Feladatok együttműködésének ellenőrzése

dr. Vörös András
https://inf.mit.bme.hu/members/vorosa
dr. Micskei Zoltán
https://inf.mit.bme.hu/members/micskeiz

Méréstechnika és Információs Rendszerek Tanszék

- 2003. augusztus 14:
  áramszünet Észak-Amerikában
  - **8 állam**, kb. **45 millió** ember maradt áram nélkül
  - a teljes áramellátó rendszer percek alatt összeomlott
  - a helyreállítás több mint **egy hétig** tartott
  - 265 erőmű esett ki, 61,8 GW összteljesítményben (≈125 paksi reaktorblokk teljesítménye)
  - leállt a közlekedés, az ivóvíz-ellátás, az olajfinomítók, a gyárak, fennakadások voltak a kommunikációban
  - felbecsülhetetlen anyagi kár, számos halálos áldozat

Forrás: http://en.wikipedia.org/wiki/Northeast_Blackout_of_2003

A hiba oka

- kiesett az Eastlake erőmű egyik blokkja és három távvezeték,
- túlterhelődtek a távvezetékek,
- a First Energy operátorai nem tudták megfelelően kezelni a helyzetet …
- … mert nem működött az informatikai rendszerük órákon át (és erről nem is tudtak) …
- … mert **versenyhelyzet** volt a GE Unix-alapú rendszerében (több millió kódsor)
- *az észak-amerikai villamosenergia-rendszer összértéke kb. 1 billió USD (≈273 billió HUF akkori árfolyamon, 100 évnyi magyar GDP)*

Kölcsönös kizárás példa

kliens$_i$	**szerver**
1 … other activity … 2 $r_i := TRUE$ 3 **wait until** $g_i = TRUE$ 4 critical section 5 $r_i := FALSE$ 6 **go to** 1	1 **wait until** at least one r_i is $TRUE$ 2 let c be such that $r_c = TRUE$ 3 $g_c := TRUE$ 4 **wait until** $r_c = FALSE$ 5 $g_c := FALSE$ 6 **go to** 1

Algoritmusok helyességének ellenőrzése

- Hogyan döntsük el, hogy jó?
- Erősen nézzük, és próbálunk rájönni :)
- Végigpróbálunk néhány lefutást
  - Ha hibázik: javítjuk a kódot

Kölcsönös kizárás példa

kliens$_i$	**szerver**
1 … other activity … 2 $r_i := TRUE$ 3 **wait until** $g_i = TRUE$ 4 critical section 5 $r_i := FALSE$ 6 **go to** 1	1 **wait until** at least one r_i is $TRUE$ 2 let c be such that $r_c = TRUE$ 3 $g_c := TRUE$ 4 **wait until** $r_c = FALSE$ 5 $g_c := FALSE$ 6 **go to** 1

Kölcsönös kizárás példa

kliens$_i$

1 … other activity …
2 $r_i := TRUE$
3 **wait until** $g_i = TRUE$
4 critical section
5 $r_i := FALSE$
6 **go to** 1

```
public void run() {
while (true) {
        other_activity();
        RaceCond.set_request(id);
        while (RaceCond.get_grant(id)
                == false)
        ;
        critical_section();
        RaceCond.release_request(id);
        other_activity();
}
}
```

Kölcsönös kizárás példa

```
int chosen_id = -1;
for (int i = 0; i < client_number;
i++) {
if (RaceCond.get_request(i) ==
true) {
chosen_id = i;
break;}}
if (chosen_id > -1) {
RaceCond.set_grant(chosen_id);
while (RaceCond.
        get_request(chosen_id)
        == true);
RaceCond.
        release_grant(chosen_id);
}
```

szerver

1 **wait until** at least one r_i is $TRUE$
2 let c be such that $r_c = TRUE$
3 $g_c := TRUE$
4 **wait until** $r_c = FALSE$
5 $g_c := FALSE$
6 **go to** 1

DEMO

- Kérdés: helyes-e az algoritmus?
  - Biztosítja-e a kölcsönös kizárást?

Algoritmusok helyességének ellenőrzése

- Hogyan döntsük el, hogy jó?
- Erősen nézzük, és próbálunk rájönni :)
- Végigpróbálunk néhány lefutást
  - Ha hibázik: javítjuk a kódot
  - **Ha nem találunk hibát: ??**
- Szisztematikus megoldás kell:
  - „formális módszerek”

Algoritmusok helyességének ellenőrzése

- Milyen jó lenne egy eszköz:
  - Algoritmusaink egyszerű **leírására**
  - Rendszer működésének **szimulálására**
  - Összetett követelmények **megfogalmazására**
  - Követelmények **ellenőrzésére** gombnyomásra

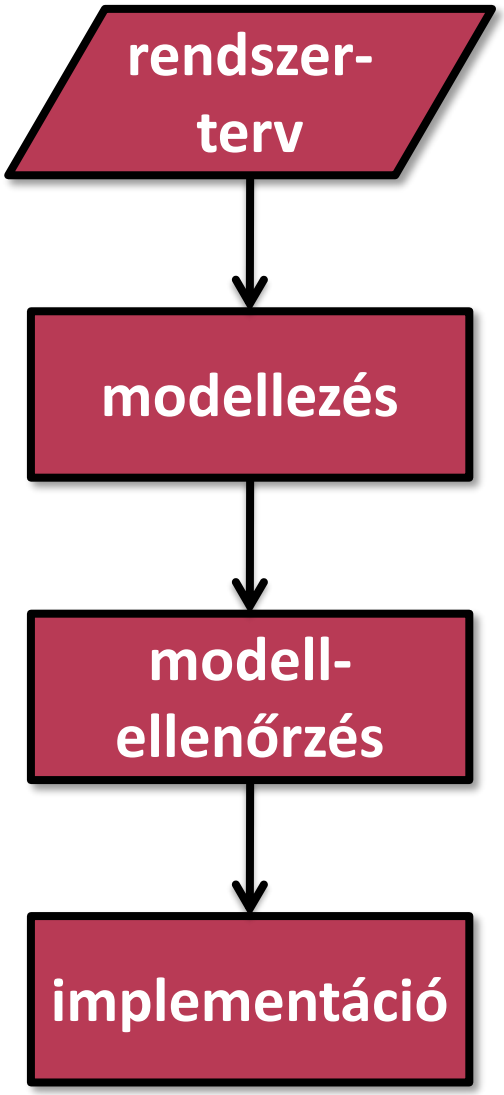
- Jó hír: vannak ilyen eszközök ☺
  - **Modellellenőrzők** (model checkers)
  - 30+ év kutatás eredménye
  - Valós ipari eredmények HW és SW rendszereknél

Turing award

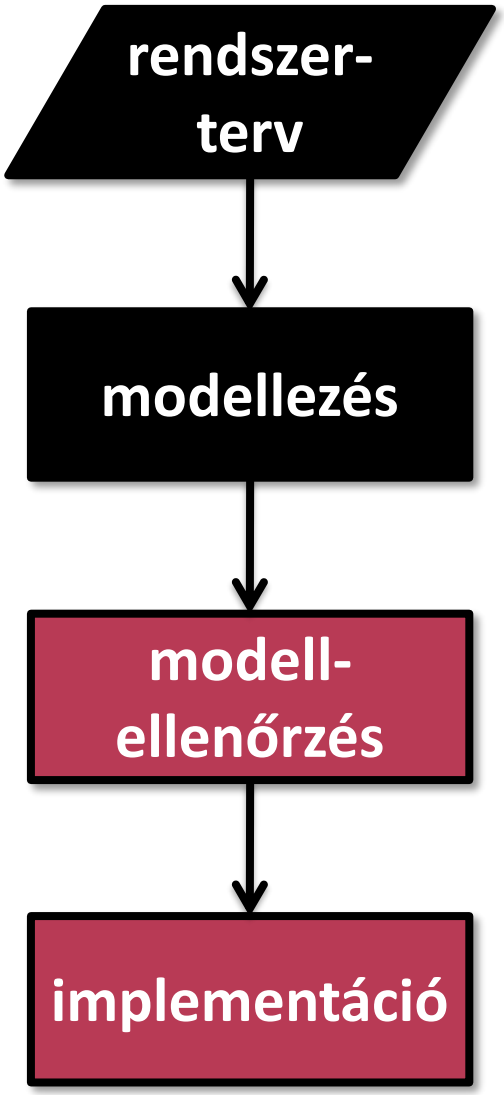
- Edmund M. Clarke,
- E. Allen Emerson,
- Joseph Sifakis

- „For their role in developing Model-Checking into a highly effective verification technology that is widely adopted in the hardware and software industries.”

Modellvezérelt fejlesztés



Modellvezérelt fejlesztés



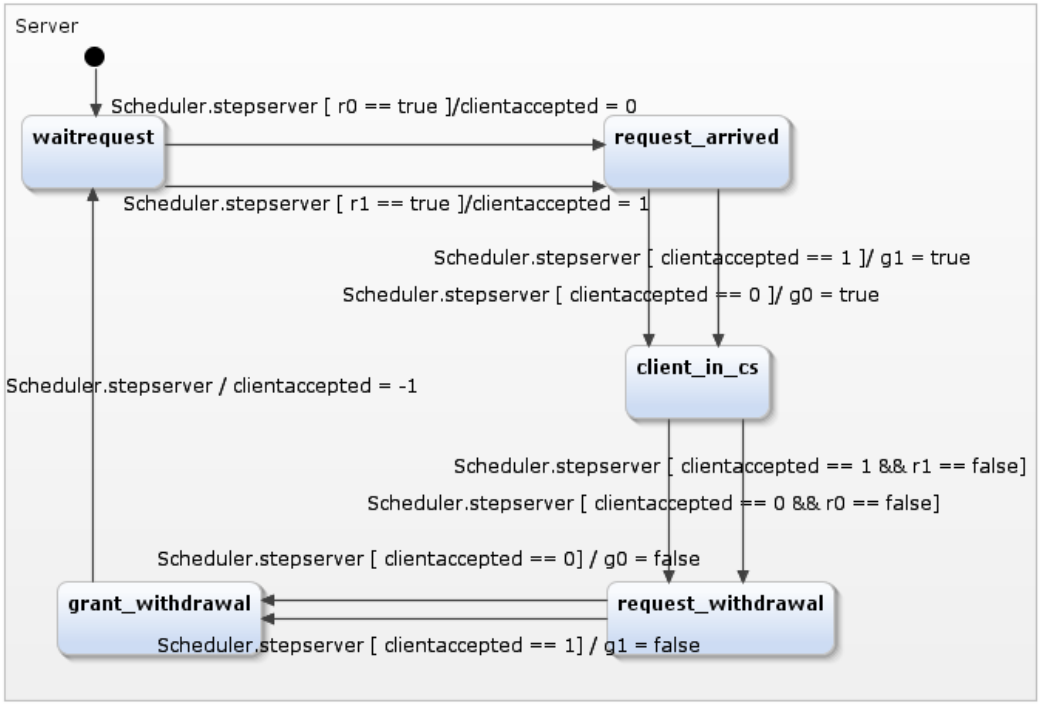
Rendszerterv

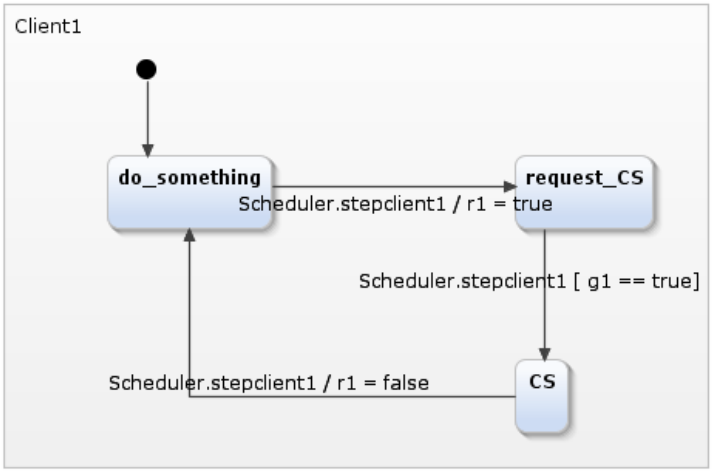
- Pszeudo kód

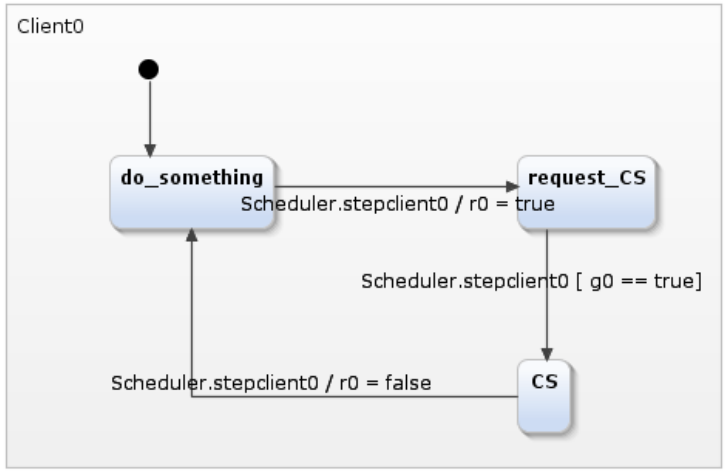
kliens$_i$	**szerver**
1 … other activity … 2 r_i := $TRUE$ 3 **wait until** g_i = $TRUE$ 4 critical section 5 r_i := $FALSE$ 6 **go to** 1	1 **wait until** at least one r_i is $TRUE$ 2 let c be such that r_c = $TRUE$ 3 g_c := $TRUE$ 4 **wait until** r_c = $FALSE$ 5 g_c := $FALSE$ 6 **go to** 1

Rendszerterv

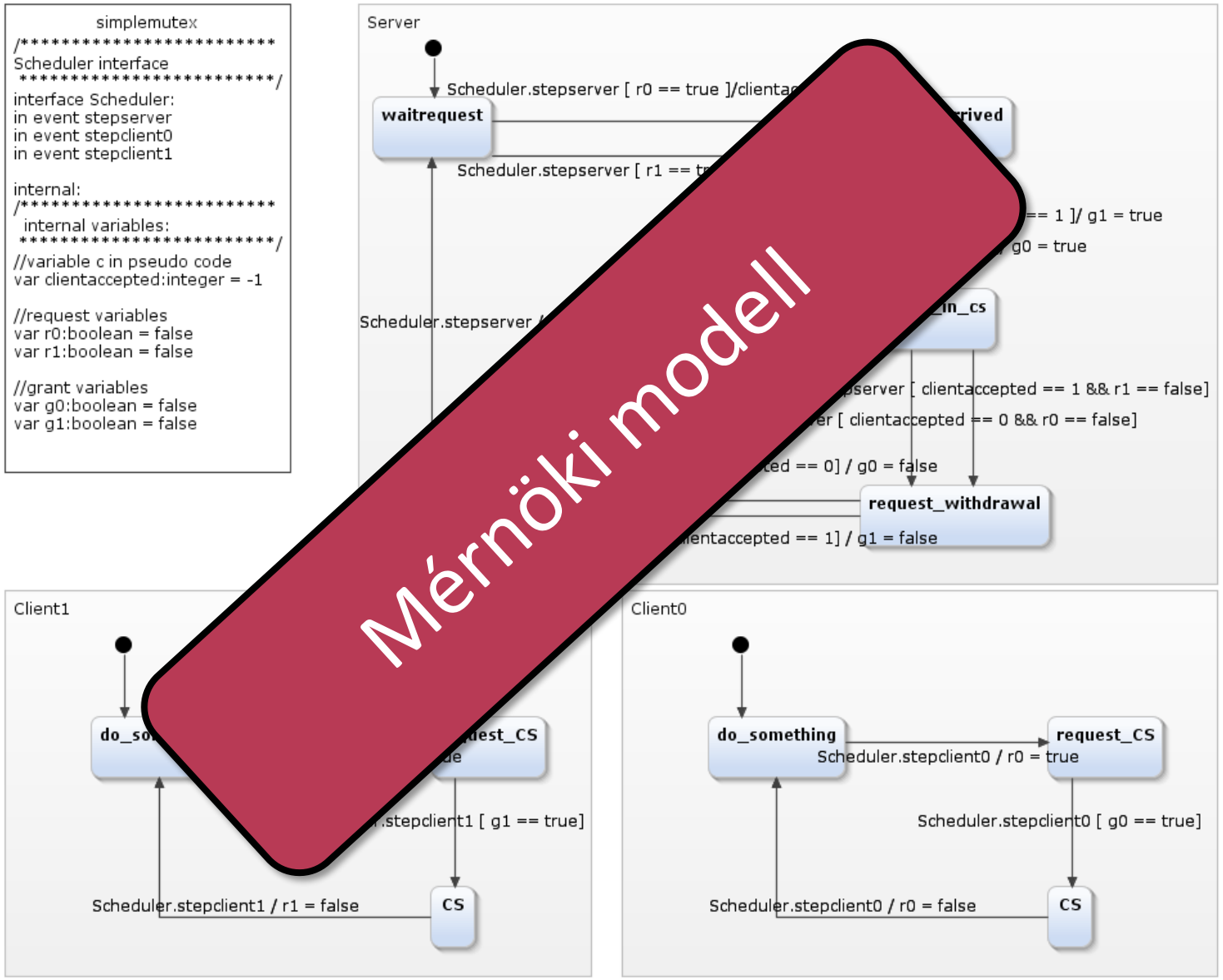
```
simplemutex
/*************************
Scheduler interface
 *************************/
interface Scheduler:
in event stepserver
in event stepclient0
in event stepclient1

internal:
/*************************
  internal variables:
 *************************/
//variable c in pseudo code
var clientaccepted:integer = -1

//request variables
var r0:boolean = false
var r1:boolean = false

//grant variables
var g0:boolean = false
var g1:boolean = false
```



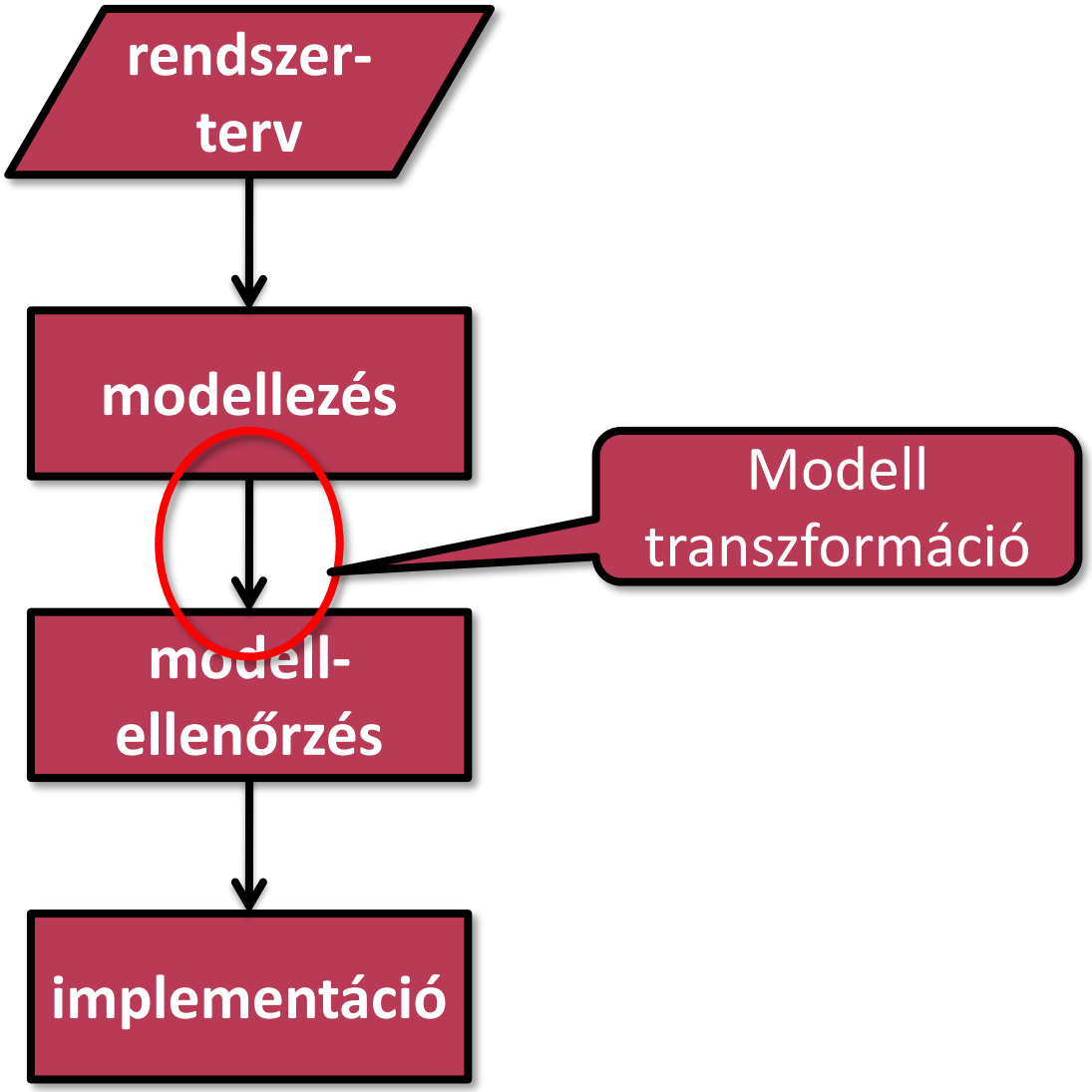




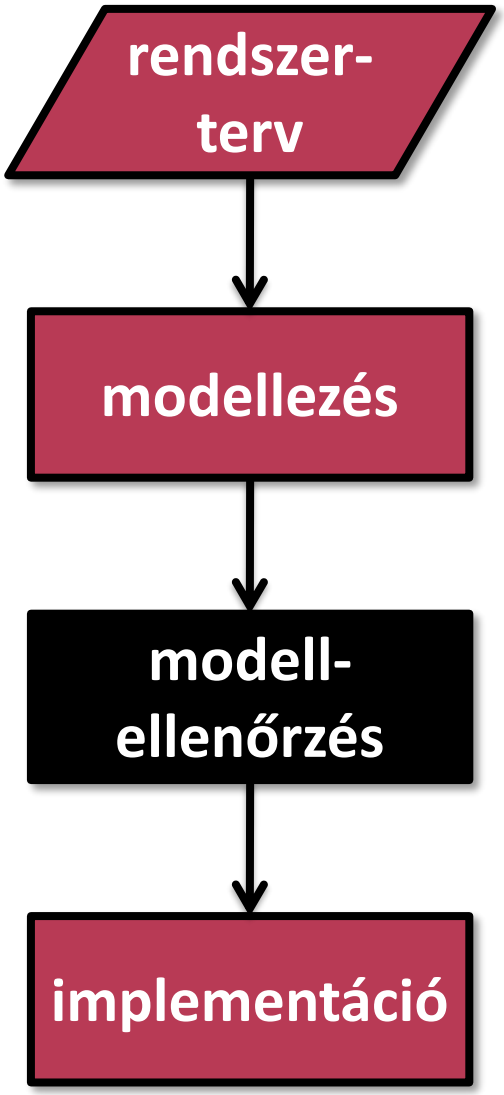
Rendszerterv



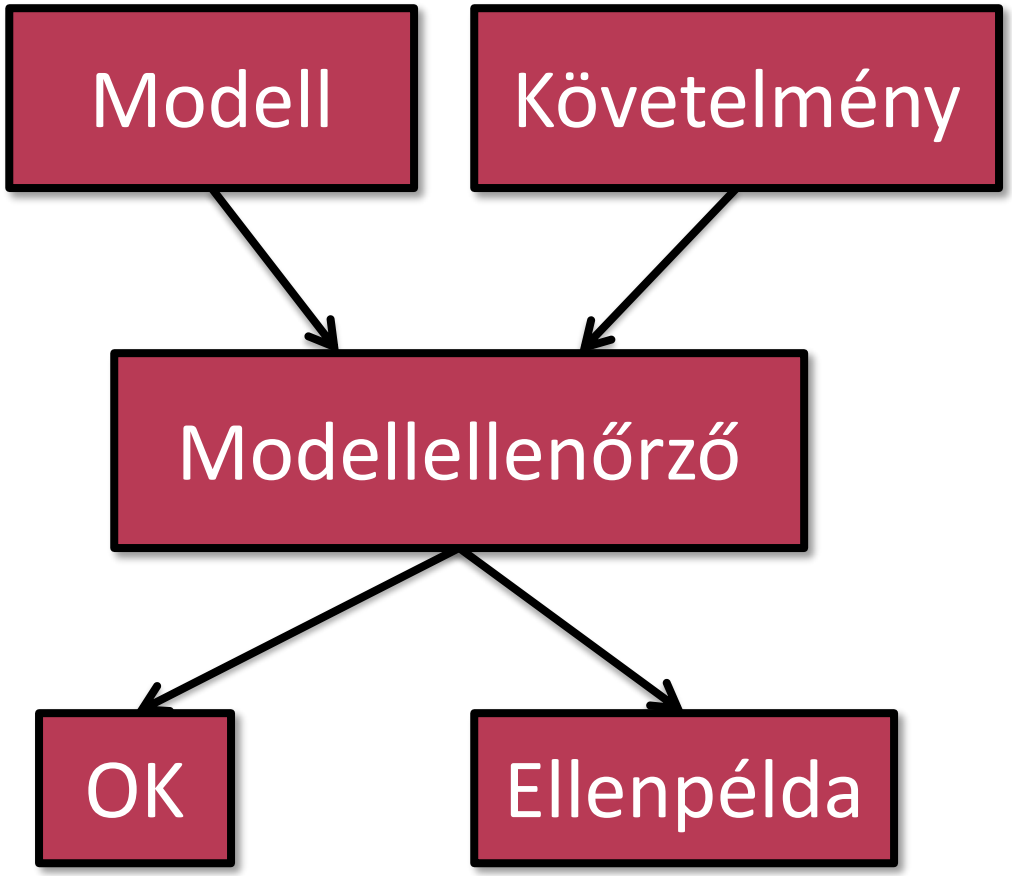
Modellvezérelt fejlesztés



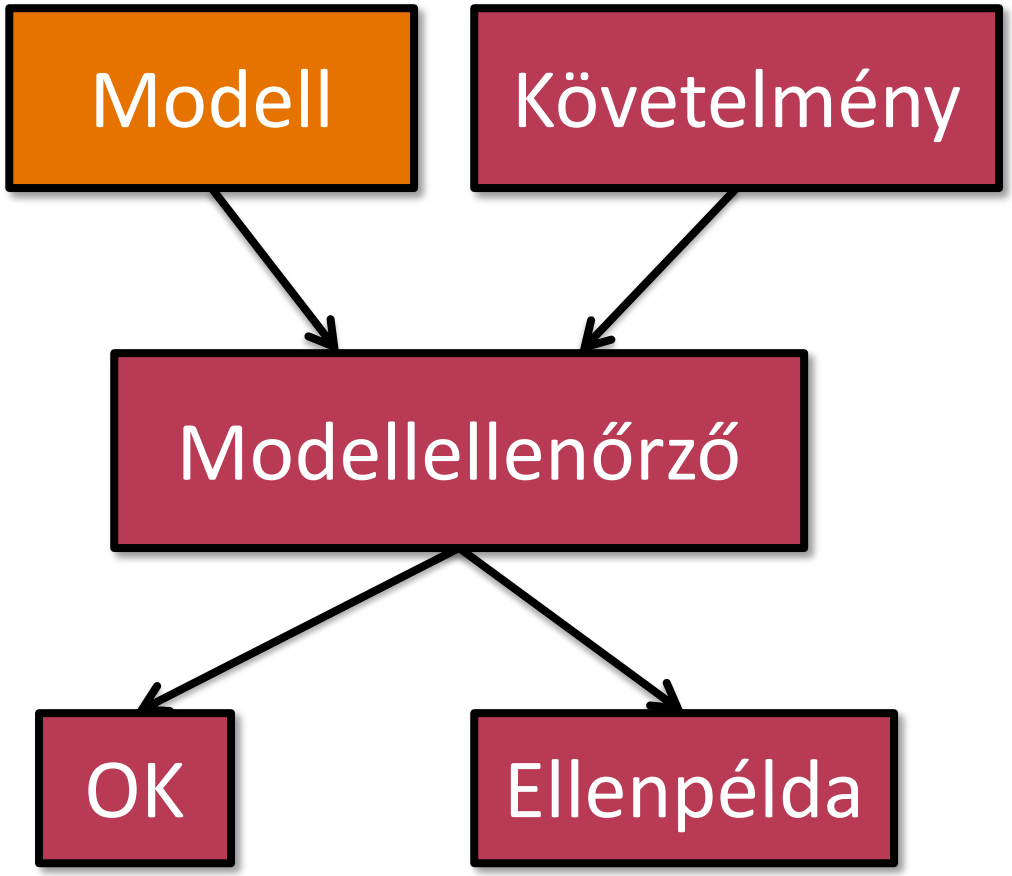
Modellvezérelt fejlesztés



Modellellenőrzők



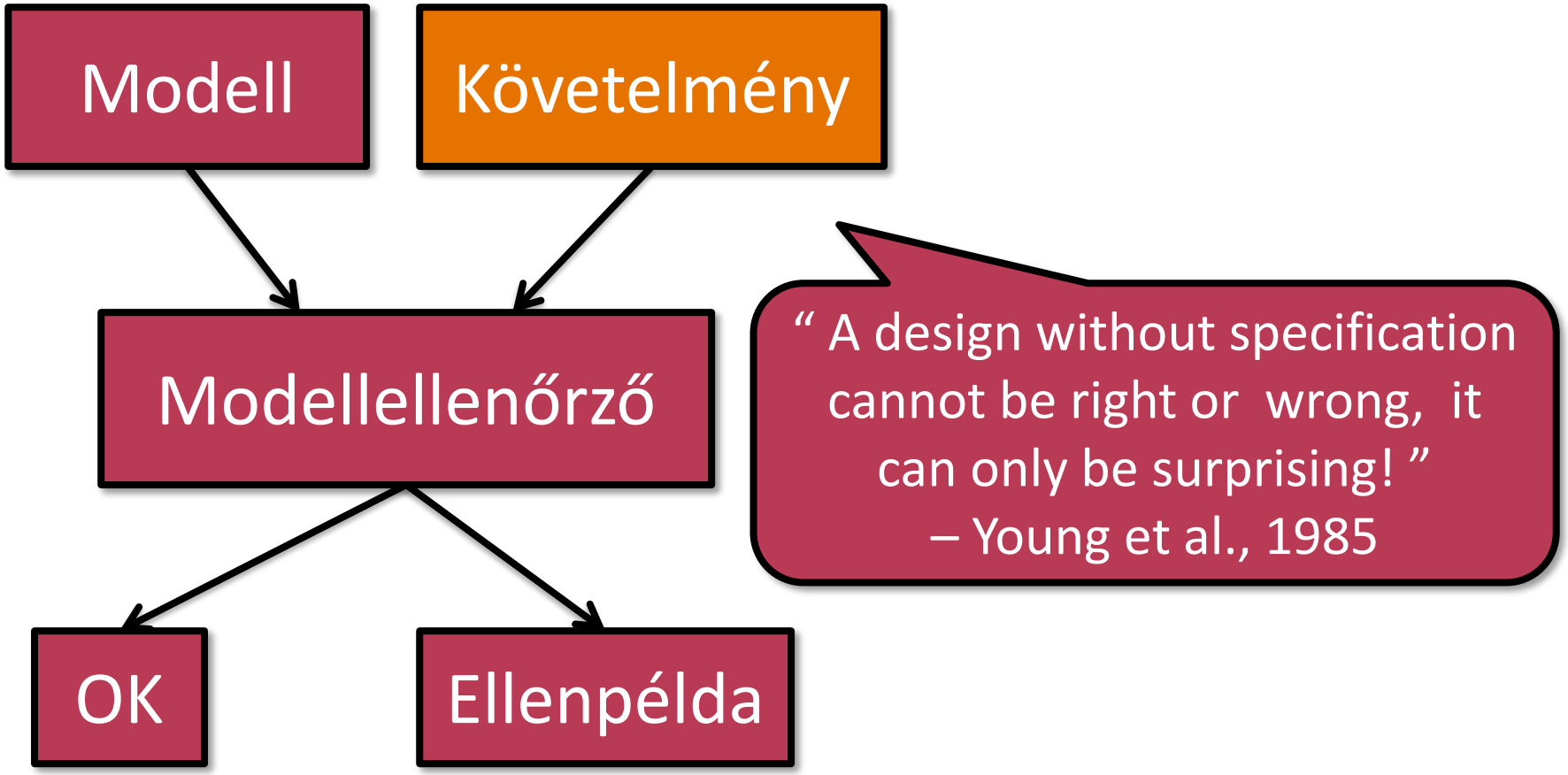
Modellellenőrzők



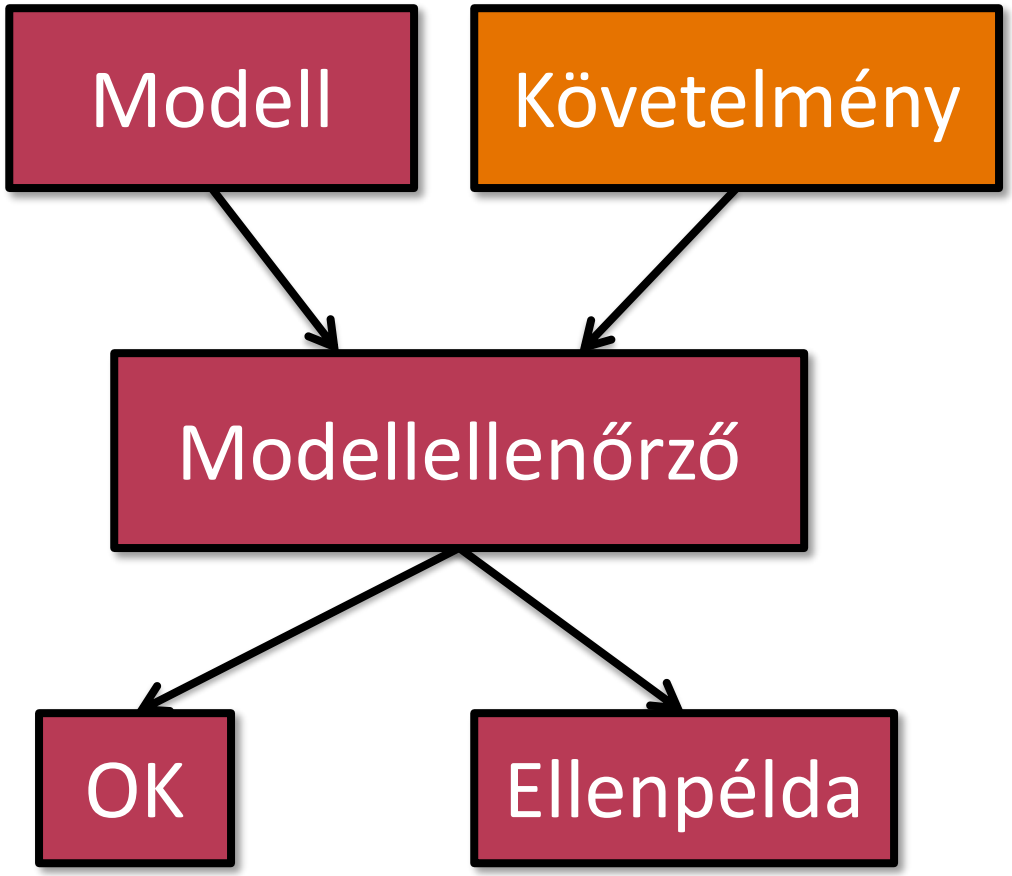
- Rendszer működésének leírása
- Formális modell
- Tipikusan állapotgépszerű

De: lehet program is!

Modellellenőrzők



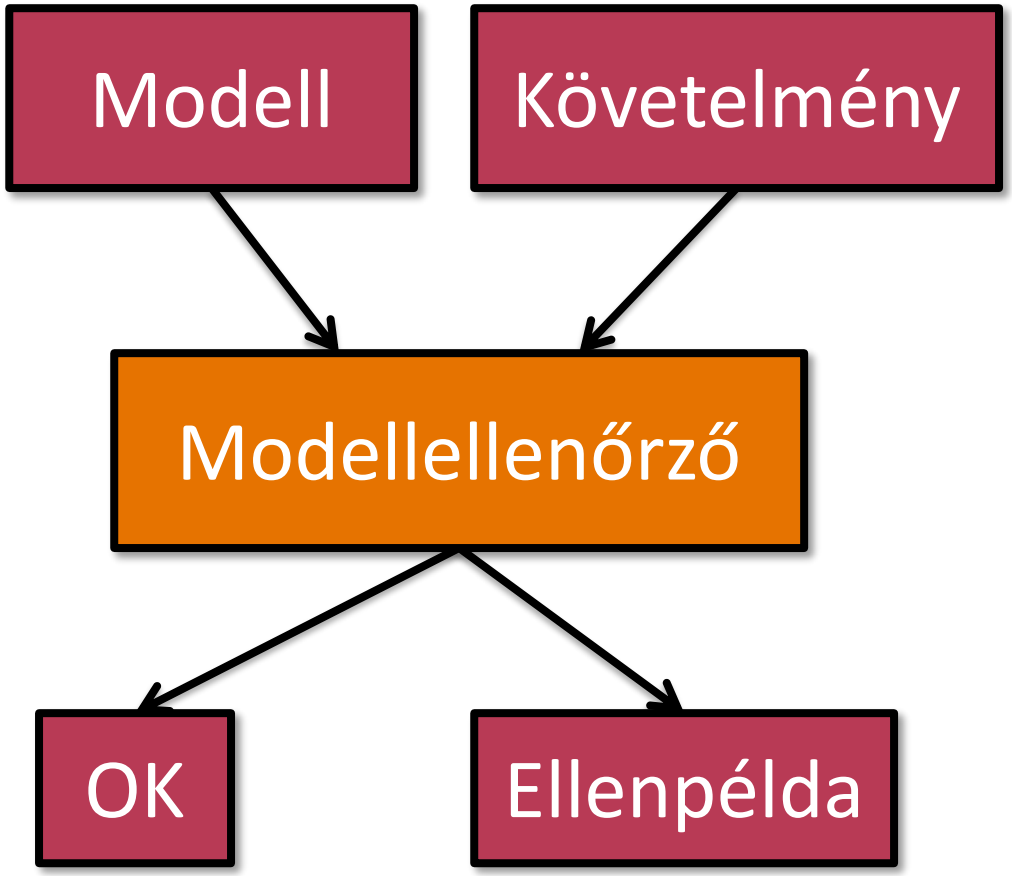
Modellellenőrzők



- Mit akarunk ellenőrizni
  - Kölcsönös kizárás
  - Holtpont mentesség
  - …
- Logikai kifejezés:
  - Pl.:

  ! (A_var AND B_var)

Modellellenőrzők

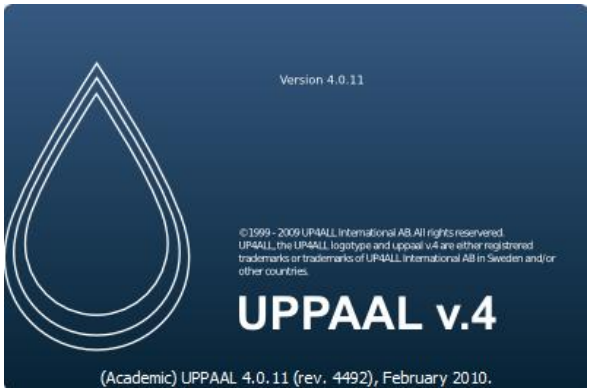


- „Fekete doboz”
- Automatikus

- Eredmény:
  - Követelmény igaz
  - Követelmény nem teljesül + ellenpélda

UPPAAL

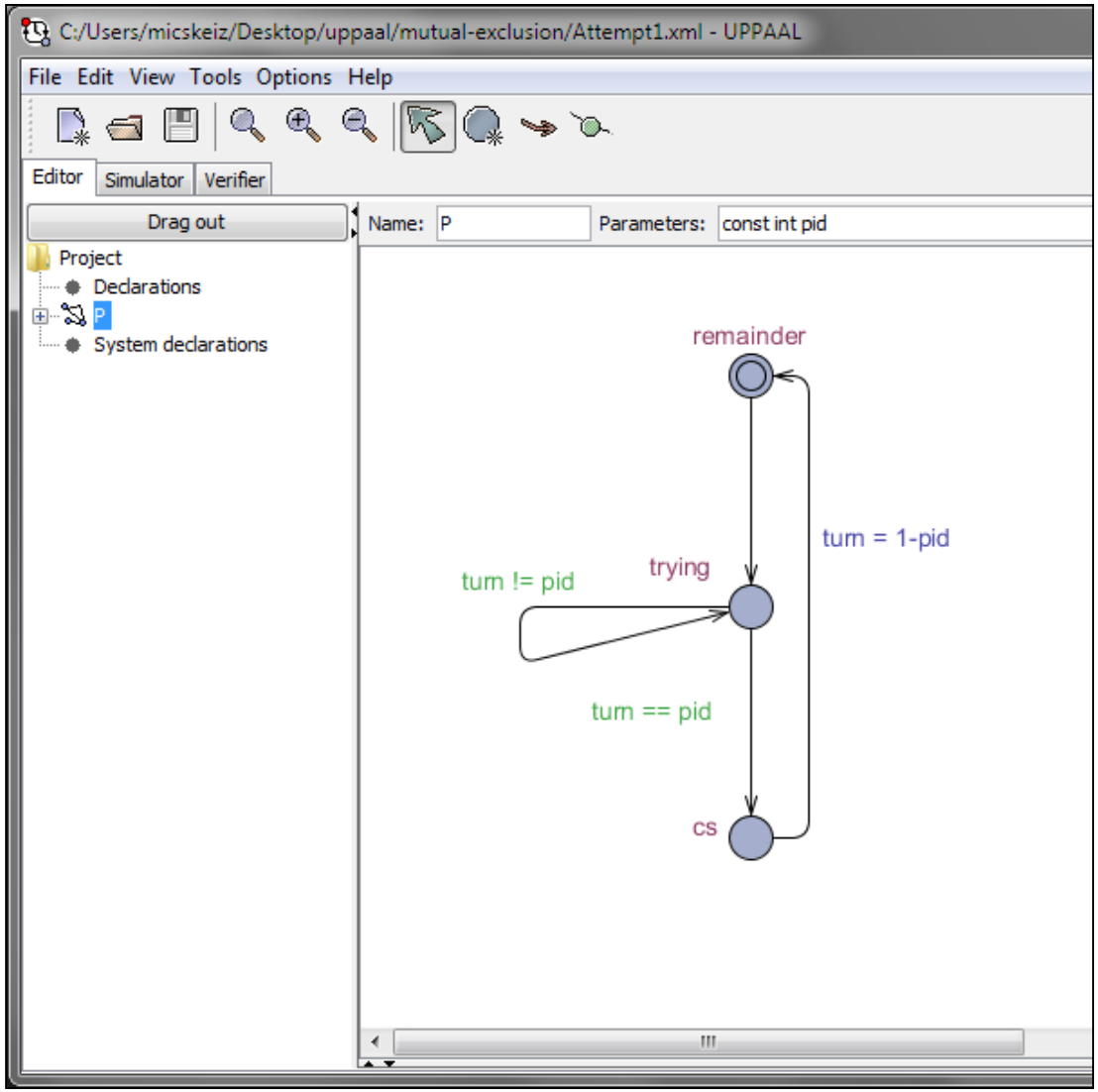
- Időzítést is támogató modellellenőrző
- Uppsala & Aalborg egyetemek, 15+ éve fejlesztik
- Cél: hatékonyság, könnyű használhatóság
- http://www.uppaal.com/
  - Akadémiai célra ingyenesen letölthető
  - Leírások
  - Részletes súgó
  - Esettanulmányok
  - Sok kiegészítés (tesztgenerálás)



DEMO Ismerkedés az UPPAAL-lal

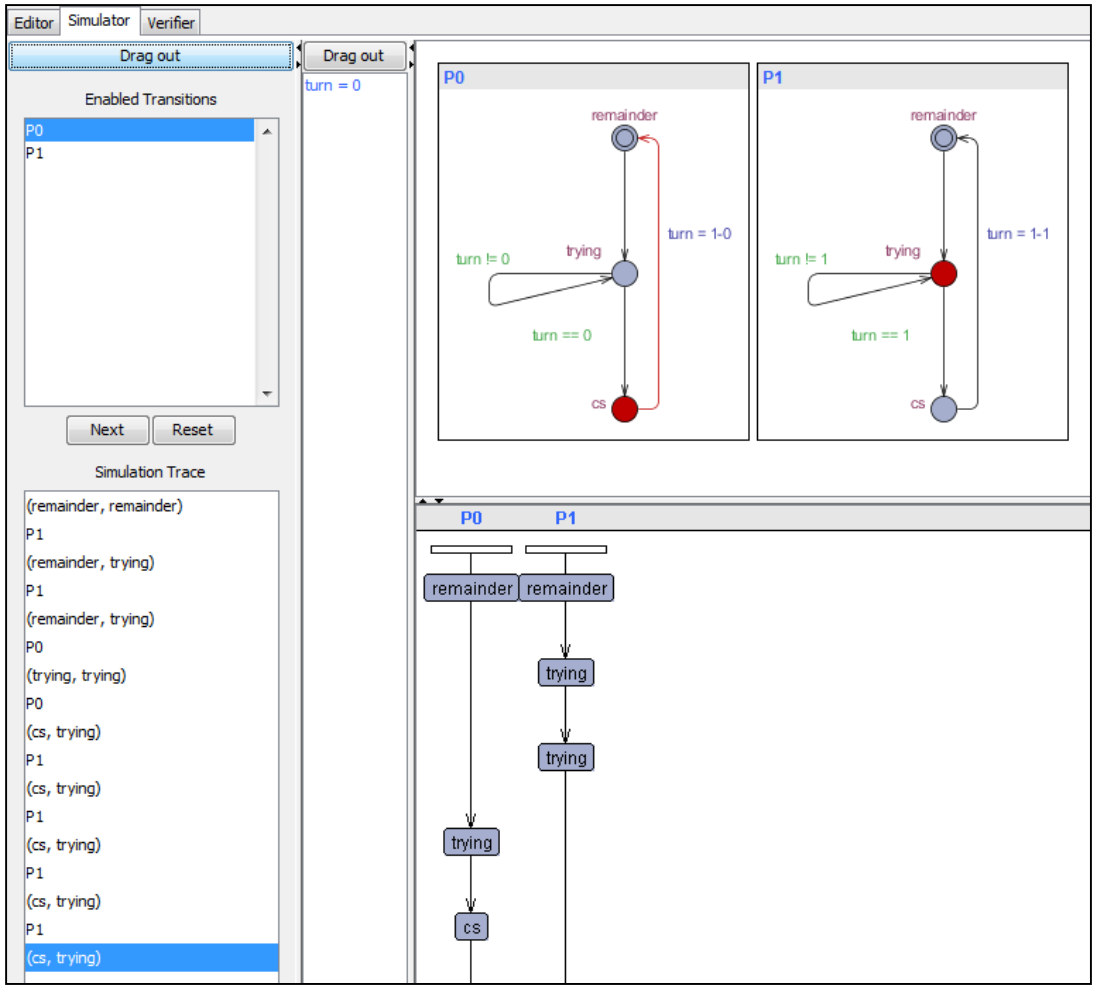
- Példa modell megnyitása
  - OPRE-hoz kapcsolódó modellek: http://www.mbsd.cs.ru.nl/publications/papers/fvaan/MCinEdu/
- Deklarációk megnézése
- Szimulátor:
  - Modell „animálása”
  - Végrehajtás visszajátszása
  - Véletlenszerű végrehajtás

Az UPPAAL felülete: modell szerkesztő



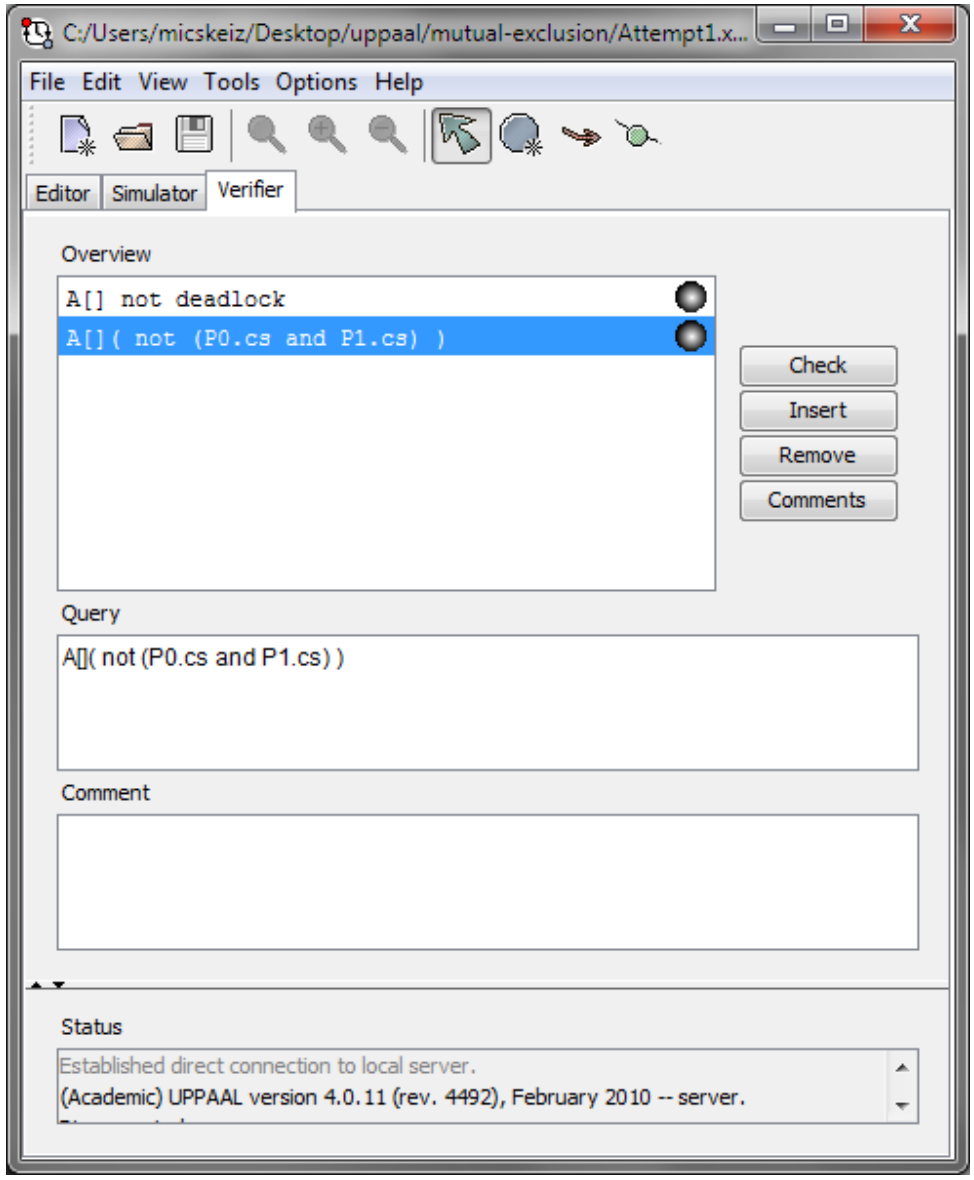
- Globális változók
- Automata
  - Állapot
  - Átmenet
    - Őrfeltétel
    - Akció
  - Órák
- Rendszer:
  - Automata példányok

Az UPPAAL felülete: szimulátor



- Átmenet kiválasztása
- Változók állapota
- Automaták képe
- Trace:
  - Szöveges
  - Grafikus: *Message Sequence Charts*

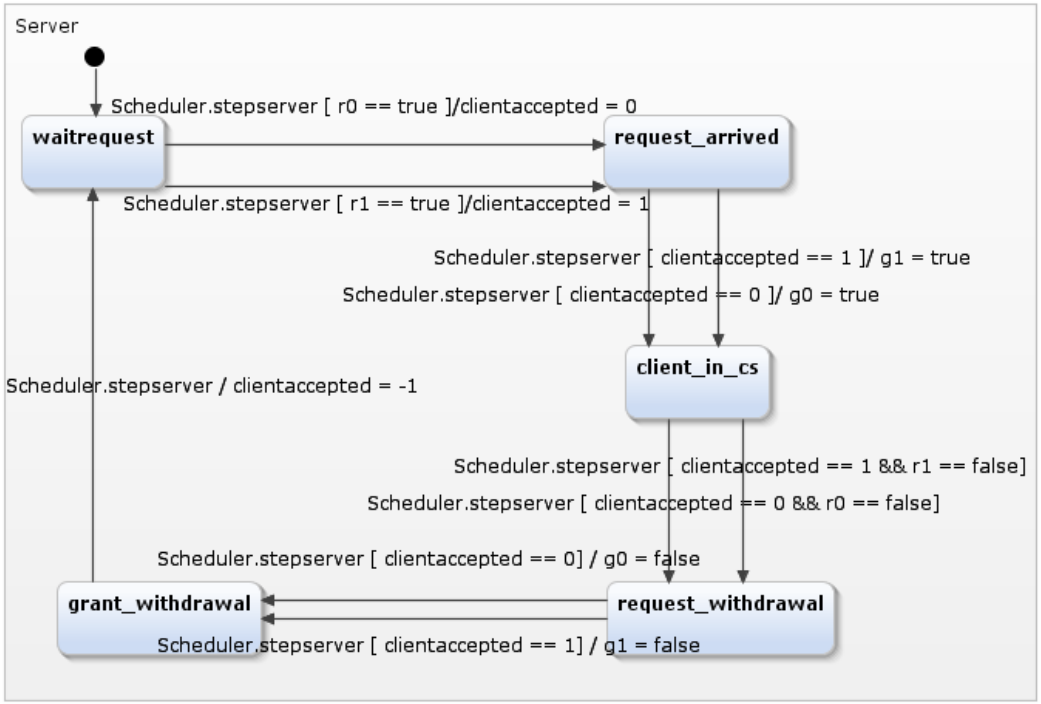
Az UPPAAL felülete: ellenőrzés

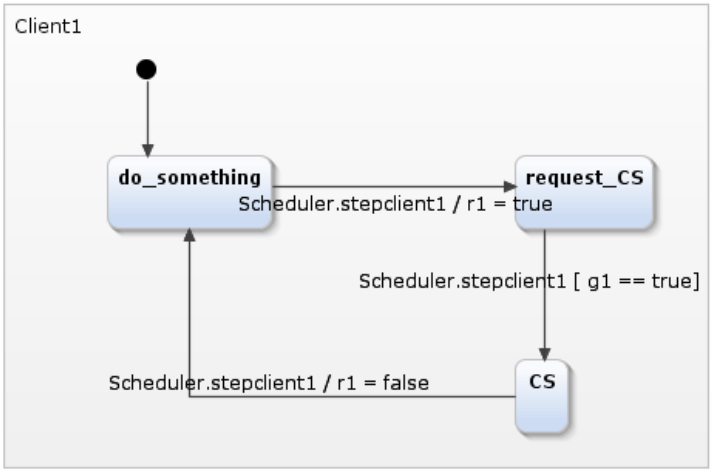


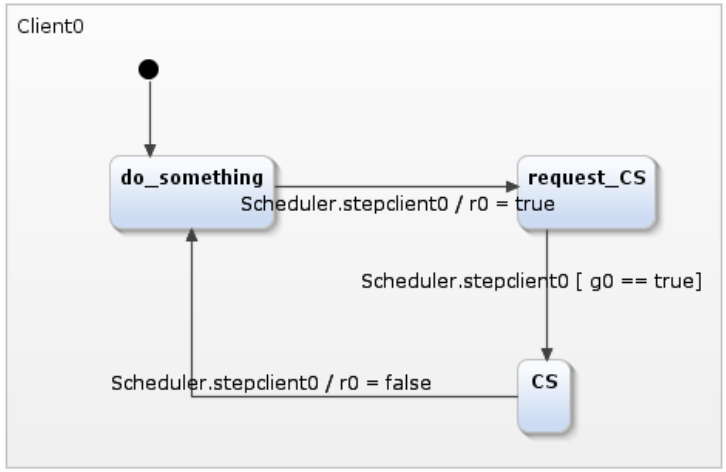
- Követelmény:
  - Logikai formula
- Elemei:
  - Állapotra hivatkozás
  - NOT, AND, OR
- További operátorok:
  - A: minden úton
  - E: legalább egy úton
  - [ ]: minden időben
  - <>: valamikor a jövőben

Vissza a saját algoritmusunkhoz

```
simplemutex
/*************************
Scheduler interface
 *************************/
interface Scheduler:
in event stepserver
in event stepclient0
in event stepclient1

internal:
/*************************
  internal variables:
 *************************/
//variable c in pseudo code
var clientaccepted:integer = -1

//request variables
var r0:boolean = false
var r1:boolean = false

//grant variables
var g0:boolean = false
var g1:boolean = false
```

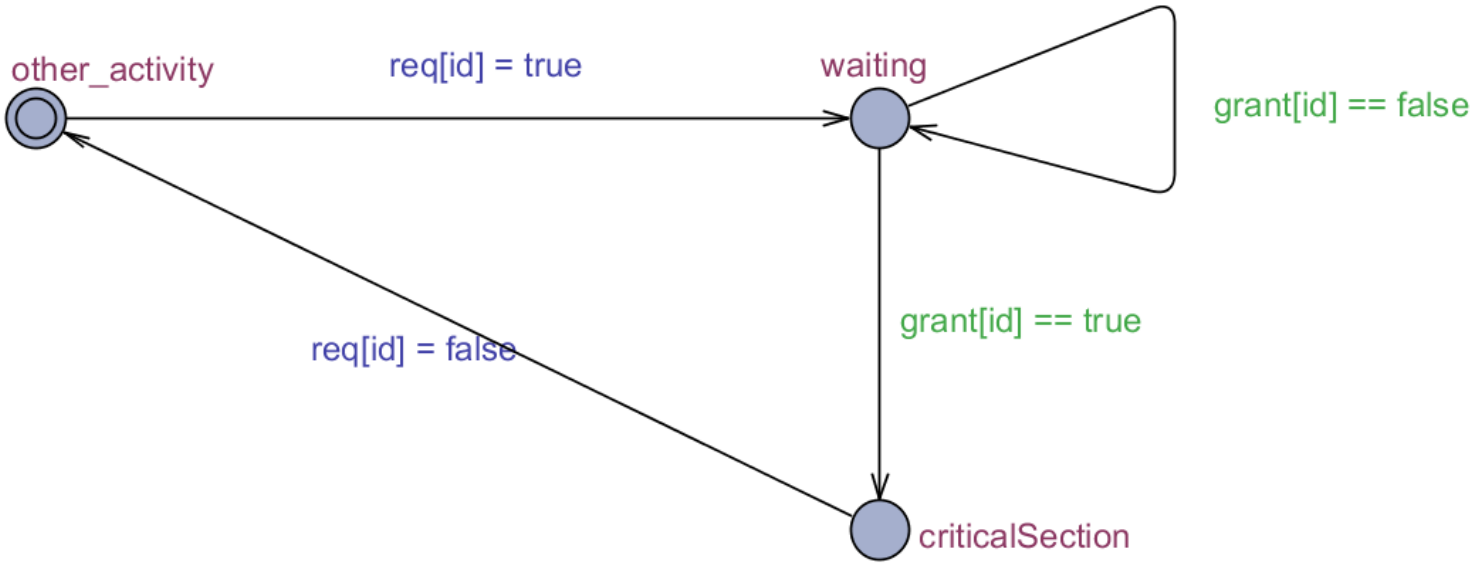






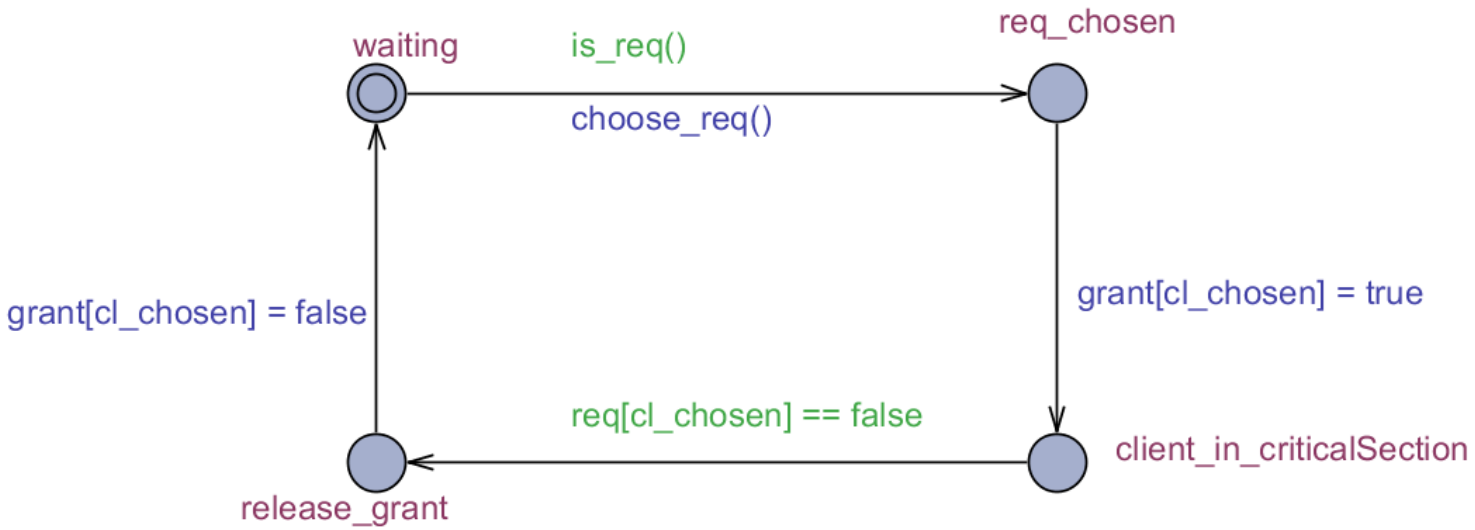
Kölcsönös kizárás példa - Kliens

1 … other activity …
2 r_i := *TRUE*
3 wait until g_i = *TRUE*
4 critical section
5 r_i := *FALSE*
6 go to 1



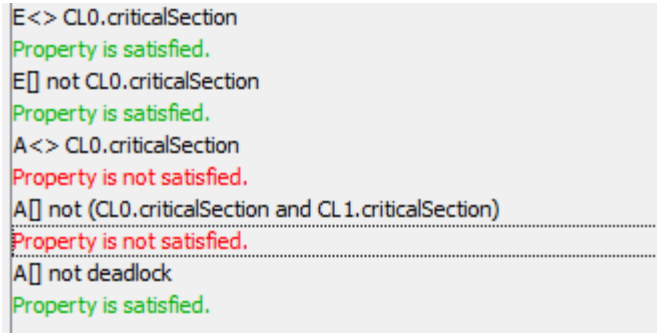
Kölcsönös kizárás példa

1 wait until at least one r_i is $TRUE$
2 let *c* be such that $r_c = TRUE$
3 $g_c := TRUE$
4 wait until $r_c = FALSE$
5 $g_c := FALSE$
6 go to 1



DEMO Kölcsönös kizárás

- Algoritmusokat leíró modellek vizsgálata
- Szimuláció
- Követelmények ellenőrzése:
  - Egyszerre csak egy példány lehet a kritikus szakaszban:
  - **A[] not (CL0.criticalSection and CL1.criticalSection)**
- Ellenpélda generálása:
  - *Options / Diagnostic Trace / Shortest*



Motiváció

- Történelem során problémák az algoritmusokkal:
  - Harris Hyman, Comments on a problem in concurrent programming control, Communications of the ACM, v.9 n.1, p.45, Jan. 1966

Hyman algoritmusa

```
turn=0, flag[0]=flag[1]=false;

Protocol (int id) {
  do {
      flag[id] = true ;
      while (turn != id) {
            while (flag[1-id]) /* do nothing */ ;
            turn = id;
      }
      CriticalSection(id);
      flag[id] = false;
  } while (true) ;
}
```

Lehetnek-e ketten egyszerre a kritikus szakaszban?

Peterson algoritmusa

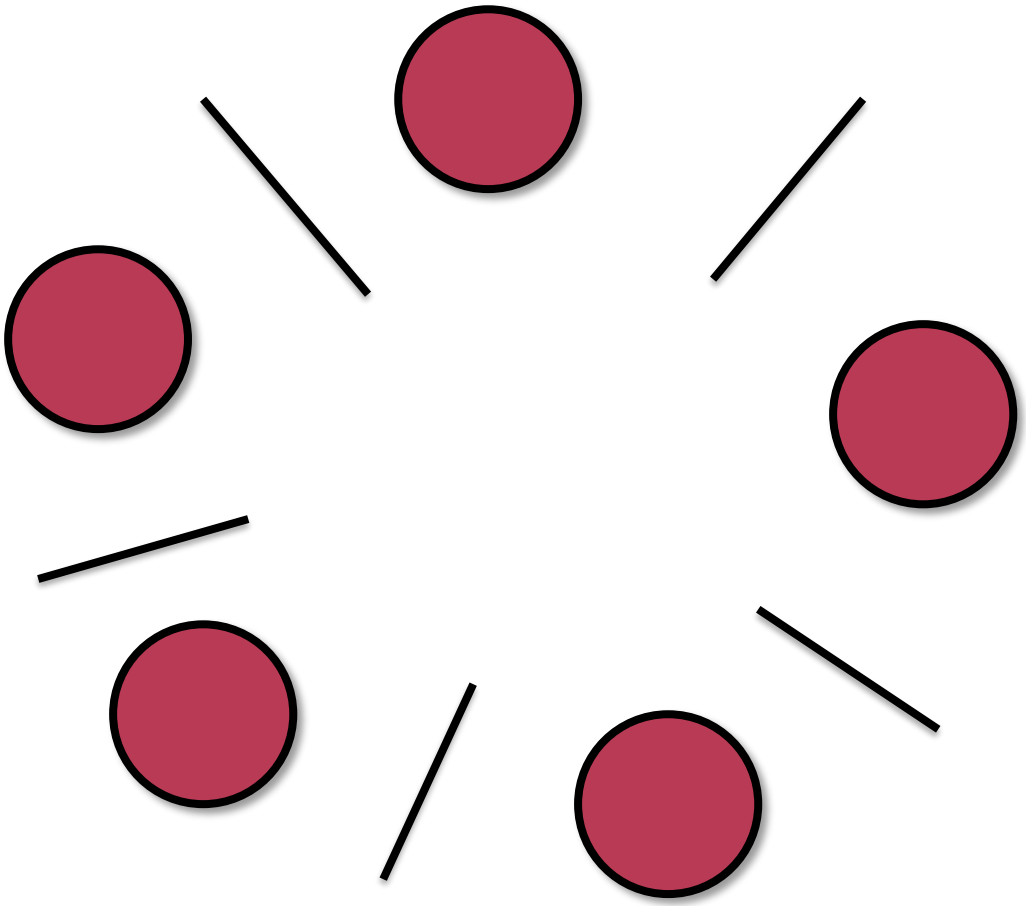
```
turn=0, flag[0]=flag[1]=false;

Protocol (int pid) {
   while (true) {
     flag[pid]=true;
     turn=1-pid;
     while (flag[1-pid]&&turn==1-pid) /**/;
     CriticalSection(id);
     flag[pid]=false;
   }
}
```

Lehetnek-e ketten egyszerre a kritikus szakaszban?

Étkező filozófusok

Kép: en.wikipedia.org

Étkező filozófusok

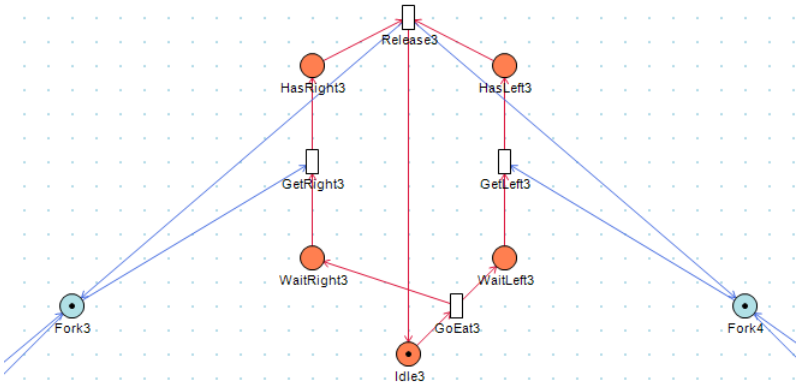


DEMO Holtpont – Étkező filozófusok

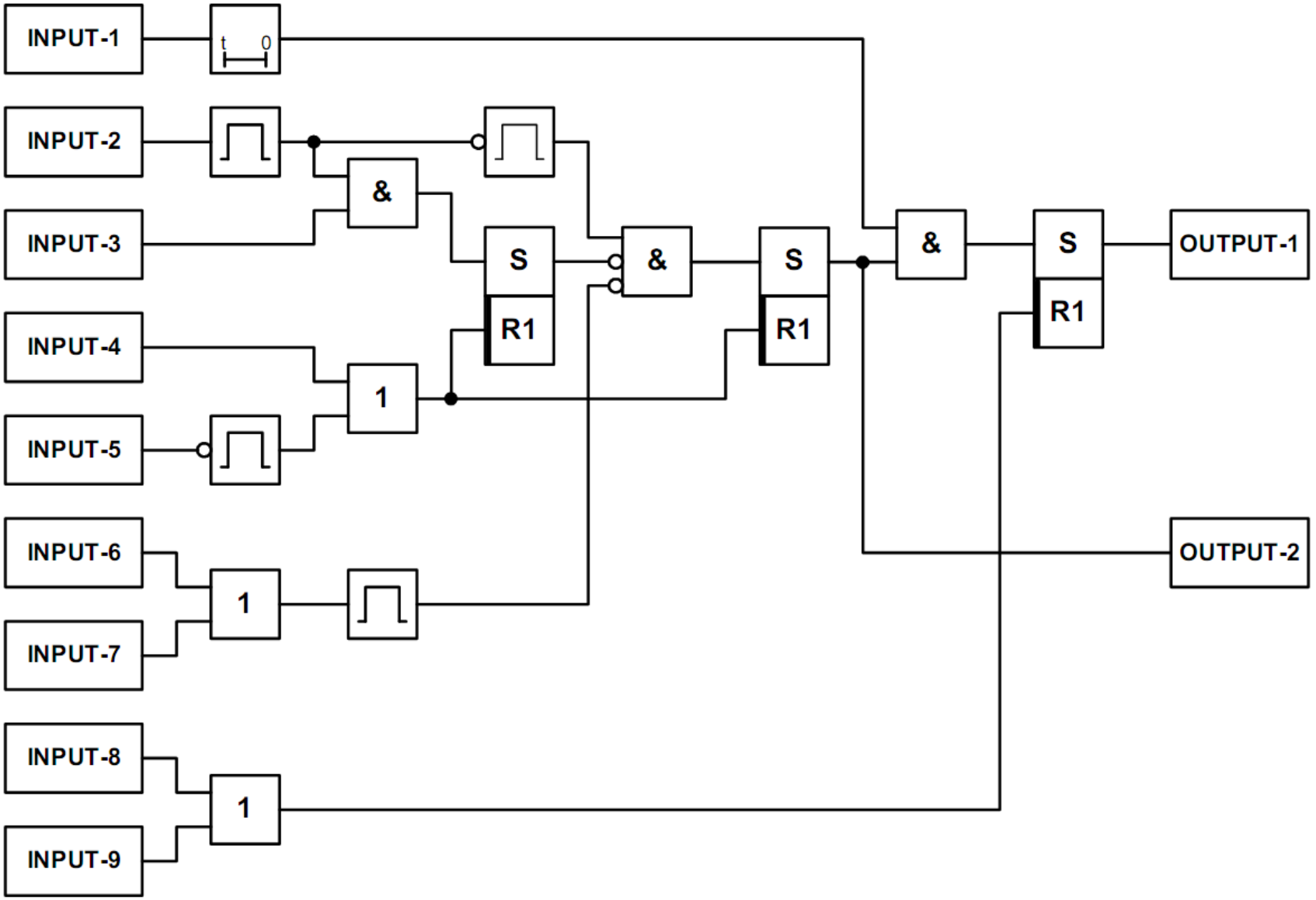
- petridotnet from BME-MIT - Tanszéki fejlesztés
  - TDK díjak az elmúlt években:
    - 6 első helyezés,
    - 3 második
  - OTDK: 

DEMO Holtpont – Étkező filozófusok

- petridotnet from BME-MIT - Tanszéki fejlesztés
- Modellek:
  - Petri-háló és Színezett Petri-háló
- Sokféle analízis lehetőség
  - Temporális logikai specifikációk
  - Végtelen állapotterű rendszerek vizsgálata



Paksi atomerőmű biztonsági logikája



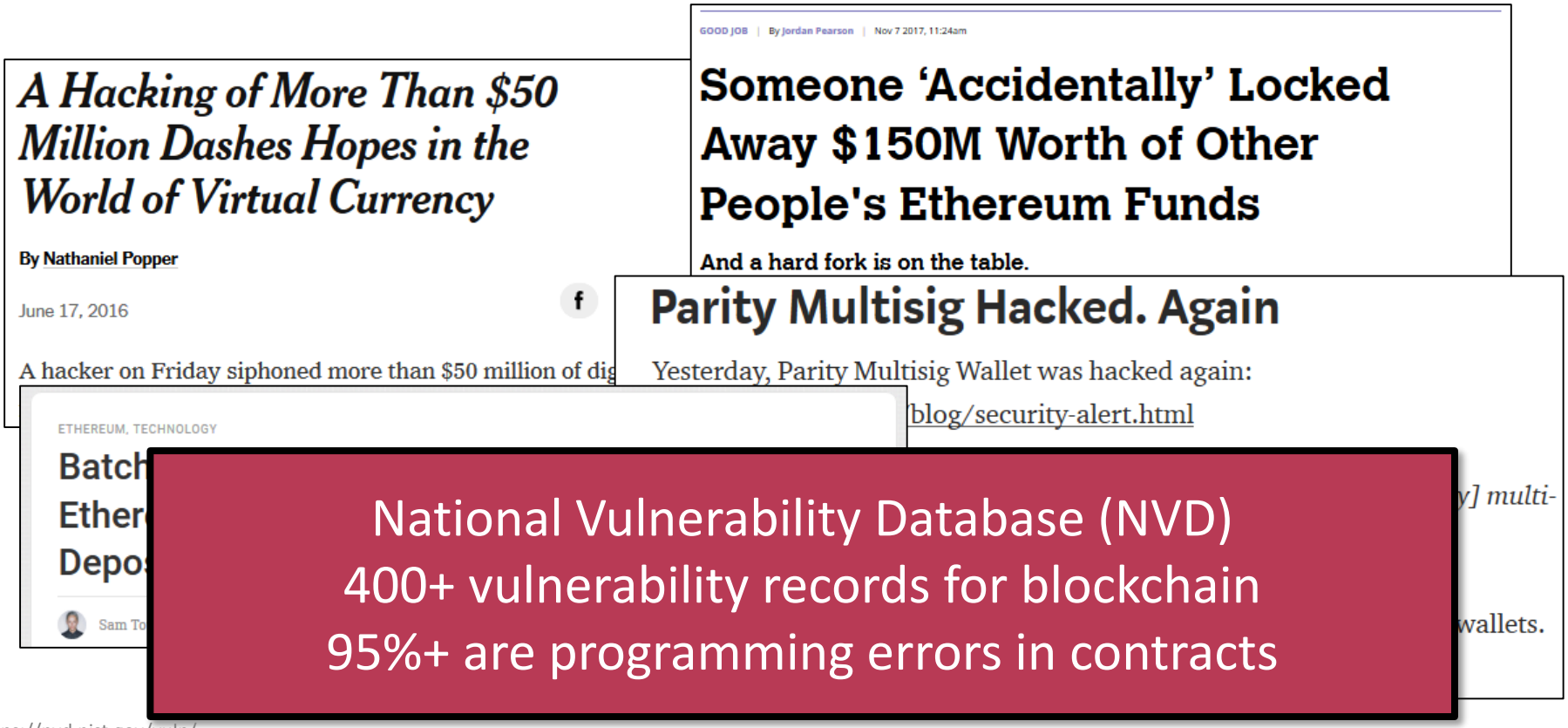
Állapotfelderítés jellemzői

- Futási idő: 950 s
- Memóriafoglalás: 2,53 GB
- Globális állapotok száma: $4{,}8 \cdot 10^{12}$

Szimbolikus kódolás hatékonysága:

- Állapottér MDD csomópontszáma: kb. 1500

Errors in Smart Contracts



https://nvd.nist.gov/vuln/

Formal verification of Smart Contracts

SRI approach: automated formal verification

- Specification annotations
- Automatic translation to verification language
- Modular reasoning with logic solvers
- Automated, user friendly, expressive
- Correct and precise

Érdeklődők számára: TDK nyertes dolgozatok

- Dobos-Kovács Mihály: Formális verifikációval támogatott tesztgenerálás autóipari környezetben (I. helyezett **Rektori különdíj**)
- Bajczi Levente: Konkurens programok HW-SW együttes verifikációja (I. helyezett **Rektori különdíj**)
- Klenik Attila, Marussy Kristóf: Configurable stochastic analysis framework for asynchronous systems (I. helyezett **Rektori különdíj**)
- Sallai Gyula: Fordító optimalizációk szoftver verifikációhoz
- Élő Dániel, Soltész Adrián: Symbolic model checking and trace generation by guided search
- Molnár Vince, Segesdi Dániel: Múlt és jövő: Új algoritmusok lineáris temporális tulajdonságok szaturáció-alapú modellellenőrzésére.
- Hajdu A
  viselke ás
  (CEGAR

https://inf.mit.bme.hu/edu/results/tdk

További eszközök

- Java Pathfinder
  - Modellellenőrző Java byte kódhoz
  - NASA fejlesztés, 2005 óta nyílt forrású
- CHESS
  - .NET-es kódokhoz
  - Párhuzamosságból fakadó hibák keresése
- jchord
  - Java kód statikus analízise
  - Versenyhelyzet, holtpont detektálás
- Facebook: Infer
  - Statikus analízis
  - C, Java, Objective-C

További eszközök

- Static Driver Verifier (SDV, korábban SLAM)
  - Windows eszközmeghajtók ellenőrzése
- Linux Driver Verification
  - Linux eszközmeghajtók ellenőrzése
- Linux Deductive Verification
  - Linux kernel kód ellenőrzése

Alkalmazási példák, pár siker

- Futurebus+ Cache Coherence Protokoll
  - IEEE szabványban hibát találtak
- Microsoft Hyper-V Hypervisor ellenőrzése
  - Virtualizációs technológia
- FlexRay (for Avionics) protokoll verifikációja
  - UPPAAL segítségével
- Airbus repülőgépek kernel moduljának verifikációja
- The seL4 Microkernel
- Amazon: elosztott rendszerek tervezése

Amazon

Amazon Simple Storage Service

high-performance "no SQL" data store

Elastic Block Store

em	Components	Line Count (Excluding Comments)	Benefit
S3	Fault-tolerant, low-level	804 PlusCal	Found two bugs, then others in proposed optimizations
	of	645 PlusCal	Found one bug, then another in the first proposed fix
DynamoDB	Replication and group-membership system	939 TLA+	Found three bugs requiring traces of up to 35 steps
EBS	Volume management	102 PlusCal	Found three bugs
lock manager	Lock-free data structure	223 PlusCal	Improved confidence though failed to find a liveness bug, as liveness not checked
	Fault-tolerant replication-and-reconfiguration algorithm	318 TLA+	Found one bug and verified an aggressive optimization

Összefoglalás

- Feladatok együttműködésénél sok hibalehetőség
- Versenyhelyzet, holtpont…
- DE: léteznek eszközök a vizsgálathoz
- Modellellenőrzők, tételbizonyítók, statikus ellenőrzők…

További információ

- R. Hamberg and F. Vaandrager. Using Model Checkers in an Introductory Course on Operating Systems. OSR 42(6):101-111.
- Formális módszerek MSc tantárgy (VIMIM100)

További információ

- UPPAAL modellellenőrző
- PetriDotNet modellellenőrző

  - http://petridotnet.inf.mit.bme.hu/publications/
- Theta szoftverellenőrző keretrendszer
  - https://github.com/FTSRG/theta
  - CERN
- Gamma Statechart Composition Framework
  - https://inf.mit.bme.hu/en/gamma
  - Rendszerek modell-alapú tervezése és ellenőrzése

Specializáció

- BSc: Szoftverfejlesztés, Rendszertervezés
- MSc: Kritikus rendszerek, Intelligens rendszerek

- Szoftver és rendszerfejlesztés/ellenőrzés,
- formális módszerek és algoritmusok,
- adat és mesterséges intelligencia alapú rendszerek;

Május 8. 13:00-14:00
Csatlakozás a Teams csoporthoz: **9xmh636**
(lépj be most és kérdezz a specializációkról!)