Beágyazott Rendszerek Modellezése

Számítási Modellek
A Ptolemy modellezési eszköz

Készítette:
Völgyesi Péter és Simon Gyula

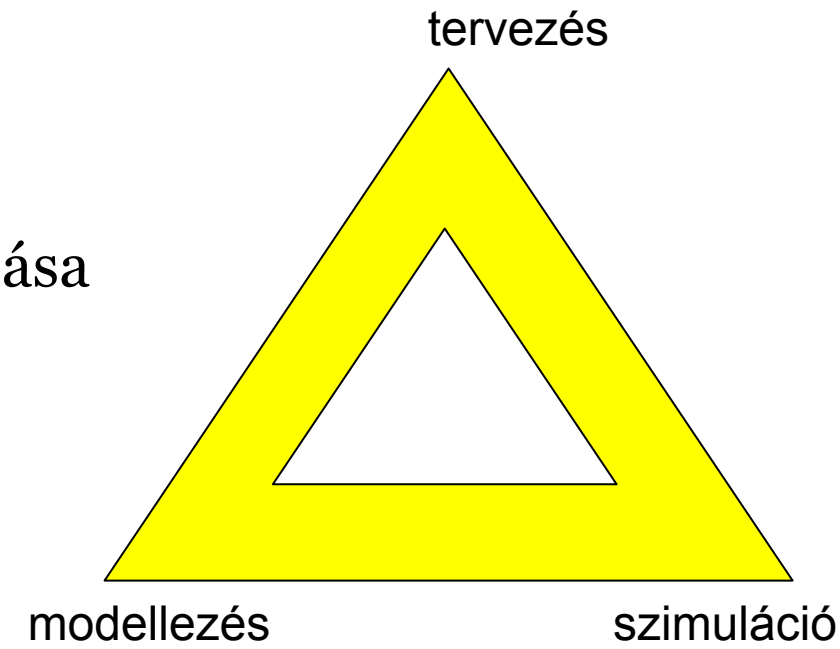
Tartalom

- Tervezés, modellezés, szimuláció
- Számítási modellek – végrehajtási logika
- A Ptolemy eszköz bemutatása
- Számítási modellek bemutatása:
  - Diszkrét események
  - Folytonos idejű rendszerek
  - Szinkron adatfolyam gráfok
  - Állapot automaták
  - Idővezérelt rendszerek
  - Kommunikáló folyamatok
  - Folyamat hálózatok

Tervezés, modellezés, szimuláció

Cél: komplex rendszerek tervezése

- Különböző feladatok keveredése. Pl.:
  - Hálózatkezelés
  - Jelfeldolgozás
  - Szabályozástechnika
  - Működési módok váltakozása
  - Felhasználói felületek



Modellezés

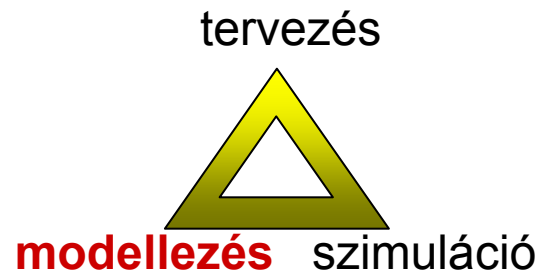
Modellezés = rendszer formális leírása

Matematikai modellek

- Rendszer tulajdonságairól, viselkedéséről szóló állítások halmaza

Konstruktív modellek

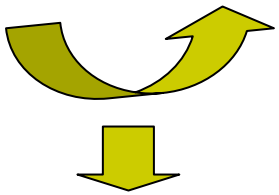
- Számítási egységek, amik utánozzák a rendszer bizonyos aspektusainak viselkedését
- Gerjesztés – válasz
- „végrehajtható” modellek



Tervezés

Tervezés = rendszerkomponensek definiálása

- Modellalkotás
- Modellfinomítás



- Kívánt viselkedés

Modellek:
- Kézben tartható, változtatható (tervezett beágyazott rendszer)
- Külső kényszer által definiált (környezet)

Szimuláció

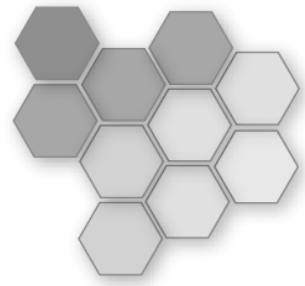
Szimuláció = végrehajtható modell futtatása

- Fizikai rendszer + beágyazott rendszer
- Modell viselkedésének meghatározása
- Modellfinomítás eszköze

- Egyes modellek implementációvá válhatnak

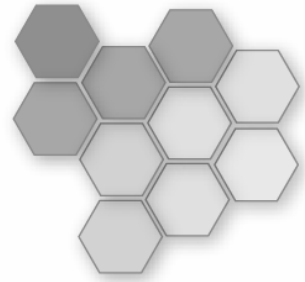
  ⇒ kódgenerálás

Számítási modellek (Models of Computation)



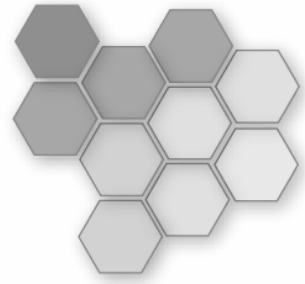
- „Fizikai törvényeket” definiálják
  - Végrehajtás módja (nem a struktúra!)
  - pl. mechanika, állapotgráf, folyamatháló
- Absztrakt szabályrendszer, mely a modellek viselkedését szabályozza: **szemantika**
- Fontos feladatok
  - Párhuzamosság kezelése
  - Idő kezelése
  - Komponensek közötti kommunikáció

Számítási modellek (Models of Computation)



- Modellezendő világhoz *illeszkedő* számítási modell választása
- Milyen a megfelelő számítási modell?
  - Nincsenek felesleges kényszerek
  - Mégis elég kötött, hogy
    - Hasznos eredményeket tudjon adni
    - Effektíven tudjon működni
- Rossz szemantika → költséges, gyenge modell

Számítási modellek (Models of Computation)

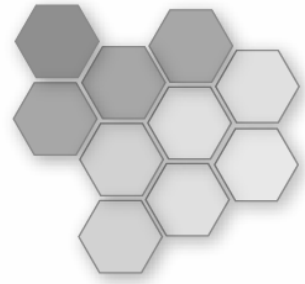


- A fizikai rendszer, a HW és a SW együttes modellezése
- Több számítási modell
  - Mindig olyat választunk, amely a részfeladathoz megfelelő
- Különböző számítási modell *együttes* használata
  - Pl. a fizikai rendszer és a beavatkozó logika modellezése
- Jól definiált számítási modellek (szemantika)
  - Közös nyelvek, mindenki érti a modellt, a végrehajtás menete egységes

Ptolemy http://ptolemy.eecs.berkeley.edu

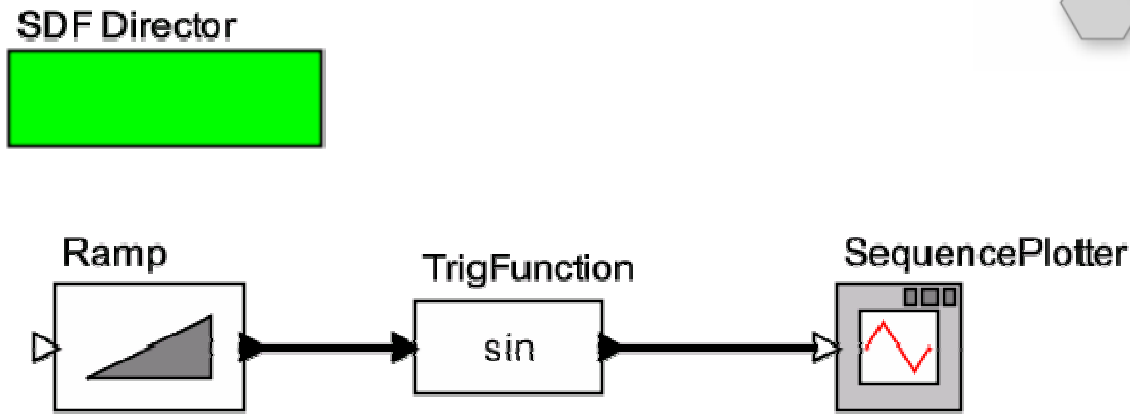
- *A major emphasis in Ptolemy II is on the methodology for defining and producing embedded software together with the systems within which it is embedded.*
- *A principle of the Ptolemy project is that the choices of models of computation strongly affect the quality of a system design.*

Ptolemy

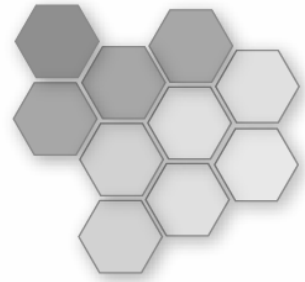


- Grafikus modellezési eszköz
- Számítási modellek és azok együttes használata: heterogén modellek
- Hangsúly a szemantikai különbségeken van
  - Meta-modellek nincsenek
- Szimuláció (Java)
- Korlátozott mértékben kódgenerálás

Ptolemy

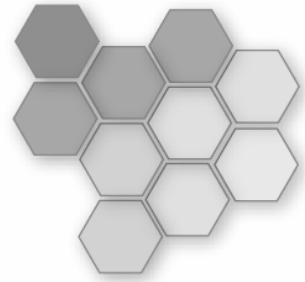


- Director: a modell szemantikai értelmezése
- Actor: végrehatható elemek
  - Általános és szemantika függő elemek
- Portok (kapuk): kommunikációs pontok (input/output)
- Kapcsolatok: kommunikációs csatornák
- Hierarchia: összetett actor-ok
  - Heterogén modellek: az összetett actor különböző vezérlőt (director) tartalmaz



Bemutató: sin.moml

Számítási modellek - különbségek



- Párhuzamos végrehajtás
  - Milyen mértékben „szakadnak el” egymástól a végrehajtó egységek
- Determinisztikus viselkedés
  - Pontosan megismételhető-e a szimuláció
- Az idő fogalma
  - Foglalkozik-e a szimuláció idő jellegű információval
  - Ha igen, hogy viszonyul a szimulációs idő a valós időhöz
  - Elosztott (lokális órák) vagy globális idő
- Speciális végrehajtó egységek

Folytonos idejű rendszerek [Continuous Time]

- Elsősorban a fizikai környezet leírására használjuk
- A rendszer kimenetén és bemenetén folytonos idejű jelek, differenciál-egyenletek
- Matematikai modell:

$$\dot{x} = f(x,u,t)$$

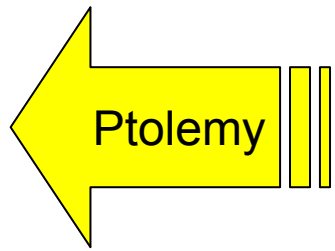
$$y = g(x,u,t)$$

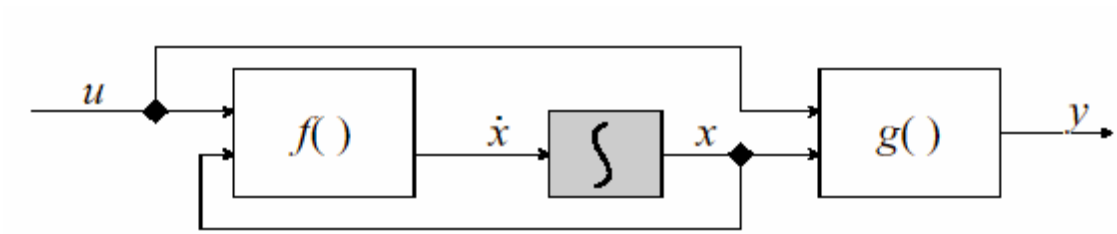
$$x(t_0) = x_0$$

Folytonos idejű rendszerek [Continuous Time]

Modellezési lehetőségek:

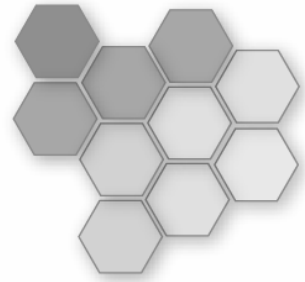
- Fizikai modell (megmaradási törvények)
  Egy meglévő rendszerből könnyű előállítani
  A matematikai modell nem látszik közvetlenül
- Jelfolyam gráfok
  Absztraktabb leírás
  Könnyebben kapcsolható más leírási formákhoz
  A matematikai leírás közvetlenül előállítható





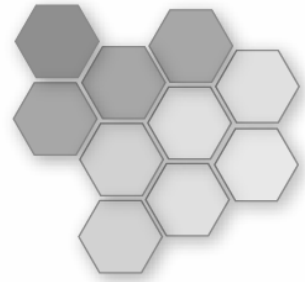
$$\dot{x} = f(x,u,t)$$
$$y = g(x,u,t)$$
$$x(t_0) = x_0$$

Folytonos idejű rendszerek [Continuous Time]



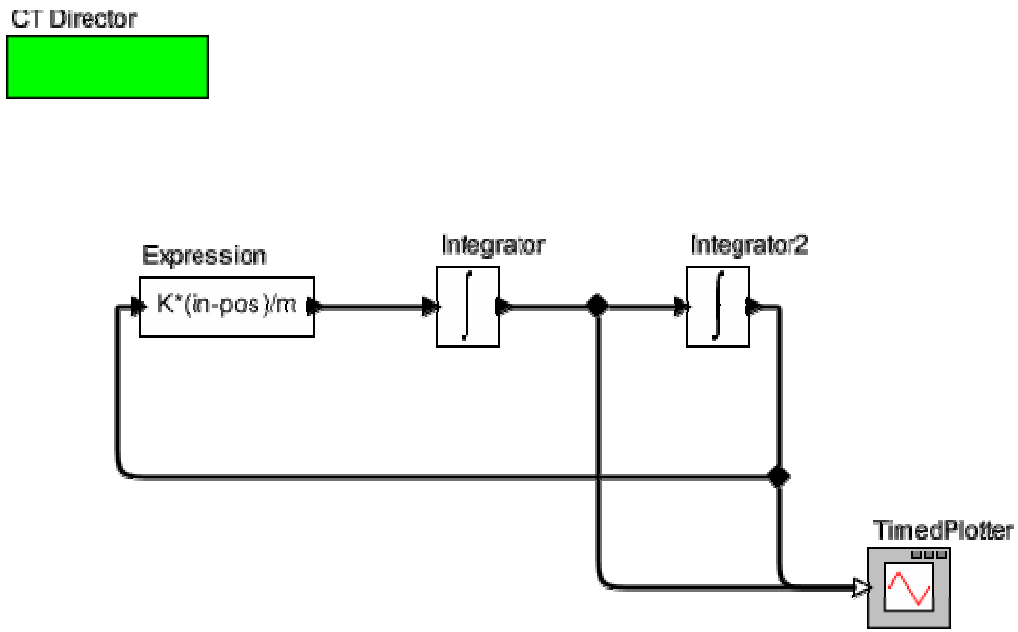
- Végrehajtás, szimuláció:
  - Numerikus DE megoldó algoritmusok (Euler, Runge-Kutta, trapéz módszer)
  - Előre menetelés az időben, differencia egyenletekkel való közelítés
  - A lépés nagysága (Δt) rögzített vagy adaptív (speciális actor-ok képesek korrigálni)
  - A kapcsolatokon a jelek lépcsősek (nulladrendű tartó)

Folytonos idejű rendszerek [Continuous Time]

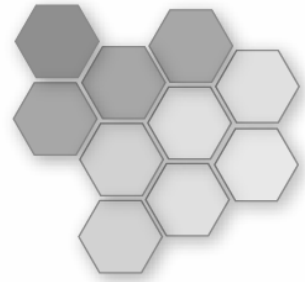


- Idő kezelése:
  - Valós idejű végrehajtás, folytonos idő
  - Közös óra
- Determinisztikus viselkedés
- Speciális végrehajtó egységek:
  - Integrátor
  - Esemény generátorok (pl. nullátmenet érzékelő)
  - Jelgenerátorok (pl. nulladrendű tartó)

Bemutató: spring.moml

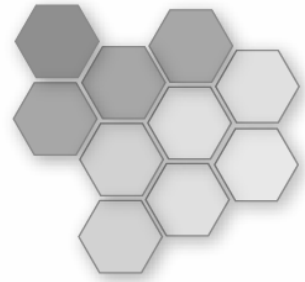


Diszkrét események [Discrete Events]



- A végrehajtó egységek időbélyeggel ellátott eseményeket (token) generálnak
- Az események egy közös rendezett listába kerülnek
- A vezérlő (director) lépteti az időt a listában szereplő első token időbélyegére
- Egy iterációs lépésben az összes olyan esemény feldolgozásra kerül, melynek közös az időbélyege (egy végrehajtó egység többször is meghívódhat)

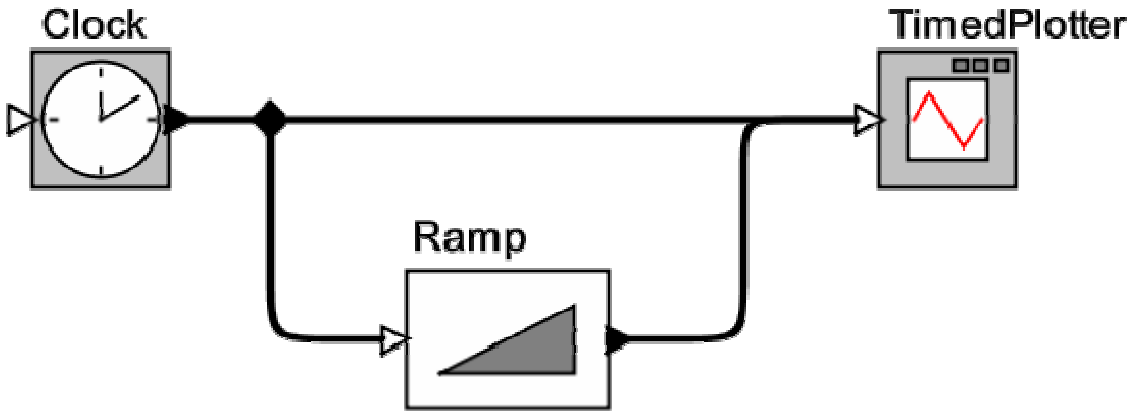
Diszkrét események [Discrete Events]



- Idő kezelése:
  - Globális
  - Nem folytonos (a szimuláció az események számával arányos ideig tart, a szimulált idő „nem számít”)
  - Nem valós idejű
- Párhuzamosítás foka alacsony (az ütemezési logika szekvenciális)
- Determinisztikus viselkedés
- Speciális végrehajtó egységek: késleltető logika
  - Az általános actor-ok késleltetése: 0

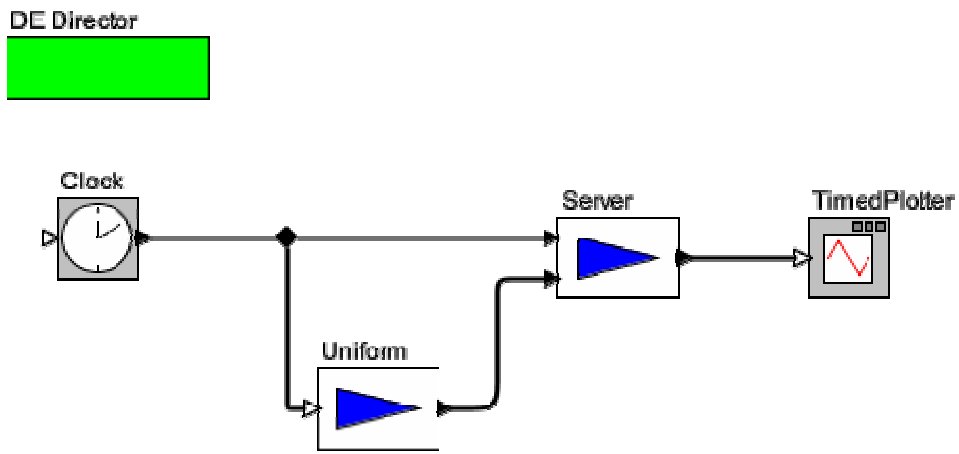
Diszkrét események [Discrete Events]

- Szimultán események végrehajtása
- Intuíció: a „Ramp” végrehajtó egység hamarabb fog lefutni, mint a „Plotter”
- Megoldás: topológiai rendezés a szimuláció elején

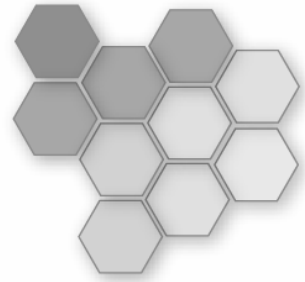


- Probléma: köröket tartalmazó gráfok ⇨ ilyenkor mindig kell késleltetés

Bemutató: server.moml



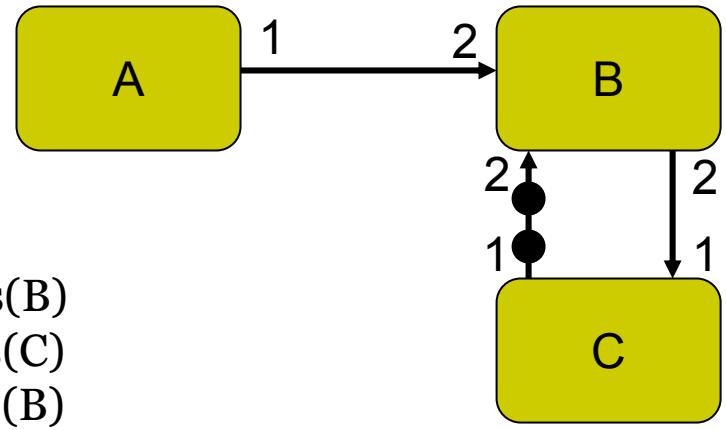
Szinkron adatfolyam gráfok [Synchronous Dataflow]



- Jelfeldolgozó feladatoknál használjuk
- Az egyes végrehajtó elemek minden iterációs lépésben token(eke)t fogyasztanak állítanak elő
  (homogén: egy token/iteráció, nem homogén: több is lehet)
- Minden actor (konstans számszor) fut az iterációs lépésben
  (homogén: egyszer, nem homogén: többször is)
- A végrehajtási sorrend előre eldönthető (topológiai viszonyok alapján), deadlock felismerhető
- Visszacsatolás csak késleltetővel oldható meg. (A késleltetés nem időt, hanem iterációs lépést jelent)

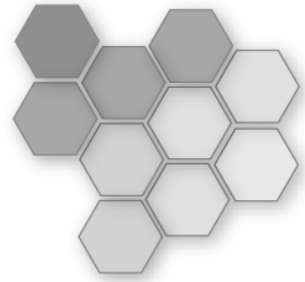
Szinkron adatfolyam gráfok [Synchronous Dataflow]

- Nem homogén adatfolyam gráf: egy végrehajtó egység több tokent fogyaszt vagy állít elő a portjain
- Végrehajtási sorrend:
  A, A, B, C, C
- Token-megmaradási egyenletek:
  Fire(A) x Prod (A) = Fire(B) x Cons(B)
  Fire(B) x Prod (B) = Fire(C) x Cons(C)
  Fire(C) x Prod (C) = Fire(B) x Cons(B)



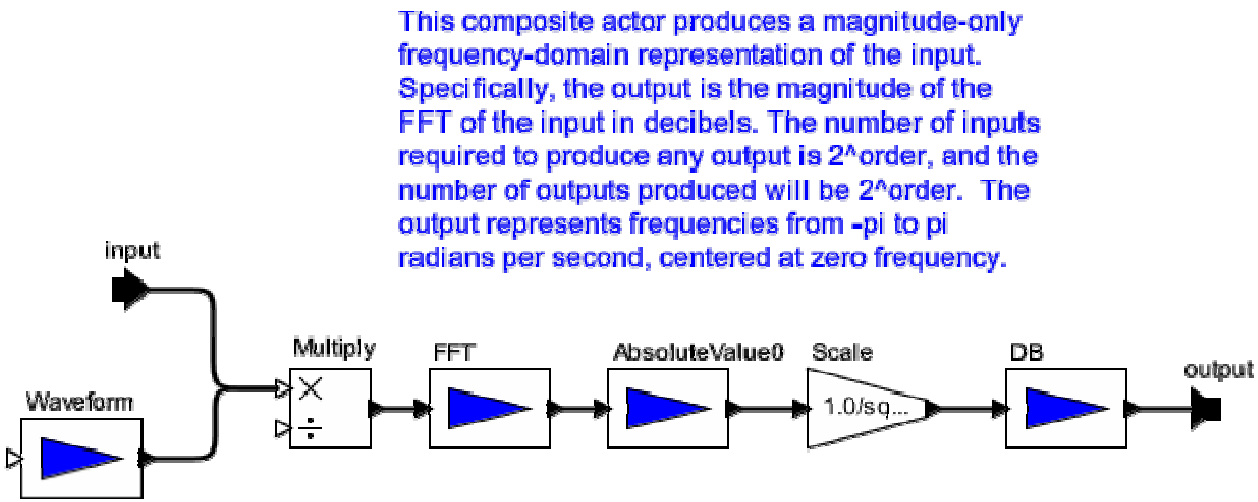
- Az iterációs lépés végén minden kapcsolaton annyi token áll, mint a lépés végrehajtása előtt
- Példa: FFT algoritmus

Szinkron adatfolyam gráfok [Synchronous Dataflow]

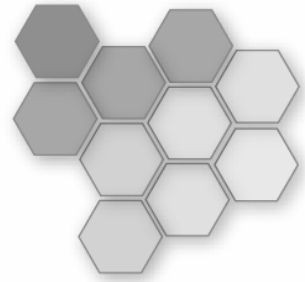


- A szinkron adatfolyam gráf nem kezeli az idő fogalmát (helyette iterációs lépések vannak)
- A viselkedés determinisztikus (előre elkészített ütemezés) → gyors!
- Párhuzamosság foka alacsony (erős függőség a többi végrehajtó egységtől)
- Speciális elemek: minden olyan elem, mely több tokent gyárt vagy fogad (pl. FFT)

Bemutató: spectrum.moml



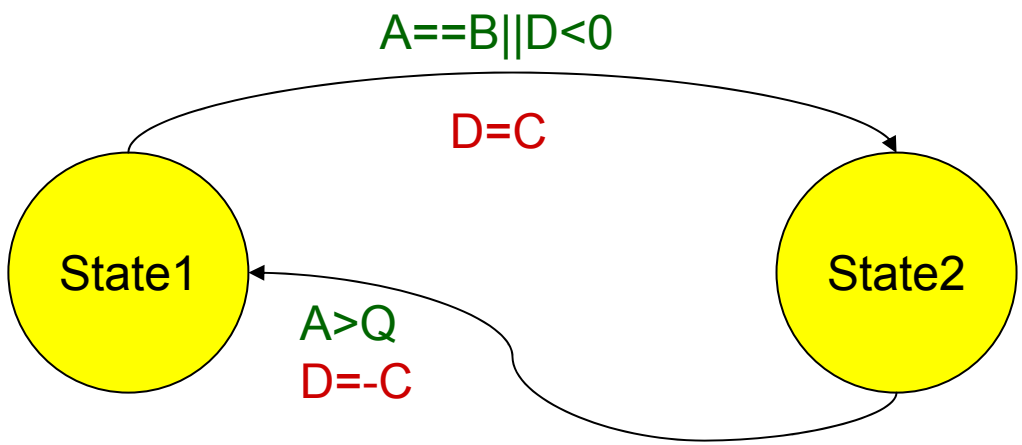
Állapotautomaták [Finite State Machines]



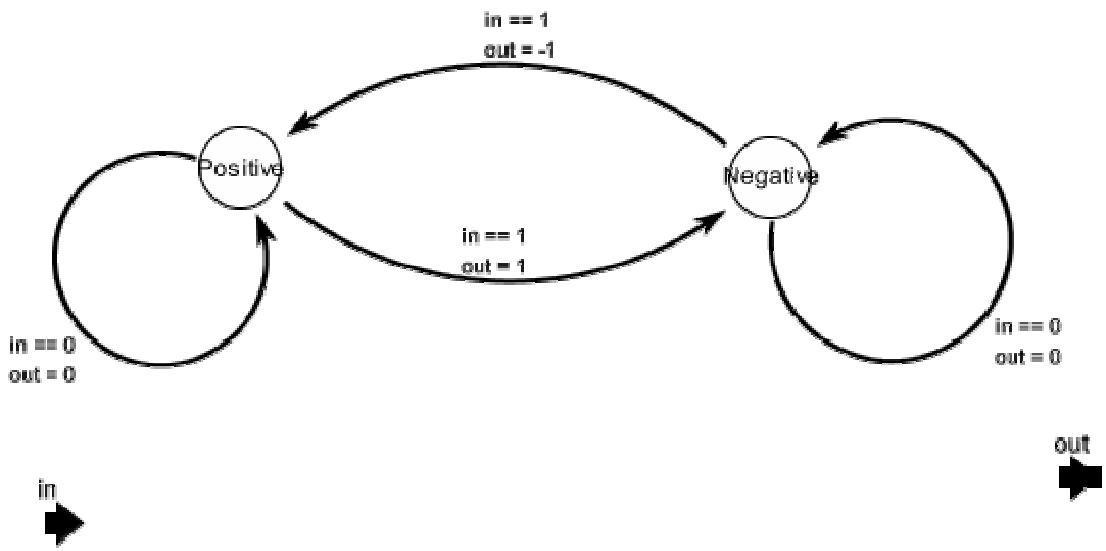
- Kakukktojás: a dobozok nem kommunikálnak, hanem állapotokat írnak le
- Nagyon jól használható szekvenciális logikai műveletek leírására (pl. szoftverkomponensek, szabályzók)
  - Intuitív
  - Formális analízis és verifikáció
- Hagyományos állapotautomata kiterjesztése (*charts):
  - Hierarchia
  - Felsőbb szinteken párhuzamosítás (pl. adatfolyam gráf)

Állapotautomaták [Finite State Machines]

- Állapotok (state)
- Állapot-átmenetek
  - Feltételek (guard-expressions)
  - Műveletek (actions)



Bemutató: ami_coder.moml (Alternate Mark Inversion Coder)

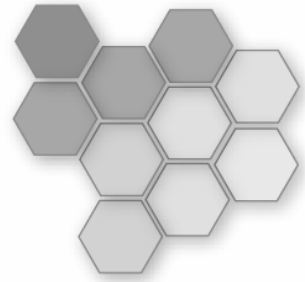


Idővezérelt rendszerek [Time Triggered Systems]

GIOTT 

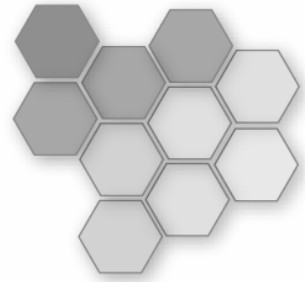
- Kommunikáció periodikus, idő-triggerelt (TT) modulok között
- Biztonságkritikus alkalmazások, RT rendszerek

Idővezérelt rendszerek [Time Triggered Systems]



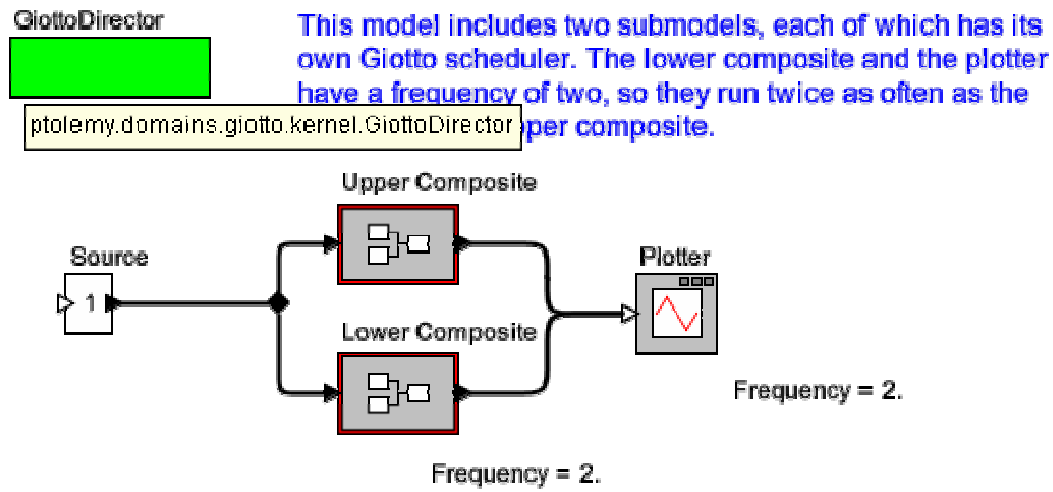
- Jelentős különbség: a végrehajtó elemeket nem beérkező tokenek, hanem időzítő hajtja végre (adott frekvenciával)
- Minden végrehajtási elemnek létezik egy „worst case execution time” paramétere: az ütemező előre felismerheti a hibákat
- A végrehajtó elem által előállított tokenek csak a végrehajtás után érhetőek el más elemek bemenetein (pl. ugyanolyan frekvenciával működő összekötött komponensek közül a második egy ciklussal későbbi tokeneket lát)

Idővezérelt rendszerek [Time Triggered Systems]

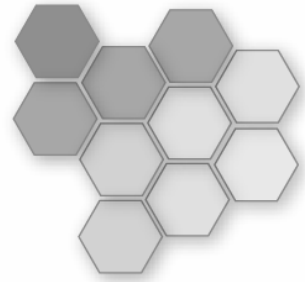


- Idő kezelése:
  - „Erősen” valós idejű
  - Globális óra
- Párhuzamos végrehajtás (nincs adat jellegű függés)
- A viselkedés determinisztikus
- Tipikus felhasználás: real-time rendszerek modellezése (ipari folyamatok)

Bemutató: multirate.moml

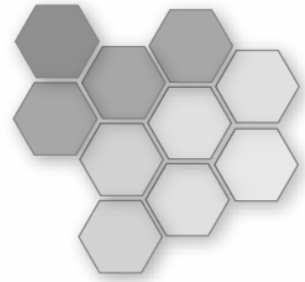


Kommunikáló folyamatok
[Communicating Sequential Processes]



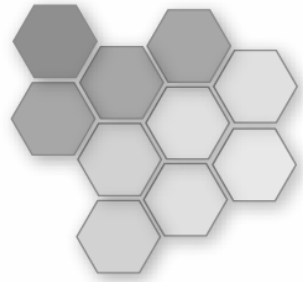
- Külső kontroll nélkül futó folyamatok
- Egyirányú kommunikációs csatornák a folyamatok között – FIFO nélkül
- Blokkoló „olvasás” és „írás” műveletek (randevú)
- Az adatküldés atomi módon zajlik le, mely után a két folyamat tovább fut
- Felhasználás: erőforrás menedzsment

Kommunikáló folyamatok
[Communicating Sequential Processes]



- Az eredeti modell az idő fogalmát nem kezeli, de kiterjeszthető
- Magas szintű párhozamosság
- Deadlock felismerése
  - minden folyamat blokkolva van
  - csak futás időben ismerhető fel
- Nem determinisztikus viselkedés
  - több párhuzamos csatorna egyidejű használata
  - Az első kommunikáció blokkol

Kommunikáló folyamatok [Communicating Sequential Processes]



Példa: vacsorázó filozófusok

http://ptolemy.eecs.berkeley.edu/ptolemyII/ptIIlatest/ptII/ptolemy/domains/csp/doc/

Pálcika életciklus:

Asztalon fekszik
Felveszik
Esznek vele
Leteszik
Asztalon fekszik...

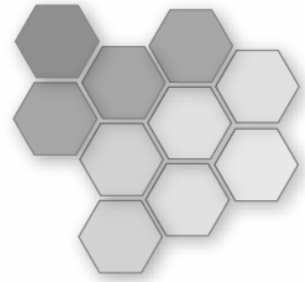
Filozófus életciklus:

Meditál
Megéhezik
Felveszi az egyik pálcikát
Felveszi a másik pálcikát
Eszik
Leteszi a pálcikákat
Meditál ...

Pálcika életciklus:

Asztalon fekszik
Felveszik
Esznek vele
Leteszik
Asztalon fekszik
Felveszik
Esznek vele
Leteszik
Asztalon fekszik...

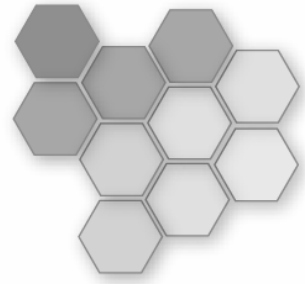
Folyamathálózatok [Process Networks]



- Aszinkron adatfolyam gráfok
- A végrehajtó elemek FIFO-val bufferelt csatornán keresztül kommunikálnak
  - az írás (küldés) művelet nem blokkolódik
  - üres buffer olvasása blokkol

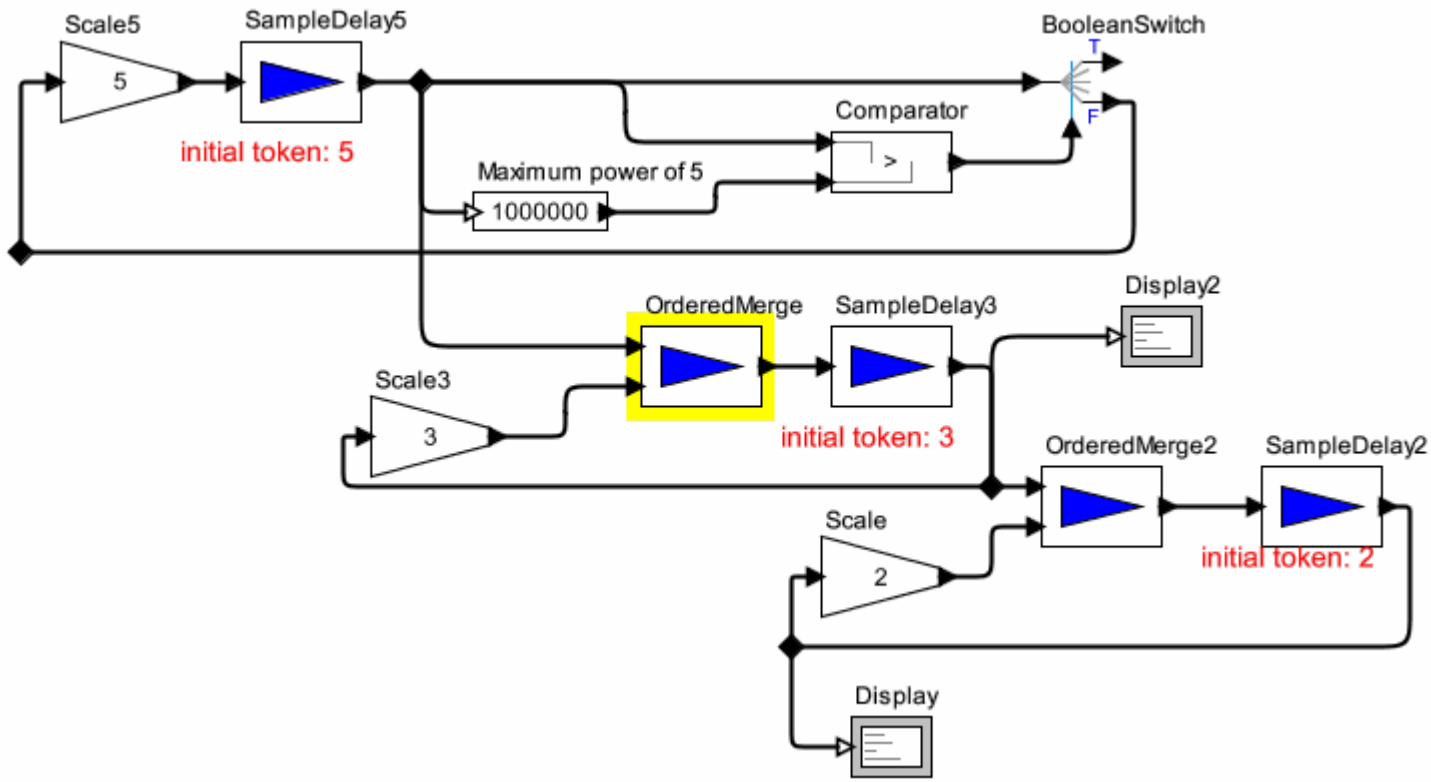
  aszinkron üzenetküldés, Kahn process network
- Tipikus felhasználás: jelfeldolgozás
- Előre nem kitalálható végrehajtási sorrend (ütemezés)
- Deadlock felismerése csak futás alatt történhet

Folyamathálózatok [Process Networks]

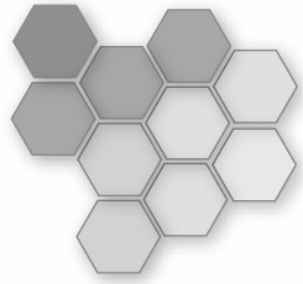


- Az eredeti modell az idő fogalmát nem kezeli, de kiterjeszthető
- Nagyon magas szintű párhozamosság (csak az olvasás blokkolhat)
- Determinisztikus viselkedés
  - Kivétel: mutációk időfogalommal nem rendelkező PN-ben

Bemutató: OrderedMerge.moml



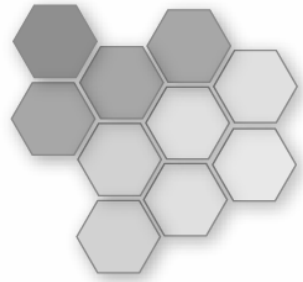
Heterogén rendszerek: kevert (folytonos – diszkrét) jelek



- Tipikusan minden beágyazott alkalmazásban jelentkezik: folytonos fizikai rendszer – diszkrét idejű számítógép

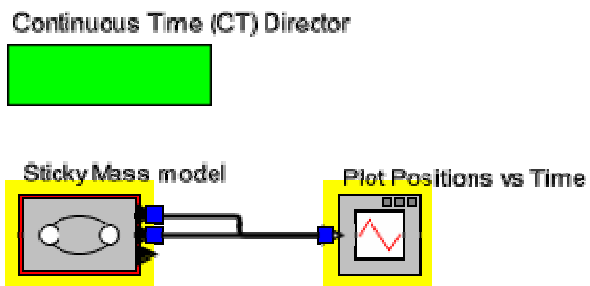
Bemutató: mixed.moml

Hibrid rendszerek



- A beágyazott rendszer bizonyos feltételek teljesülése esetén egy más viselkedési módba kapcsol át: átkonfigurálás
- FSM állapotok belsejében szerepelnek a különböző működési módok
- Nehéz feladat: tranziensek kezelése

Bemutató: StickyMasses.xml



Vezeték nélküli szenzorhálózatok [Wireless]

Szenzorhálózatok modellezése

- Csatornák:
  - Rádiós kommunikáció
    - Modellek:
      - Kör
      - Veszteséges
  - Egyéb jelenségek
    - Akusztikus

Bemutató:
SmallWorld.xml
EvaderAndPursuer.xml