Performance Analyses



UPPAAL 4.0

```

A[] not ( GearBox.Neutral and ( Interface.Gear1 or I...
A[] not ( GearBox.Idle and Interface.Gear1
Query
E<=> GearControl.GearChanged
Comment
P1. It is possible to change gear.
Status
Property is satisfied.
A[] ( Clutch.Closed imply ( GearControl.ReqTorqueC or GearControl.GearChanged or GearControl.Gear or GearC...
Property is satisfied.
A[] ( GearBox.Idle imply ( GearControl.ClutchClose or GearControl.CheckClutchClosed or GearControl.CClose
Property is satisfied.
A[] ( GearBox.Neutral imply ( GearControl.ReqSetGear or GearControl.ChClchClosed or GearControl.C
Property is satisfied.
    
```

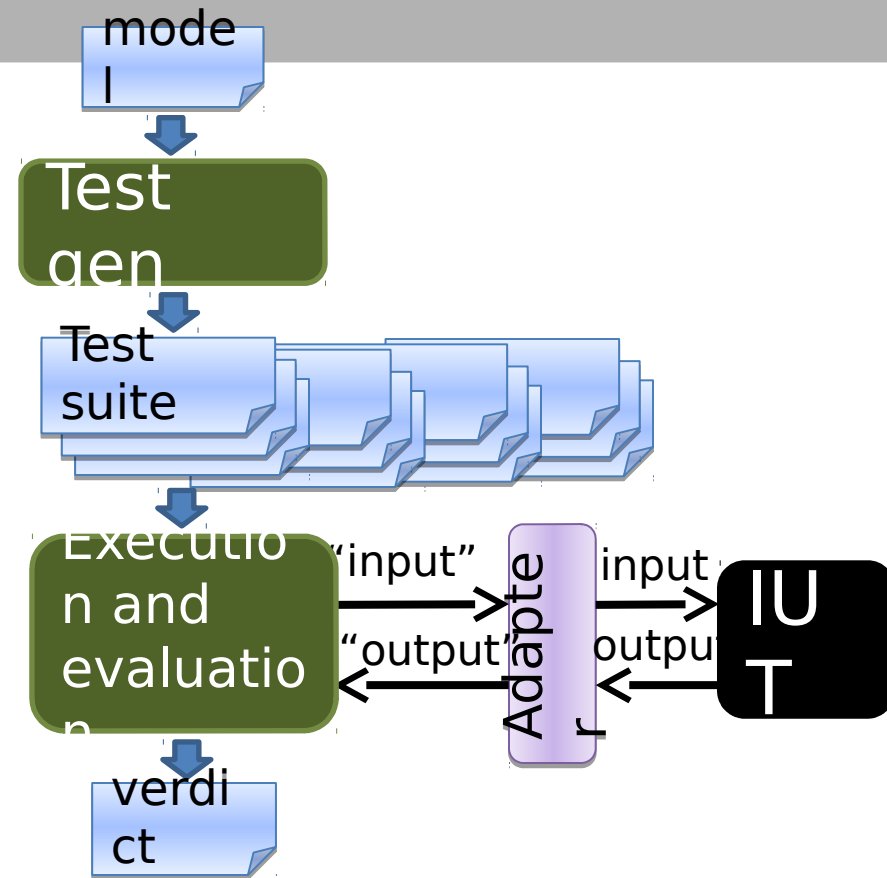
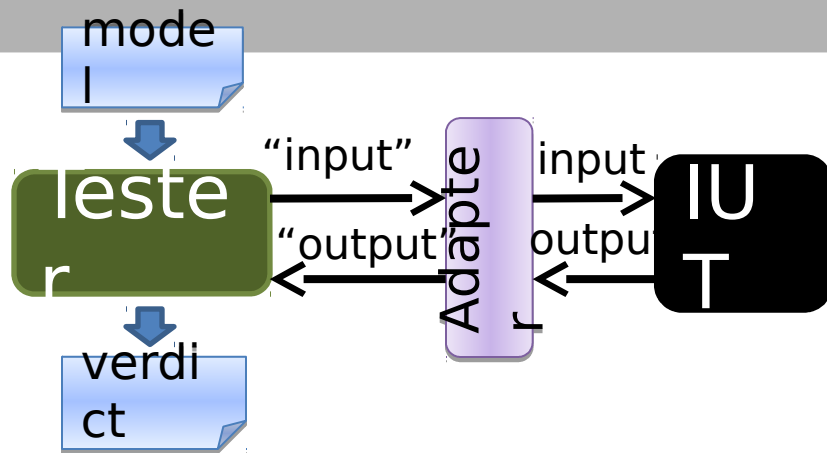
Verifier

Testing

TRON & YGGDRASIL



Online vs. Offline



Online testing:

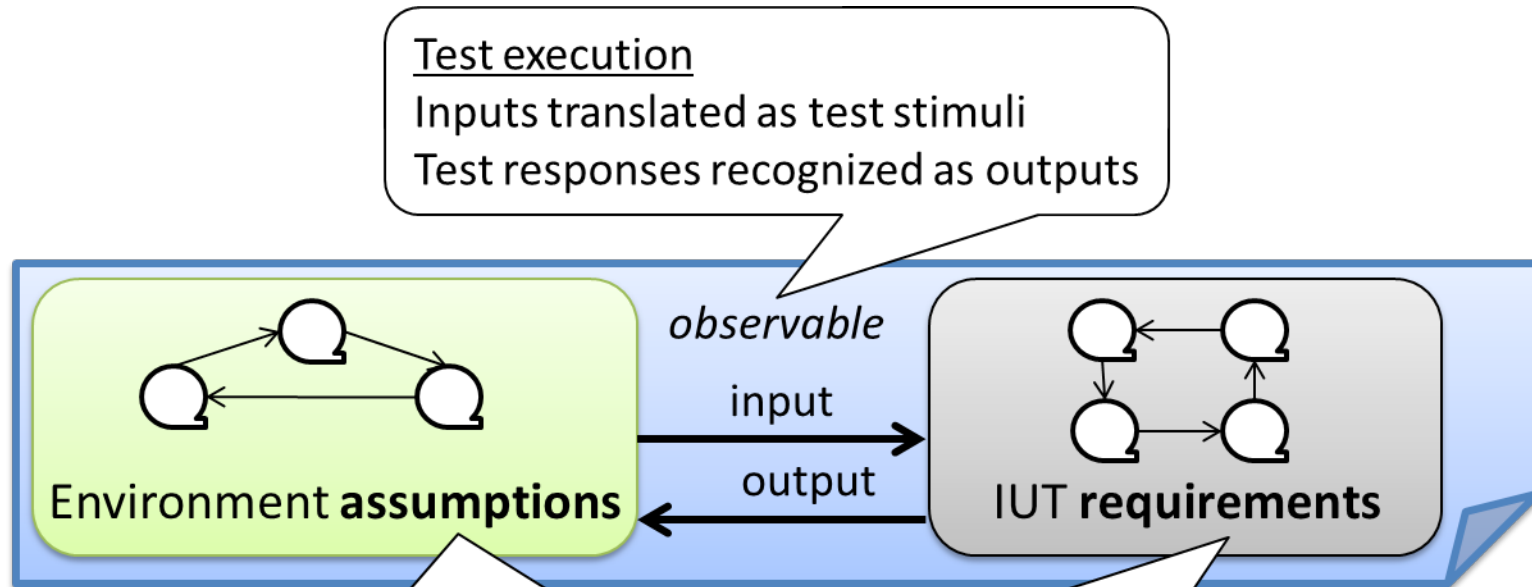
- Pros:**
- Abstract system-level behavior
 - Realistic setup, many components
 - Adaptive, explores only relevant states
 - Allows concurrency, non-determinism
 - Long and intricate interactions
 - Automatic check against model

- Cons:**
- Does not guarantee coverage
 - Interpreting model can be slow
 - Can be difficult to replicate
 - Does not replace offline testing

Offline testing:

- Real-time systems are inherently non-deterministic
- Non-determinism yields exponentially large test spaces
- Few or no concurrent components
- Short and specific interactions
- Evaluation requires careful assertion programming

Model Interpretation



Test generation

How the **tester** should behave:

- Anything is possible (stress testing)
- Emulate physical processes
- Specific use-case scenario
- Replay a previous test trace (regression)

Test evaluation/monitoring

How the **IUT** should behave:

- Consume any input at anytime
- Produce outputs expected by model
- Neither too late nor too early
- Non-deterministic:
 - multiple outcomes
 - imprecision of timing
 - concurrency

Yggdrasil (offline)

MBAT Daimler Case (2014)



The screenshot shows the Yggdrasil software interface. The title bar indicates the file path: C:\Users\kgi\Desktop\DESKTOP12\UPPAAL\UPPAAL examples\FM Forum2014\TModelRequirement0829.xml - UPPAAL. The menu bar includes File, Edit, View, Tools, Options, and Help. The toolbar contains various icons for file operations and simulation. The main window is divided into several sections:

- Options:** Includes checkboxes for "Query file", "Depth search", and "Single step". A text field contains the value "200". There are "Generate" and "Output" buttons.
- Traces:** A list of trace coverage results, including "Query" and "Depth" sections, and a final line "Total Coverage: 84/84".
- Trace statistics:** A list of statistics for various components, such as "Locations: 48/48 = 100%", "Environment.wait: 18", and "StatusCar.L0: 33".
- Outout folder:** A text field containing the path "C:\Users\kgi\Desktop\DESKTOP12\UPPAAL\uppaal-4.1.20-beta2\testcases" and a "Browse" button.

Test Code & Output



The image shows a screenshot of a model checker interface. In the foreground, an "Edit Edge" dialog box is open, with the "Test Code" tab selected. The code entered in the dialog is `turnLightOff();`. The dialog has "OK" and "Cancel" buttons. In the background, a window titled "C:\Users\kg\Desktop\DESKTOP12\UPPAAL\UPPAAL exam" is visible, showing a project tree with "Lamp" selected. A diagram element is partially visible, showing a light bulb labeled "off" and "light TIM". To the right, a large text box contains the following code snippets:

```
DrumScaleOneWeighing_IsInTopPos = 2;  
DrumScaleOneWeighing_StartSTM( &me->itsDrumScaleOneWeighing );  
HouseSettings_SetRamChar( &me->itsHouseSettings,  
HS_CURRENT_ACTIVE_SILO, (UCHAR)0);  
me->itsDrumScaleWeighingStable.IsReady_Return = 1;  
me->itsDrumScaleRollDrum.IsReady_Return = 1;  
Test_Validate( &me->itsTest, "DrumScaleOneWeighing_running",  
(UCHAR)IS_IN( &me->itsDrumScaleOneWeighing,  
DrumScaleOneWeighing_running ) );  
Test_Validate( &me->itsTest,  
"DrumScaleOneWeighing_StartOneWeighingState", (UCHAR)IS_IN(  
&me->itsDrumScaleOneWeighing,  
DrumScaleOneWeighing_StartOneWeighingState ) );  
PrintCurrentState(me);  
DrumScaleOneWeighing_StartOneWeighing(&me->  
itsDrumScaleOneWeighing, (FLOAT32)12.0,(void (*)(void * const,  
FLOAT32))OneWeighingFinishedCb, (UCHAR (*)(void * const))  
OkToRollDrumCb, me);  
Test_Validate( &me->itsTest,  
"DrumScaleOneWeighing_WeighingDrumEmpty", (UCHAR)IS_IN( &me->  
itsDrumScaleOneWeighing,  
DrumScaleOneWeighing_WeighingDrumEmpty ) );  
PrintCurrentState(me);  
Test_Comment( &me->itsTest, "DrumScaleRollDrum_IsInTopPos is %  
id", (UCHAR)DrumScaleRollDrum_IsInTopPos(&me->  
itsDrumScaleRollDrum) );  
me->itsDrumScaleWeighingStable.StableWeighingFinishedCb(me->  
itsDrumScaleWeighingStable.owner, 0.6F, 9.62F );
```

Yggdrasil Industrial Use



- Novo Nordisk
 - Reduction in time for testing a module 30 days
30 days ✉ 2 days
- Skov A/S
- TK Validate
 - Ambitious business plan
- Evaluation at
 - Daimler
 - Infineon Austria
 - EADS
 - Bombardier
 - Cov. Inc 40%
 - Reduced test time
20% (80% for unit test)

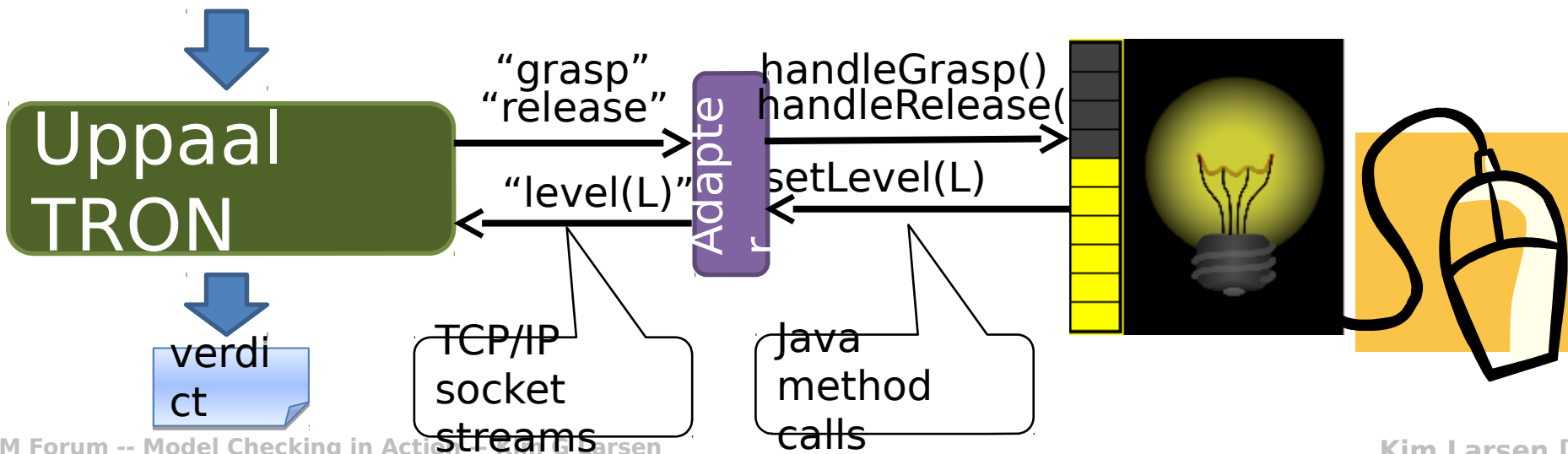
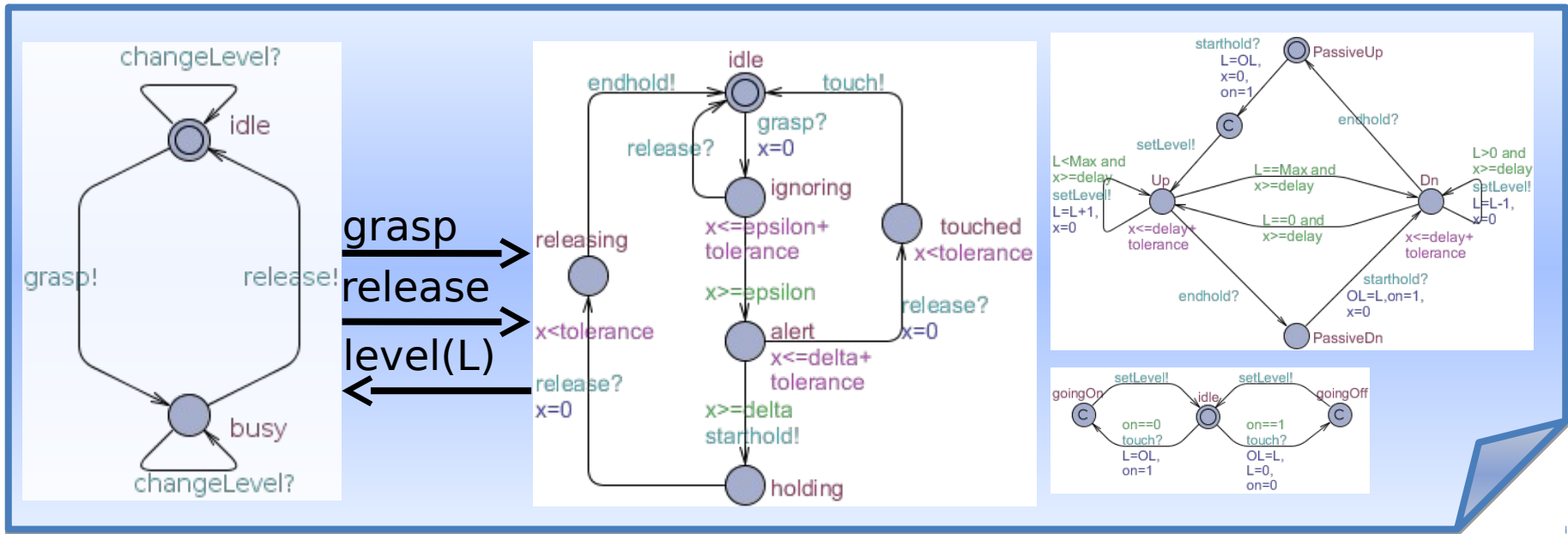


DAIMLER



BOMBARDIER

TRON (online)



TRON GUI



File Edit View Tools Options Help

Editor Simulator Verifier Tron

Partition Settings Test

Processes	Sends on	Receives On	Controls
levelAdapter graspAdapter releaseAdapter switcher interface dimmer user	grasp release	level	add to Input add to Output Complete Clear Save Load


```

    graph LR
      Env[Environment user] -- grasp() --> Imp[Implementation]
      Env -- release() --> Imp
      Imp -- level(envLevel) --> Env
  
```

File Edit View Tools Options Help

Editor Simulator Verifier Tron

Partition Settings Test

Timing Options

1 model time unit (mtu): 10,000 μs

Test duration: 0 mtu

Precomputation: 100 mtu

Input latency: 0 μs

Use clock: Real Clock Virtual Clock

Virtual Clock Port: 6,521

Delay Strategy

Eager Lazy Random

Bounded by: Short 10 long 200

Engine options

Show progress indicator

Test events applied in the Uppaal engine

Available input and delay choices for simulation

Backup state set and prepare for final diagnostics

Dump current state set on each update

Adapter Options

Adapter: SocketAdapter

Adapter Location: built in

Adapter options: localhost 9999

File Edit View Tools Options Help

Editor Simulator Verifier Tron

Partition Settings Test

Start Test in progress

mtu	Environment	Implementation
3,223,750,000		level(1)
3,224,000,000		
3,224,250,000		
3,224,500,000		
3,224,750,000	release()	level(2)
3,225,000,000		
3,225,250,000	release()	
3,225,500,000		
3,225,750,000		

Output

```

Created clock on port 6521
UPPAAL TRON 1.5 using UPPAAL 4.1.3 (rev. 4551), June 2010
Compiled with g++-4.4.4 -Wall -DNDEBUG -O2 -float-store -march=prescott
-DENABLE_CSS_MERGE -DBOOST_DISABLE_THREADS
Copyright (c) 1995 - 2010, Uppsala University and Aalborg University.
All rights reserved.
Options for UPPAAL TRON:
Search order is breadth first
  
```

Error

```

socket connect:
Connection refused
(* 9 tries left *)
  
```

File Edit View Tools Options Help

Editor Simulator Verifier Tron

Partition Settings Test

Start Test passed

mtu	Environment	Implementation
999,057.0		level(4)
999,157.0		level(3)
999,257.0		level(2)
999,357.0		level(1)
999,457.0		level(0)
999,557.0		level(1)
999,657.0		level(2)
999,757.0		level(3)
999,857.0		level(4)
999,957.0		level(5)
999,986.9	release()	
999,988.8	grasp()	
999,990.3	release()	

Output

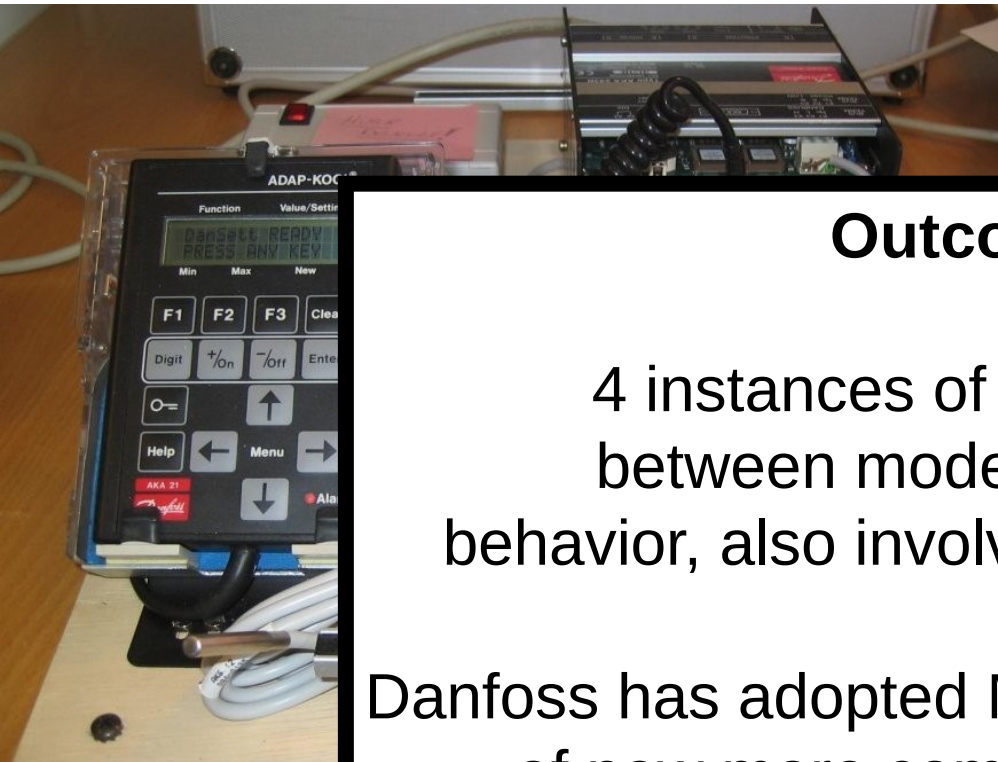
```

Input delay exceeded by: 0
OS scheduler: non-real-time.
Emulation invariants: user.
Timeunit: 10000us
Timeout: 1000000mtu
Inputs: grasp(), release()
Outputs: level(envLevel)
TEST PASSED: Time out for testing
  
```

Error

```

socket connect:
Connection refused
(* 9 tries left *)
CONN::readchannel:
Connection reset by peer
  
```

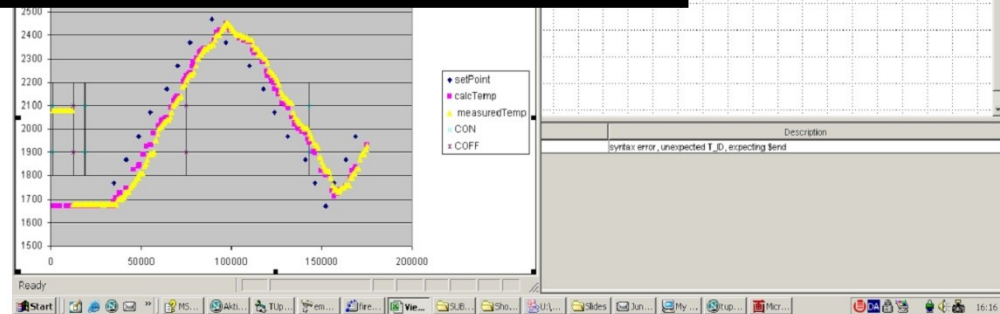


Outcome

4 instances of discrepancy between model and actual behavior, also involving timing errors.

Danfoss has adopted MBT in development of new more complex controller!

- Sequanto SeqZap test harness
- Programmable Logic Controllers (PLC)



- Engineer focus on **what** to test at a high level of abstraction
- Avoids cost of making scripts
 - As much test code as production code
 - Maintenance nightmare
- Heard of, but is still considered an **advanced technique** by industry
- Industry is very motivated, MB A&T will give
 - **10% cost reduction**
 - **20% quality improvement**

Model Checking & Testing



Model Checking

- Abstract models
- Exhaustive “proof”
- Many mature tools
- Early detection of errors
- State space expl

Testing

- Checks the actual implementation
- Only few executions checked
- But is the most direct method

How to effectively ***combine*** the different model checking and testing techniques?



DESIGN VERIFICATION FOR EMBEDDED SYSTEMS

Our world-leading and internationally acclaimed model-checking tool UPPAAL is now available for commercial use!

[HOME](#) [PRODUCT](#) [SOLUTIONS](#) [PARTNERS](#) [SUPPORT](#) [WEB HELP](#) [CONTACT US](#) [ABOUT UP4ALL](#)

DOWNLOAD & BUY UPPAAL SOFTWARE 	NON PROFIT USER? 	NEWS & EVENTS Feb. 15, 2011 - UP4ALL in OEM agreement with Elivor OU.	SUCCESSFUL USECASES See how UPPAAL is used to verify industrial systems.
---	-----------------------------	---	--

P.O. Box 337, SE-75105 Uppsala, Sweden | +46 21 151741 | sales@uppaal.com | login

Copyright © 2009 Up4All International AB. All rights reserved