

**Rendszermodellezés**

kidolgozott jegyzet

**2018. tavasz**

**Összeállította: Csia Kitti**

Tartalomjegyzék

[1. előadás: Modellezési alapismeretek 2](#_Toc515033476)

[2. előadás: Strukturális modellezés 6](#_Toc515033477)

[3. előadás: Állapot alapú modellezés 12](#_Toc515033478)

[4. előadás: Folyamatmodellezés 20](#_Toc515033479)

[5. előadás: Modellek ellenőrzése 28](#_Toc515033480)

[6. előadás: Vizuális adatelemzés 38](#_Toc515033481)

[7-8. előadás: Teljesítménymodellezés 48](#_Toc515033482)

[9. előadás: Modellező eszközök, kódgenerálás 60](#_Toc515033483)

[10. előadás: Modellek paraméterezése: regresszió, benchmarkok 67](#_Toc515033484)

*Felhasznált irodalom:*

**A rendszermodellezés nevű tantárgy hivatalos oldalán (**[**https://inf.mit.bme.hu/edu/courses/remo/materials**](https://inf.mit.bme.hu/edu/courses/remo/materials)**) található aktuális előadások.**

**A tanszék által kiadott, felhasználható jegyzetek (**[**http://docs.inf.mit.bme.hu/remo-jegyzet/**](http://docs.inf.mit.bme.hu/remo-jegyzet/)**).**

**Az előadásokon való részvételem alatt készült saját jegyzetek.**

# **1. előadás: Modellezési alapismeretek**



[**BACK**](#_top)

1. **Modell**
   * Definíció
     + egy valós vagy *hipotetikus* világ egy részének **egyszerűsített** képe
       - bonyolult rendszer egyszerűsített, kisebb, áttekinthetőbb, véges képe
     + rendszert helyettesíti
     + modell egy kérdés megválaszolására épül
       - probléma szempontjából lényeges szempontok kiemelése
     + modell nem valóság!
     + modell és a modellezett rendszer viszonya
       - modell egyszerűsítheti, összemoshatja, elhagyhatja a modellezett rendszer bizonyos részleteit, részeit
   * Példa
     + modellvasút
     + matematika → gráfok (*úthálózat – leggyorsabb út keresés*)
2. **Diagram**
   * Definíció
     + modell egy nézete, amely annak bizonyos aspektusait grafikusan ábrázolja
       - tehát a diagram csak egy ábrázolási módja a modellnek, amivel olvashatóvá tesszük, szövegessé
3. **Modellezési nyelvek**
   * Célja: kommunikáció gép-gép, ember-gép, ember-ember között
   * Definíció
     + modellezési nyelv négy része:
       - ***absztrakt szintaxis/metamodell***
         * meghatározza nyelv elemeinek típusait, viszonyait
         * pl. gépek a modell belső tárolására
       - ***konkrét szintaxis***
         * szöveges/grafikus jelölésrendszert definiál elemtípusokhoz, kapcsolatokhoz
       - ***jólformáltsági kényszer***
         * modellnek milyen követelményeknek kell megfelelnie
         * tovább szűri a lehetséges modellek körét

pl. azonos nevű elemek tiltása

* + - * ***szemantika***
        + absztrakt szintaxis által megadott nyelvi elemek jelentését definiáló szabályrendszer

megadja, hogy pontosan hogy működik, mit jelent a modell

jelentést ad a nyelvi elemeknek

* + Példa
    - szöveges: Verilog, VHDL, Java…
      * előny: könnyebb modellt építeni
    - grafikus: 3D tervezők, UML diagramok…
      * előny: könnyebb olvasni

1. **Nyílt és zárt világ feltételezése**
   * Definíció
     + ***zárt világ feltételezése***: minden állítás, amiről nem ismert, hogy igaz, hamis
     + ***nyílt világ feltételezése***: egy állítás annak ellenére is lehet igaz, hogy ez a tény nem ismert
     + különbség: nyílt világ elismeri, és használja az *ismeretlen* fogalmát, zárt világ minden tudást ismertnek tekint
   * Példa
     + zárt világ
       - olyankor alkalmazzuk, mikor a rendszernek minden szükséges információ a rendelkezésére áll
       - metrójáratok - tudjuk, hogy az adott helyen van-e megálló vagy nincs
       - minden modell zárt világ!
     + nyílt világ
       - amikor nem feltételezzük, hogy minden információ a rendelkezésünkre áll
       - orvosi nyilvántartás – beteg allergiás valamire, annak ellenére, hogy nincs a nyilvántartásában
2. **Rendszer és környezete**
   * Definíció
     + ***rendszer***
       - egyértelműen definiálja a határait
       - határon belül eső dolgok
     + ***környezet/kontextus***
       - rendszerre ható tényezők összesége
       - határon kívül eső dolgok
       - *releváns környezeti elemek*
         * rendszerrel közvetve vagy közvetett módon kapcsolatban áll
       - *irreleváns környezeti elemek*
         * nincs kapcsolatban a rendszerrel
       - *fekete doboz*: teszteléskor a rendszer belső felépítését és viselkedését nem ismerjük
       - *fehér doboz*: ismerjük
3. **Absztrakció és finomítás**
   * Definíció
     + ***finomítás***
       - modell részletezése/ pontosítása
       - környezet szempontjából akár helyettesíteni is tudja az eredeti modellt
       - több végkimenetelt eredményezhet
     + ***absztrakció***
       - finomítás inverz művelete
       - modell részletezettségének csökkentése
       - modellezett ismeretek egyszerűsítése
       - finomítás után az eredeti modell absztrakcióval kapható vissza
       - egy végkimenetelt eredményez
   * Példa
     + közlekedési lámpa
       - absztrakt modell: *tilos* és *szabad*
       - finomított modell: *szabad – zöld*, *tilos – piros, kettő közötti – sárga*

# **2. előadás: Strukturális modellezés**



[**BACK**](#_top)

1. **Strukturális modell**
   * Definíció
     + a rendszer felépítésére vonatkozó tudás
     + a rendszer felépítését reprezentálja
       - alkotórészei
       - azok tulajdonságai
       - egymással való viszonya alapján
     + *statikus*, tehát:
       - változhat az idő során – metróhálózat fejlődése
       - de időben nem ír le változásokat – miként mozognak a szerelvények
     + kiindulópontja egy (rész)rendszer, amelyet a része *reláció* mentén alkotórészekre bont, ezek lehetnek
       - további *részrendszer*ek
       - tovább nem bontott (*elemi*) *komponens*ek
   * Célja:
     + rendszer részekre bontása
       - kisebb egységeket könnyebb tervezni
       - részegységek újrahasználhatósága
       - általános célú komponensek használata
     + létező rendszer dokumentálása – „*rendszertérkép*”
     + adatszerkezet megalkotása – milyen információt kezel
2. **Tulajdonságmodell**
   * Definíció
     + ***jellemző***
       - modell által megadott *parciális függvény*
       - modellelemeken értelmezünk
   * Példa
     + *kapacitás* jellemzőhöz tartozó függvény

: modellelem azonosítója

: nemnegatív egész szám

pl.:

* + - *funkció* jellemzőhöz tartozó függvény

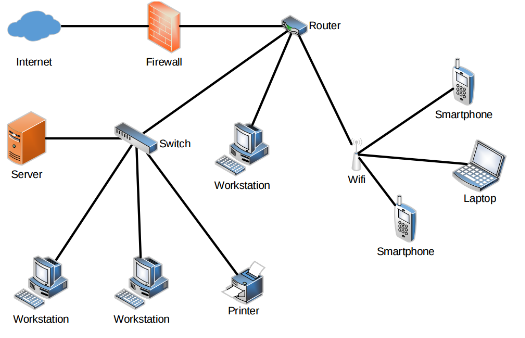
: modellelem azonosítója

: egy {*metró járműtelep, autóbuszgarázs*} halmaz eleme

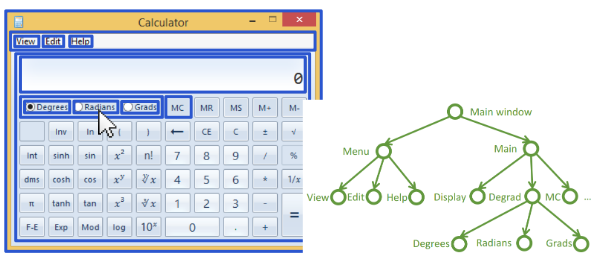
pl.:

* + - fentiekhez hasonlóan bevezetünk egy *vágányhossz* jellemzőhöz tartozó parciális függvényt, csak bizonyos modellekre értelmezhetünk
    - csak akkor vesz fel értéket, ha a funkció attribútum értéke: *metró járműtelep*
    - tehát a *vágányhossz* jellemző csak a *metró járműtelep* típusú elemekre értelmezett
    - ***típus***
      * meghatározza, milyen más jellemzők értelmezettek az adott modellelemre
      * milyen más modellekkel lehet kapcsolatban
      * többi jellemző: *tulajdonság*
    - ***példány***
      * egy adott típus példányainak nevezzük azon modellelemeket, amelyek típusa

1. **Gráfmodell**
   * Definíció
     + ***gráf***
       - egy gráf rendezett pár,
       - nem üres halmaz
         * elemei: pontok/csúcsok
         * csúcsok száma:
         * csúcshalmaz jelölése:
       - -ből képezhető párok egy halmaza
         * elemei: élek
         * élek száma:
         * élhalmaz jelölése:
     + ***típusgráf***
       - egy olyan gráf, amelyben minden csomóponthoz egy típuscsomópont, minden éltípushoz egy típusél tartozik
     + ***példánygráf***
       - elemei a típusgráf csomópont – és éltípusainak példányai
       - minden él forrása és célja rendre az éltípus forrásának és céljának példánya
       - rendszert alkotó elemek egymáshoz való viszonyának leírása
2. **Hierarchia**
   * Definíció
     + ***dekompozíció (faktoring)***
       - egy rendszer kisebb komponensekre bontása
       - azok könnyebben érthetők, fejleszthetők, karbantarthatók
       - *a rendszer hierarchiája a rendszer dekompozíciójával állítható elő*
       - részrendszer dekompozícióval további részekre bontható
       - *helyes dekompozíció*
         * ha a dekompozícióval kapott rendszer minden elemének megfeleltethető az eredeti rendszer valamelyik eleme
         * az eredeti rendszer minden eleméhez hozzárendelhető a dekompozícióval kapott rendszer egy vagy több eleme
3. **Nézetek**
   * strukturamodellekből különböző nézeteket állíthatunk elő
   * tulajdonságmodelleken leggyakrabban használt műveletek:
   * Definíció
     + ***szűrés***
       - a modell elemein kiértékelünk egy feltételt
       - azokat tartjuk meg, amelyek megfelelnek a feltételnek
         * tulajdonságmodell esetén a szűrés az elhagyott modellelemek, modell sorai
         * gráfnál a gráf csúcsai, élei
   * Példa
     + *tulajdonságmodell szűrése*
       - szeretnénk megtudni, hogy mely telephely képes legalább 100 jármű befogadására
     + *gráfmodell szűrése*
       - soroljuk fel az M2-es metró megállóit
   * Definíció
     + ***vetítés***
       - a modell egyes jellemzőit kiválasztjuk és a többit elhagyjuk
       - az is érvényes, ha az összes elemet megtartjuk
   * Példa
     + olyan kimutatást adjunk, amely csak a budapesti autók színét tartalmazza
4. **Strukturális modellezési technikák**
   * ***hierarchia modellezés***
     + (*gráfoknál*) két részre tagolódik
       - modell *szerkezeti váz*át tartalmazó hierarchia (fa/erdő)
         * alkotóelemek rész-egész viszonyát mutatja
         * élcímekre szűrve is egy vagy több fa marad
       - *kereszthivatkozás* élek
         * *tartalmazási rend*től függetlenül, körmentesség korlátozása nélkül köthetnek össze elemeket
     + metamodell megmondhatja, mely éltípusok példányait fogjuk a szerkezeti vázat alkotó *tartalmazási él*eknek tekinteni
   * elkészítési sorrend/megközelítés
   * Definíció
     + ***top-down***
       - a modellezés során fentről lefelé (összetett rendszertől az alkotóelemig) haladva építjük
       - alaplépése a dekompozíció
     + fontosabb jellemzők:
       - részrendszer tervezésekor a szerepe már ismert
       - „félidőben” még nincsenek teljesen működő részek
       - részek problémái, igényei később derülnek ki
     + ***bottom-up***
       - alulról felfelé haladva (elszigetelt alkotóelemekből az összetett rendszer felé) építjük
       - alaplépése a kompozíció
       - egész rendszer összeszerkesztése külön modellezett vagy fejlesztett részrendszerekből
     + fontosabb jellemzők:
       - rendszer részei önmagukban kipróbálhatók, tesztelhetők
       - részleges készültségnél könnyebben előállítható a rendszer prototípusa
       - nem látszik előre a rész szerepe az egészben
   * van „aranyközépút” is a kettő között
   * Példa
     + számítógép hálózat



* + - grafikus felhasználói felület (GUI) is hierarchikus modell



# **3. előadás: Állapot alapú modellezés**



[**BACK**](#_top)

1. **Egyszerű állapotgépek**
   * Definíció
     + ***állapottér***
       - egymástól megkülönböztetett rendszerállapotok halmaza
         * nem minden állapothalmaz lehet állapottér
       - minden időpontban pontosan egy eleme jellemzi a rendszert (*pillanatnyi állapot)*
       - elemeit állapotoknak nevezzük, két kritériumnak kell megfelelniük – **teljesség, kizárólagosság**
       - példa:
     + ***teljesség***
       - minden időpontban az állapottér legalább egy eleme jellemzi a rendszert
       - példa: nem teljes
     + ***kölcsönös kizárólagosság***
       - minden időpontban az állapottér legfeljebb egy eleme jellemzi a rendszert
       - ellenpéldák nem alkalmasak állapottérnek
       - példa: nem kizárólagos holott teljes
     + *rendszer pillanatnyi állapota*
       - állapottér egyetlen eleme, amelyik abban az időpontban jellemző a rendszerre
       - példa: ma van
     + *rendszer kezdő állapota*: olyan állapot, amely a vizsgálatunk kezdetekor (pl. ) pillanatnyi állapot lehet
     + ***események***
       - *esemény*
         * pillanatszerű változás (a rendszerben vagy in/outputon)
         * példa: egyszerre
       - *eseményfolyam*
         * pl. in/output adatforrás
         * példa: telefon értesítés jelzése
       - *eseménytér*
         * megengedett események
         * beolvasható input értékek
         * kibocsátható output értékek
         * példa:
     + ***állapotfinomítás és -absztrakció***
       - az állapottéren mint halmazon végzett halmazfinomítás illetve halmazabsztrakció
       - eredménye egy újabb állapottér
       - *állapotfinomítás*
         * tervezés előrehaladása, több megvalósítási részlet
         * specializáció, kiegészítés
         * több rendszer együttes vizsgálata
         * több információ
       - *állapotabsztrakció*
         * hasznos, ha az absztrakt állapotok

egységesek, majdnem ekvivalensek

valamilyen szempontból egyformák az összevont állapotok

* + - * + bizonyos feladatokra kevesebb információ is elég

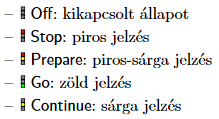
kisebb, egyszerűbb állapottérrel könnyebb tervezni

tárolás, feldolgozás könnyebb

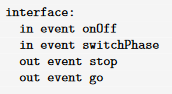
elrejtett részletek szabadon változtathatóak

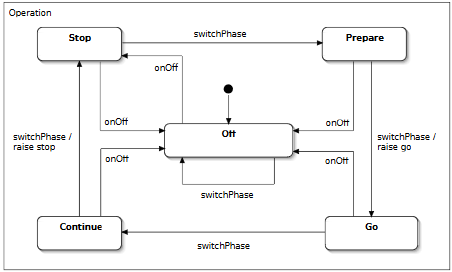
szélsőséges eset: *állapotmentes modell*

* + - * + gyakori formája: dekompozíció
        + kevesebb információ
    - ***állapottér vetítése komponensre***
      * állapotabsztrakciós művelet
      * szorzat állapotteréből egy vagy több komponenst tart meg
      * többit elhanyagolja
    - ***diszkrét állapottér***
      * nem létezik folytonos átmenet
      * a rendszer pillanatnyi állapota mindaddig állandó, amíg egy pillanatszerű *esemény* hatására másik állapotba kerül
      * *állapotátmenetekkel/tranzíciókkal* jól modellezhető
        + megengedi, hogy a rendszer állapotot váltson forrás – és célállapot között
    - ***tüzelés***
      * rendszer új állapota a célállapot lesz
      * → egy adott *esemény* válthatja ki
        + speciális esete a *spontán állapotátmenet*, mikor a kiváltó esemény kívülről nem megfigyelhető
      * vagy *akció*, amely maguk is válthatnak ki eseményeket
  + „állapot” két jelentéssel bír
    - *szintaktikai*: állapotgráf egy csomópontja, melyet lekerekített téglalap jelöl (*állapotcsomópont*)
    - *szemantikai*: állapottér egy eleme
  + Példa
    - jelzőlámpa egyszerű állapottere:



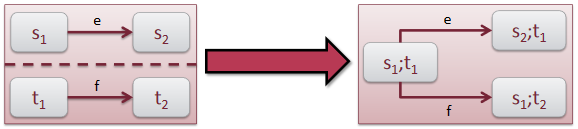
* + - jelzőlámpa modellje:
    - onOff: ki – és bekapcsolás kérése
    - swithcPhase: jelzésváltás kérése
    - stop: rendszer sárgából vörösbe váltott
    - go: rendszer zöldbe váltott





* + modell tulajdonságai
  + Definíció
    - ***determinisztikus***:
      * állapotgépnek legfeljebb egy kezdőállapota van
      * bármely állapotban, bármely bemeneti esemény bekövetkezésekor legfeljebb egy tranzíció tüzelhet
      * egy állapotgép biztos determinisztikus, ha egy állapotátmenete van
    - ***teljesen specifikált***:
      * állapotgépnek legalább egy kezdőállapota van
      * bármely állapotban, bármely bemeneti esemény bekövetkezésekor legalább egy tranzíció tüzelhet
      * minden inputra van szabály
  + **rendszer időbeli viselkedése**
  + Definíció
    - ***végrehajtási szekvencia***:
      * állapotok és eseméynek egy (véges vagy végtelen) alternáló sorozat
      * a rendszer kezdőállapota:
      * a rendszer állapotmenete minden -re:
      * egy állapot *elérhető*, ha a rendszernek létezik végrehajtási szekvenciája

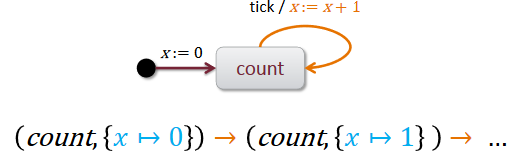
1. **Hierarchia**
   * Definíció
     + ***összetett (Harel) állapotgépek***
       - állapotok közös tulajdonságait és viselkedését általánosítja
       - *statechart*/ Harel-féle összetett állapotgép
       - Mealy-féle egyszerű állapotgép + állapothierarchia, ortogonalitás, változók, pszeudoállapotok
   * Példa
     + pl. előző ábrába minden eseményhez csatolunk egy reset gombot, mely a stopra ugrik vissza
     + szoftverek: Yakindu, UML
   * Definíció
     + ***régió***
       - összetett állapotnál
       - további állapotokat (kezdőállapotot is), tranzíciókat tartalmaz
       - *legfelső szintű régió*: magát az állapotgépet tartalmazza
     + ***állapotkonfiguráció***
       - ha egy *tartalmazott állapot* (egyszerű vagy összetett) aktív, akkor az őt tartalmazó összetett állapot is akítv
       - állapotok egy olyan maximális, tovább nem bővíthető halmaza, melyek egyszerre lehetnek aktívak a rendszerben
2. **Ortogonális dekompozíció**
   * Definíció
     + ***állapotterek direkt szorzata***
       - komponens állapottereken végzett kompozíciós művelet
       - eredménye egy újabb állapottér (*szorzat állapottér)*
       - szorzat állapotterében
         * komponens állapotterek minden állapot-kombinációjának egy-egy összetett állapot (*állapotvektornak*) felel meg
     + ***szorzatautomata***
       - aszinkron szorzás eredménye
       - ennek a vonatkozó régiójában az állapotok száma a két összeszorzott régió (egyszerű) állapotai számának szorzata
     + *állapottér-robbanás*
       - nagy modelleknél alkalmatlan használat → kezeletlenül nagy modelleket kapunk
     + ***ortogonális állapot***
       - összetett állapot
       - több régióval rendelkezik
       - régiói: ortogonális régiók
       - ezek az egyrégiós összetett állapottal megegyező módon akkor aktívak, ha a tartalmazó állapot aktív
       - aszinkron módon működik
         * tranzíciók külön – külön tüzelnek
       - *versenyhelyzet*: ha a régiók azonos eseményeket dolgoznak fel
         * működés összehangolása megosztott változókon keresztül
3. **Aszinkron szorzat**
   * Definíció
     + a (Mealy-) állapotgépek aszinkron szorzata (régióknak is nevezett) komponens állapotgépeken végzett kompozíciós művelet
     + szorzat eredménye egy (Mealy) állapotgép, melynek
       - állapottere a régiók állapottereinek direkt szorzata
       - kezdőállapotban az összes régió kezdőállapotban van
       - átmeneti szabályait az összes olyan lépés alkotja, melyben
         * pontosan egy régió állapotátmenetet végez
         * többi régió állapota nem változik



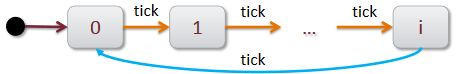
* + *változók*
    - végtelen számláló:



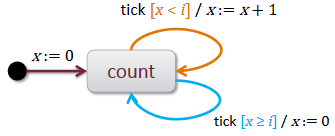
* + - váltózóval „x”:



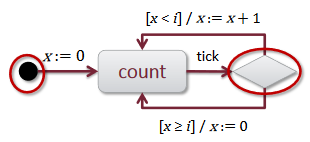
* + *változók + őrfeltételek*
    - ciklus számláló:



* + - őrfeltételekkel:



* + *pszeudoállapot*
    - szemantikailag nem állapot:
      * nincs olyan időpillanat, amikor a rendszert jellemezné
    - szintaktikailag állapot:
      * lehet tranzíció kezdő- vagy célállapot



# **4. előadás: Folyamatmodellezés**



[**BACK**](#_top)

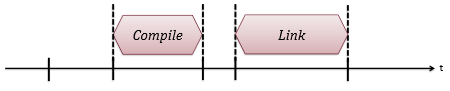
1. **Folyamatok**
   * folyamatmodellezés célja, rendszer folyamatának leírása
   * viselkedési modellek a rendszert többféle módon jellemzik
     + állapot alapú modellek
       - „miként változhat” a rendszer
       - pillanatszerű események
       - rendszereket az állapotukkal jellemezzük
       - milyen állapotokban lelhető fel a rendszer
         * és nevekkel látja el az állapotokat
       - milyen hatásokra mely állapotból mely állapotba lép át
     + folyamatmodellek
       - „mit csinál” a rendszer
       - tevékenységeknek időbeli kiterjedés tulajdonítása
       - megvizsgálva, hogy mely tevékenységek végezhetők el a másik előtt vagy után
       - rendszer állapotainak jellemzése
         * mely időpontban, mely tevékenység aktív, fejeződött már be
   * Definíció
     + ***folyamat***
       - tevékenységek összessége
       - adott rendben történő végrehajtás valamilyen célra vezet
2. **Folyamatmodellek alapjai**
   * Definíció
     + ***elemi tevékenység***
       - időbeli kiterjedéssel rendelkező tevékenység
       - kezdés és befejezésen túl további részleteket nem modellezünk
   * Példa
     + elemi tevékenység
       - C forráskódból futtatható programot szeretnénk
       - forrásállományt le kell fordítanunk „*compile”*
       - fordítóprogramot nem mi készítjük el, csak indítjuk és leállítjuk, nem kell nekünk további részlet
       - időbeli kiterjedéssel bír:
         * háromelemű állapotteret határoz meg

*kezdet* és *vég* időpontban pillanatszerű esemény

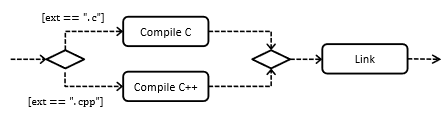
kettő között *folyamatban* van



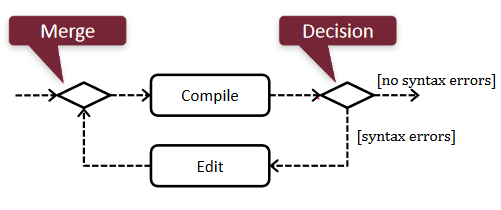
* + - ***szekvencia***
      * tevékenységek szigorú végrehajtási sorrendje
      * folyamatmodellek tevékenységeiből folyamatot építenek fel
        + egyes tevékenységek egymáshoz képest mikor hajthatók végre
        + tevékenységek *vezérlési élek/vezérlésifolyamatélek* kötik össze
  + Példa
    - C program elemi tevékenysége után a forrásfájlt össze kell linkelni más tárgykódokkal → *linkelés*
    - *jelölés szaggatott vonal az ábráknál*



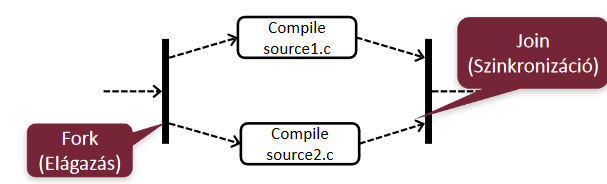
* + Definíció
    - *folyamatpéldány*: folyamatmodellben a folyamat konkrét lefutásai
    - ***vezérlési elem/ vezérlési csomópont***:
      * csomópont a folyamatban
      * a folyamatmodell tevékenységei közül választ ki egyet vagy többet végrehajtásra
  + Példa
    - bizonyos fájlokra C fordítót, másokra C++ kell hívni
    - elágazással lehet ábrázolni, lehet több *ága* is
    - ***nemdeterminizmus***: ez a modell nem adja meg, hogy melyik ágba kerülünk
      * hasznos, ha emberi döntéstől függ a választás
  + Definíció
    - ***őrfeltétel***
      * előzővel ellentétben tehetjük determinisztikussá a választást
      * más modelleknél választási lehetőségek számát csökkentheti
      * folyamatmodellen kívüli tudás alapján kizárjuk a döntés után meglevő ágak némelyikét
      * (*állapotgépeknél is megtalálhatóak*)
  + Példa
    - ugyanebben az elágazásban elhelyezünk egy őrfeltételt
    - eldönti helyettünk, hogyha .c végződik a fájl, akkor C-vel fordítja, ha .cpp, akkor C++-szal
  + Definíció
    - ***döntési csomópont/decision***
      * csomópont a folyamatban
      * belé érkező egyetlen vezérlési él hatására a belőle kiinduló *ágak* (vezérlési élek) körül pontosan egyet választ ki végrehajtásra
      * őrfeltételekkel összhangban van
  + Példa
    - találkozási pont
    - őrfeltétel bármelyik ágra is fut, kivezetésnél ugyanoda érkeznek



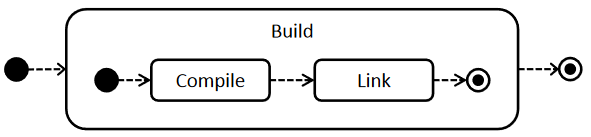
* + - speciális eset, ha csak 1 feladat van megadva, másik úgynevezett *üres vezérlési él*
    - kiszedjük C++ fordítást
    - választás opcionális lesz, hogy egyáltalán akarjuk fordítani C-n vagy nem, például, ha módosult a fáj utolsó megnyitáskor vagy nem
  + Definíció
    - ***ciklus***
      * folyamatmodell (részlet)
      * elágazás valamelyik ágán az elágazást megelőző merge csomópontba jutunk vissza
  + Példa
    - program futásakor találunk fordítási hibát, így a programot ismét le akarjuk fordítani, amíg ki nem küszöböljük
    - ilyenkor ismét visszaugrik a fordítás előtti merge pontba



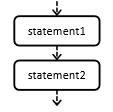
* + Definíció
    - ***konkurens***
      * két bekövetkező tevékenység vagy esemény
      * nincs megkötve milyen sorrendben következnek be
    - ***fork csomópont***
      * csomópont a folyamatban
      * belé érkező egyetlen vezérlési él hatására belőle kiinduló összes vezérlési él (*párhuzamos folyam*) kiválasztja végrehajtásra
    - ***join csomópont (találkozási/szinkronizáló)***
      * csomópont a folyamatban
      * belé érkező összes párhuzamos folyam végrehajtása után kiválasztja a belőle induló egyetlen vezérlési élet
  + Példa
    - nincs meghatározva két forrásfájlt milyen sorrendben fordítunk le
    - többmagos processzornál megpróbálható egyszerre is
    - fork csomópontból, vastag vonallal jelölt, indulnak
    - token bármelyikre mehet
    - joinban találkoznak a fordítás után, két tokent ekkor *összeolvasztjuk*
    - itt bevárják egymást a folyamatok



* + Definíció
    - ***start (flow begin)***
      * csomópont, a folyamat indító eleme
      * pontosan 1 kimenete van
    - ***cél (flow end)***
      * csomópont, a folyamat befejező eleme
      * pontosan 1 bemenete van

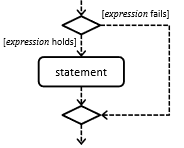


* + - ***folyamatpéldány***
      * folyamatmodellhez tartozó
      * diszkrét eseménysor
      * események alkotják, folyamatmodell által megszabott időrendben:
        + folyamat kezdete
        + folyamatot alkotó tevékenység kezdete
        + folyamatot alkotó tevékenység vége
        + folyamat vége
      * folyamatnak több folyamatpéldánya is lehet

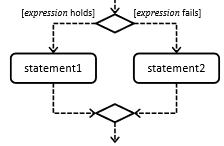
1. **Folyamatmodellek felhasználása**
   * Definíció
     + ***vezérlési folyamat (control flow)***
       - program által meghatározott folyamatmodell
       - elvégzendő lépéseket, végrehajtásukra előírt sorrendet írja le
     + ***imperatív program***
       - programok, melyeket vezérlési folyamot határoznak meg
   * C-szerű vezérlési struktúrák vezérlési folyamatai

<statement1>

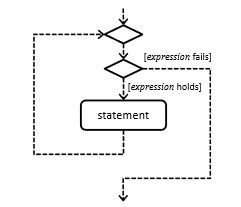
<statement2>

if (<expression>)

<statement>

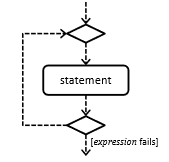
if (<expression>)

<statement>

else

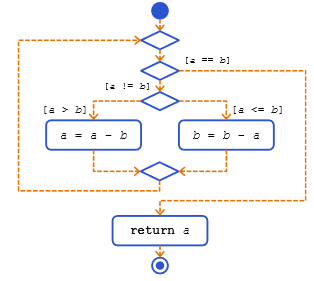
<statement>

while (<expression>)

<statement>

do

<statement>

while (<expression>)

* + Példa:

while (a!=b){

if (a > b)}

a = a – b;

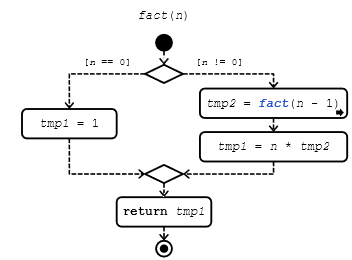
} else {

b = b – a;

}

}

return a;

* + Definíció
    - ***ciklomatikus komplexitás***
      * vezérlési folyamhoz tartozó:
        + vezérlési élek száma
        + vezérlési csomópontok száma
  + Példa:
    - n! meghatározása

int fact (int n){

int tmp1, tmp2;

if (n == 0)}

tmp1 = 1;

} else {

tmp2 = fact(n – 1);

tmp1 = n \* tmp2;

}

}

return tmp1;

* + Definíció
    - ***jólstrukturált blokk/részfolyamatok***
      * egyetlen elemi tevékenység önmagában
      * egyetlen folyamathivatkozás/hívás
      * üres vezérlési élszakasz
      * „*soros kapcsolás*”: jólstrukturált blokkok szekvenciája
      * „*fork-join kapcsolás*”: jólstrukturált blokkok egy ágú fork és ágú join közé zárva szekvenciája
      * „*decision-merge kapcsolás*”: jólstrukturált blokkok egy ágú decision és ágú merge közé zárva szekvenciája
      * „*ciklus*”:
        + kétágú merge csomóponttal kezdődik
        + majd jólstrukturált blokk jön
        + majd kétágú decision

ennek egyik ága a részfolyamat vége

másik jólstrukturált blokktól merge-be tér vissza

* + - * egyetlen start és egyetlen cél csomópontja van

# **5. előadás: Modellek ellenőrzése**



[**BACK**](#_top)

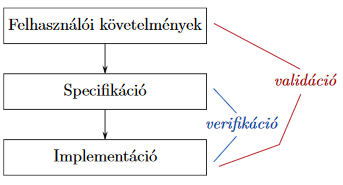
1. **Követelmények a modellekkel szemben**
   * Definíció
     + ***helyesség***
       - modell vagy kód megfelelése a követelményeknek
         * megfelel a funkcionálisnak
         * nemfunkcionálisak ellenőrzése
       - szempontok
         * mindig képes teljesíteni a feladatot
         * hibamentes
         * nincs tiltott viselkedés
     + ***funkcionális követelmény***
       - követelmény, mely egy rendszerösszetevő által ellátandó funkciót definiálnak
     + **csoportosítás**
       - *megengedett viselkedés*
         * milyen állapotban lehet/nem lehet a rendszer
         * milyen viselkedés tilos
         * univerzális követelmények

mindig igaznak kell lenniük

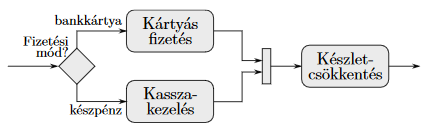
* + - * *elvárt viselkedés*
        + milyen állapotokba kell eljutniuk
        + milyen funkciókat kell tudnia
        + egzisztenciális követelmények

lehessen lehetőség a teljesítésülésükre

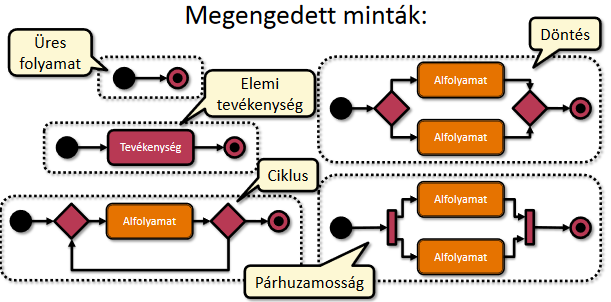
* + - ***nemfunkcionális követelmény/extrafunkcionális***
      * ezeken kívül eső követelmények
      * ezek a rendszer minőségére vonatkoznak, pl. megbízhatóság, teljesítmény kritériumai
    - ***biztonsági követelmény***
      * definiálják mely viselkedés engedett, mely tiltott
      * univerzális követelmények, minden időpillanatban teljesülniük kell
    - ***élőségi követelmény***
      * definiálják az elvárt viselkedést
      * egzisztenciális követelmények, megfelelő körülmények közt előbb-utóbb teljesíteni képes bizonyos elvárásokat
    - ***holtpont (deadlock)***
      * állapot a rendszerben, amelyben a végrehajtás megáll
      * rendszer többé nem képes állapotot váltani
      * nem mutat semmilyen viselkedést
      * csak külső segítséggel képes kilépni
    - ***livelock***
      * végrehajtás megáll
      * résztvevő komponensek egy végtelen ciklusban ragadnak → nem végez hasznos tevékenységet
  + Példa
    - livelock
      * két ember szembe találkozik egymással, de udvariasan kitérnek egymás elől, de mindig azonos irányba
      * nem tudnak mozogni, képesek állapotot váltani, de nem haladnak előre
      * „végtelen ciklus”
    - deadlock
      * kölcsönösen nem tudnak megmozdulni a másiktól
      * „örök várakozás”
  + **helyességvizsgálatok**
  + Definíció
    - ***verifikáció***
      * azt vizsgáljuk, hogy az implementáció (az elkészített modell vagy rendszer) megfelel-e a specifikációnak
      * kérdés:
        + helyesen fejlesztjük-e a rendszert
        + megfelel-e az előírtaknak
    - ***validáció***
      * a rendszert a felhasználói elvárásokhoz hasonlítjuk
      * kérdés:
        + megfelelő rendszert fejlesztjük-e
  + Példa
    - kereszteződést felszerelünk jelzőlámpákkal
    - minden tökéletesen működik, verifikáció helyes
    - megrendelő megkérdezi: „És hol lehet átkapcsolni villogó sárgára?”
    - ez nem volt a specifikáció része → nem tudjuk teljesíteni, de tudjuk, hogy amúgy a rendszer jól működik, csak a rendelői elvárásoknak nem felel meg



1. **Statikus ellenőrzés**
   * Definíció
     + ***statikus ellenőrzés***
       - a vizsgált rendszert vagy modellt annak végrehajtása, szimulálása nélkül elemezzük
   * Példa
     + „ránézésre látszik” hibákat érdemes ezzel javítani
     + szintaktikai hibák keresésénél is alkalmazzuk, amikor a rendszer tipikusan nem is futtatható
   * ***szintaktikai hiba***
     + azok a hibák, amikor a modell nem felel meg a metamodelljének vagy a program nem felel meg a programozási nyelv formai megkötéseinek
     + szöveges leírások esetén könnyebben előforduló hiba
     + nagy biztonsággal kimutathatók (legkésőbb futtatás vagy végrehajtás során)
     + ritka, hogy egy statikus ellenőrző helyes kódot vagy modellt szintaktikailag hibásnak értékel
   * Példa
     + állapotokhoz nem kötött állapotátmenet egy állapotgépben
     + hiányzó zárójel vagy elgépelés egy programkódban
   * ***szemantikai hiba***
     + ha a rendszer szintaktikailag helyes, de logikailag nem
     + nem olyan egyértelmű hibákat „code smellnek” nevezünk, ezeket a statikus ellenőrző ki szokta szúrni
   * Példa
     + szintaktikailag helyes, szemantikailag nem
     + join mindkét bemeneten tokent vár, de nem léphet tovább (holtpont)

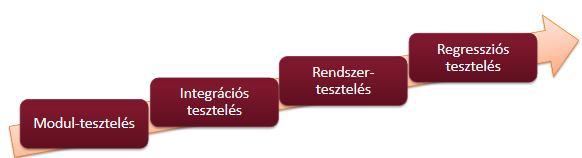


* + **védekezés szemantikai hiba ellen**
    - hibaminták megfogalmazása
      * statikus ellenőrző ezeket a hibamintákat keresi majd
    - megkötések alkalmazása
      * programozásnál kódolási szabályok
        + ne használjunk pointereket C-ben
      * jólstrukturált folyamatmodell kizárólag adott mintákból:

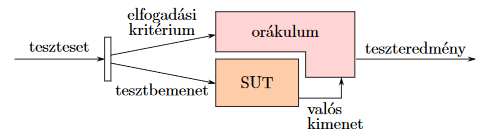


* + ***szimbolikus végrehajtás***
    - bonyolultabb szemantikai hibák kiszűrésére
    - konkrét értékek helyesett szimbolikus értékekkel való „imitálás”
    - belső elágazások által támasztott feltételeket összegyűjtük → következtetünk az egyes változók értékeire

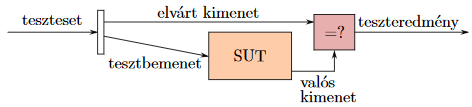
1. **Tesztelés**
   * Definíció
     + ***tesztelés***
       - olyan tevékenység, amely a rendszert vagy egy futtatható modelljét bizonyos meghatározott körülmények közt futtatjuk/szimuláljuk
       - célja: vizsgált rendszer minőségének felmérése, javítása, hibák azonosítása
         * hibajelenségek meglétét vagy hiányát vizsgáljuk
     + **fajtái, szakaszai**
       - *modultesztelés*
         * egy komponens leválasztása és tesztelése
       - *integrációs tesztelés*
         * több komponens együttes tesztelése
       - *rendszertesztelés*
         * a teljes rendszer együttes tesztelése
       - *regressziós tesztelés*
         * változtatások utáni (szelektív) újratesztelés



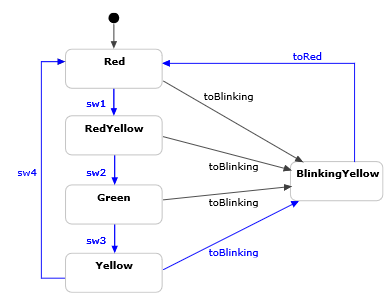
* + - ***debuggolás***
      * egy bizonyos hibajelenség lokalizálása, kiváltó okának megkeresése
  + **tesztelés komponensei**
  + Definíció
    - ***tesztelendő rendszer (SUT, system under test)***
      * az a rendszer, amelyet a teszt során futtatni fogunk vizsgálat céljából
    - ***tesztbemenetek***
      * a tesztelendő rendszer számára biztosítandó bemeneti adatok
    - ***tesztorákulum***
      * olyan algoritmus és/vagy adat, amely alapján a végrehajtott tesztről eldönthető annak eredménye
    - ***teszteset***
      * adatok összessége, amelyek egy adott teszt futtatásához és annak értékeléséhez szükségesek
    - ***tesztkészlet***
      * tesztesetek adott halmaza
    - ***tesztfuttatás***
      * egy vagy több teszteset végrehajtása
  + tesztfuttatás után orákulum segítségével megtudjuk a teszt eredményét (pass, fail, error)
  + error: nem lehet eldönteni, hogy a teszt sikeres vagy nem
  + teszteredmény megkapható
    - kapott, tesztesetben megfogalmazott elvárt kimenetek összehasonlításával
    - referenciaimplementációval összehasonlításból
    - ellenőrizhetjük implicit elvárásokat
      * pl. a kód nem dob-e kivételt



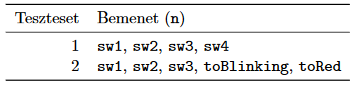
* + - legegyszerűbb eset: teszteset közvetlenül tartalmazza az adott tesztbemenetekre elvárt kimeneteket



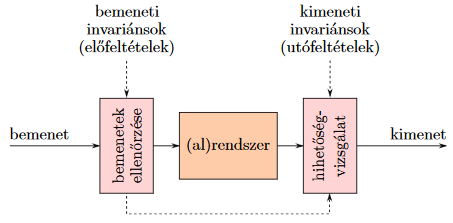
* + **tesztelés metrikái**
    - tesztelés célja a minőség javítása, hibák megtalálása, javítása
    - egy idő után elfogynak a hibák, a rendszer „elég jónak” mondható → utána nem gazdaságos folytatni a tesztelést
    - honnan tudjuk mikor jutottunk el ide?
      * tesztkészlet fedésének mérése
      * tesztkészlet meglátogat minden állapotot, meghív minden metódust akkor sem vizsgáltunk meg mindent
      * ha mindent meghív, nem biztos, hogy minden utasítást érint (pl. elágazások)
    - ***állapotfedettség***
    - ***átmenetfedettség***
    - ***utasításfedettség***
  + Példa
    - közlekedési lámpa állapotgépe



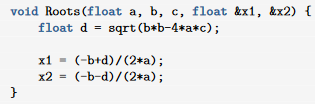
* + - tesztkészlet



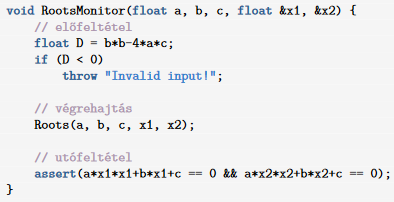
* + - ez a két teszt az összes állapotot be fogja járni, az állapotlefedettség 100%
    - átmenetfedettség nem teljes, 9-ből 6 tranzíciót fed le → átmenetefedettség 66,6%
* **Tesztelés futásidőben**
  + **„öntesztelés” vagy monitorozás**
    - magas minőségi elvárásoknál
      * pl. biztonságkritikus alkalmazási területek
    - külső komponensek használata, melyeknek minőségéről nem tudunk
    - ***invariáns***
      * követelmények, amelyek teljesülését folyamatosan, minden állapotban elvárjuk



* + **monitorozás 2 fő lépése**
    - *bemenetek ellenőrzése*
      * bemeneti adatok megfelelőssége
      * definiált bemenetei invariánsok alapján
    - *hihetőségvizsgálat*
      * kimeneti adatok megfelelőssége
      * definiált kimenetei invariánsok alapján
  + Példa
    - másodfokú egyenlet gyökeit kiszámoló

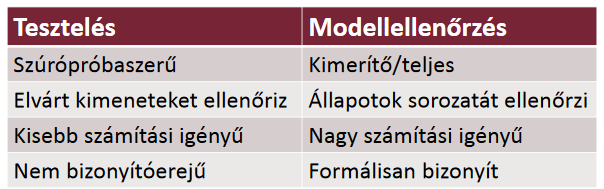


* + - nem helyes, hiányoznak elő – és utófeltételek (diszkrimináns nem lehet negatív stb.)



* + - ***logelemzés***
      * monitor futtatása naplózott bemeneten/kimeneten

1. **Formális verifikáció**
   * Definíció
     + olyan módszerek, amelyek segítségével adott modellek vagy programok helyességét matematikailag precíz eszközökkel vizsgálhatjuk
     + fontosabb módszerei
       - **modellellenőrzés**
         * lehetséges viselkedések kimerítő vizsgálata



* + - * **automatikus helyességbiztosítás**
        + axiómarendszerek alapján tételbizonyítás
      * **konformanciavizsgálat**
        + adott modellek közt bizonyos konformanciarelációk teljesülését vizsgáljuk
        + beláthatjuk, hogy különböző modellek viselkedése azonos vagy nem
        + megfelelősség ellenőrzése

# **6. előadás: Vizuális adatelemzés**



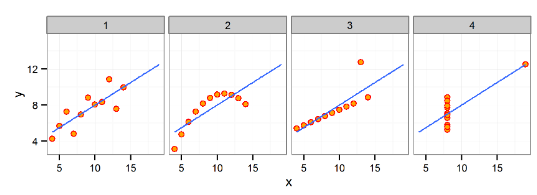
[**BACK**](#_top)

1. **Felderítés, megerősítés, szemléltetés**
   * **vizualizáció eszköze**

felderítő adatelemzés

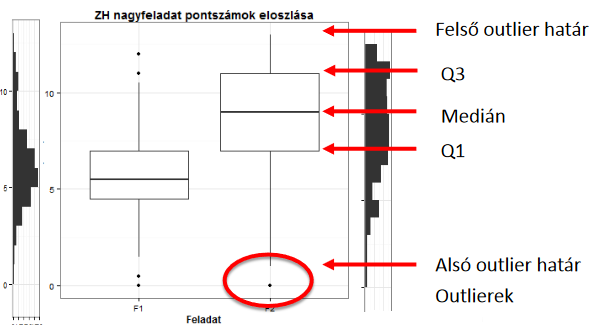
(*EDA*)

* + - adatok megértése, hipotézisek megsejtése
    - hipotézisek és modellek megerősítése
    - eredmények szemléletesebb prezentálása → megerősítő adatelemzés (*CDA*)
  + ***EDA***
    - az adatelemzési folyamat nagy hangsúlyt fektet az adatok „megértésére”
      * az adatok által leírt rendszer
      * adatokon belüli és az adatok és általuk leírt rendszer közötti összefüggések
    - adatok grafikus reprezentációja, mint ennek eszköze, kiemelt szerep
    - modelljelölt-építés, hipotézisek felállítása iteratív módon
      * modellspecifikáció – analízis – modell-újraspecifikáció lépéssorozat ismétlésével
    - CDA-val összevetve
      * CDA: pontos statisztikai mértékeket számol
      * EDA: sejtéseket fogalmaz meg, amelyeket a statisztikai/elemzés matematikai eszközeivel igazolhatunk (vagy cáfolhatunk, módosíthatunk)
  + Példa
    - Dr. John Snow 1854-ben londoni kolerajárvány alatt bepontozta a térképen a halálesetek számát
    - kirajzolódott, hogy a Broad Streeten levő kút lehet az oka
  + **Anscombe négyese (*Anscombe’s quartet)***
    - *példa arra, hogy a vizuális elemzés könnyen ad olyan plusz információt, melyet az alap statisztikai módszerek nem feltétlenül vesznek észre*
    - hibás feltételezések elkerülése és intuíció



* + **folyamat alapja interakció**
    - adatvizualizáció
      * több ábra együttes vizsgálata
    - vizuális kiértékelés
      * emberi kognitív képességek használata
    - vizuális kiválasztás, manipuláció
    - interpretáció, korreláció más modellekkel, kiértékelés

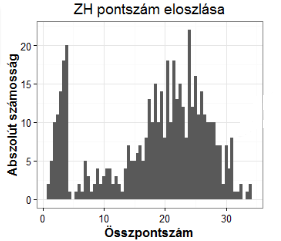
1. **Numerikus és kategorikus változók**
   * ***numerikus (numerical)***
     + az alapvető aritmetikai műveletek értelmesek
     + **folytonos**
       - mért, tetszőleges értéket felvehet
         * adott tartományon belül
         * adott pontosság mellett
       - például: teremben ülők ZH pontszámának átlaga
     + **diszkrét**
       - számolt, véges sok értéket vehet fel adott tartományban
       - például: előadáson ülők száma
   * ***kategorikus (categorical)***
     + matematikai műveletek nem értelmezhetőek rajtuk, max sorbarendezés
       - **ordinális**
         * teljes rendezés az értékeken
         * például: szállodai csillagok
       - **nominális**
         * például: férfi, nő
   * ***terjedelem jellemzése: percentilis***
     + -edik percentilisnél az értékek -a kisebb
   * Példa:
     + - percentilis:
       - percentilis:
       - percentilis:
2. **Egydimenziós diagramok**
   * reprezentálása pontdiagramon (*dot plot*)
   * ***doboz-ábra (box plot)***
     + 5 jellemzőt szemléltet
       - minimum
       - maximum
       - medián
       - első kvartilis
       - harmadik kvartilis
     + kvartilisek távolságot egy doboz mutatja
     + **„bajusz**” **(*whisker*):** 
       - első kvartilis alatti és harmadik feletti vonal
       - legfeljebb a kvartilisek közötti távolság 1,5x nyúlik
     + ezen kívül eső megfigyelések pontként ábrázolva
     + egy kategorikus változó mentén részhalmazokra bontva használjuk



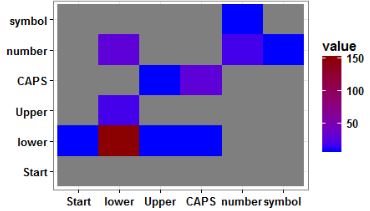
* + - több eloszlás esetén jellemzők gyors, vizuális összehasonlítására alkalmas
    - **hátránya**
      * eloszlást erősen absztrahálja
      * → multimodalitás nem olvasható le róla
  + *változó eloszlás becslése*
    - , ahol intervallumon megfigyeléssel rendelkezünk
    - képezzünk egy nemátfedő intervallumokból (-ekből, cellákból) álló partícionálását:

, ahol

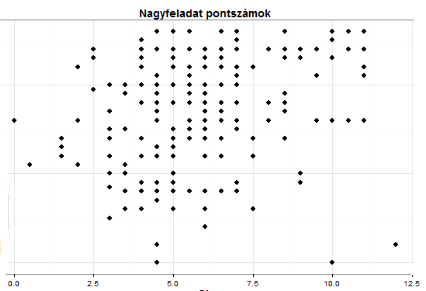
* + - , ahol -ik cella indikátor-függvénye és
    - -be eső minták száma )
    - hisztogram, mint változó eloszlás becslője:
    - EDA szempontjából hátrány, hogy alakja érzékeny a(z)
      * első cella kezdőpontja,
      * cellaszélesség megváltasztására



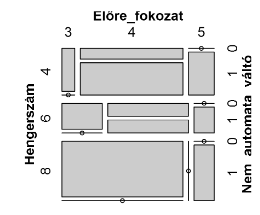
1. **Többdimenziós diagramok**
   * ***kétdimenziós diagramok***
     + **folytonos-folytonos eset**
       - pl. szórásdiagramok, hőtérképek



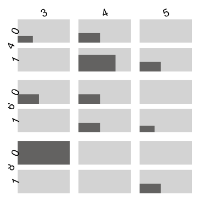
* + - **kategorikus-folytonos eset**
      * kategória-kombinációk relatív számosságának felismerése → mozaik/fluktuációs diagram, de ezek már n-diagramtípusok, 2 változós eset speciális
      * pl. kategóriánkként alkalmasan színezett hisztogram, doboz-diagram
  + ***n-dimenziós diagramok***
    - **tisztán kategorikus eset**
      * pl. mozaik, fluktuációs diagram
    - **tisztán folytonos eset**
      * pl. párhuzamos koordináta diagram
        + megjeleníthető kategorikus változó rajta
    - **SPLOM – *scatterplot matrix* – szórásdiagram-mátrix**
      * sokváltozós adatkészlet változó-vektorát megfeleltetjük egy mátrix sorainak, oszlopaink
      * cellákban változópárok szórásdiagramja



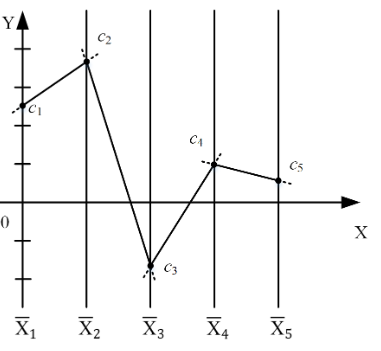
* + - * overplotting kezelése (*túl sok pont egy helyen*)
        + jitter
        + átlátszóság
        + méret növelése
    - **mozaik és fluktuációs diagram**
      * *mozaik*:
        + kategorikus változók érték-kombinációi előfordulási gyakoriságainak területarányos vizualizációja
        + ábrát alkotó csempék/lapok (*tiles*) négyzet rekurzív vízszintes és függőleges darabolásával kapjuk
        + olvashatóság miatt 8 változónál többet ne használjunk (már 4-5-nél is gond)



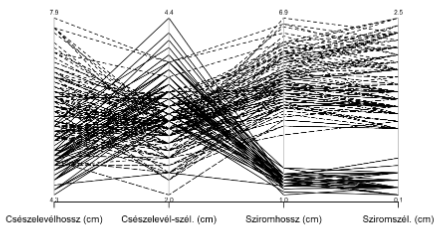
* + - * *fluktuációs*:
        + mozaik nagyobb dimenziószám esetén → fluktuációs diagram szükséges (olvashatóbb)
        + érték-kombinációkhoz azonos méretű lapokat rendelünk → kombinációk könnyen beazonosíthatók
        + legnagyobb elemszámú lapot teljesen kitöltjük, többit az előfordulások relatív gyakorisága alapján
        + hátrány: kitöltött idomból nem tudjuk leolvasni a változókban értelmezett gyakoriságot



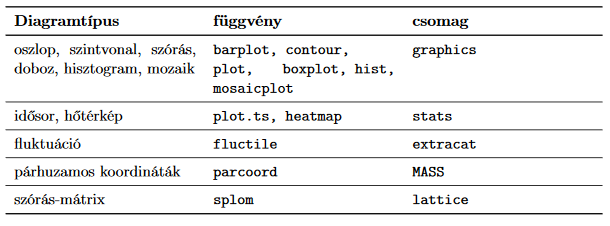
* + ***párhuzamos koordináta diagram***
    - Descartes-féle koordinátákkal ellátott euklideszi síkban egymástól egységnyi távolságra elhelyezi az valós egyenes másolatát
    - ezeket tengelyként használva ábrázolja -dimenziós euklideszi tér pontjait
    - koordinátákkal rendelkező pont képe teljes poligon-vonal
    - , szegmenseket meghatározó pontja rendre párhuzamos tengelyekre eső pontok



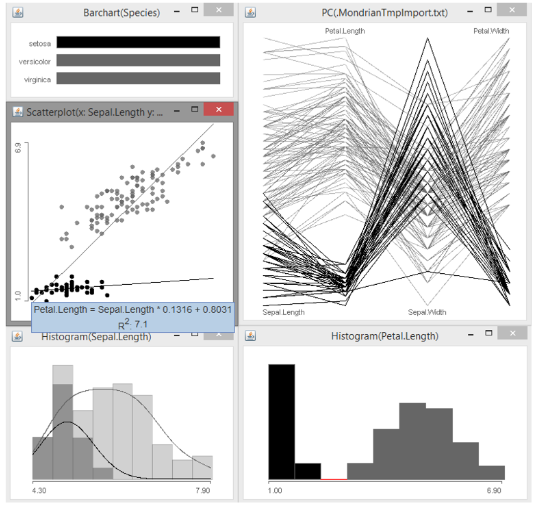
* + - magas dimenziószám esetén áttekintésre alkalmas
    - hátrány: növekvő pontszám esetén egyre kevésbé átlátható
      * pl. Fisher-féle írisz-adatkészlet párhuzamos koordinátákra



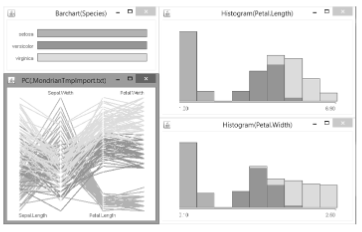
* + - * DE, ha teszünk bele:
        + faktorváltozók szín szerinti megkülönböztetés
        + részhalmaz kiválasztás
        + tengelyen intervallum kiválasztás
      * → nagyon hatékony EDA eszközzé válik
  + **eszköztámogatás**
    - táblázat megadja a diagramtípusokat megvalósító függvényeket
    - grafika területén különösen igaz, hogy ugyanannak a funkciónak több megvalósítása elérhető



1. **Interaktív statisztikai grafika**
   * **eszköz által támogatott interakciók:**
     + lekérdezések (*queries*) és helyi interakciók
     + kiválasztás és csatolt kijelölés (*selection and linked highlighting*)
     + csatolt elemzések/analízis (*linked analyses*)
   * ***lekérdezések***
     + Definíció
       - interaktív diagramon megjelenített elemekkel kapcsolatos statisztikai információk megjelenítése (egérrel)
     + **megjeleníthető adatok függenek:**
       - ábra által alkalmazott statisztikai transzformáció
       - aktív kijelöléstől
     + **szórásdiagramon**
       - pont koordinátáit
       - egy kijelölés koordináta-intervallumait kérdezhetjük le
     + **oszlopdiagramon**
       - kategória elemszámát
       - aktív kijelölésbe tartozó elemek számát
   * ***helyi interakciók***
     + diagramokkal mellékhatásmentesen interakcióba léphetünk
     + egyes operációk általánosan megjelennek, többi ábra-specifikus
   * ***kiválasztás és csatolt kijelölés***
     + ez adja a minőségi különbséget a statikus és az interaktív statisztikai vizualizáció között
     + adatkészleten több diagram olyan, mint egy reláció különböző projekciói (diagram változóira), ezek statisztikai transzformációi
     + diagramon elemeket egérrel kiválasztva az inverz transzformáció egy „sorhalmazt” határoz meg az eredeti relációban, melynek képe többi diagramon „kiemelve” vizualizálható



* + - *kategória-vezérelt színezés (colour brush)*
      * kategóriákat megjelenítő diagramoknál
        + pl. oszlopdiagram, mozaik-diagram, hisztogram
      * színezés a kezdeti diagram minden eleméhez egy színt rendel → többit ezzel színezi



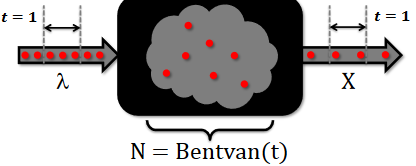
* + ***csatolt analízis***
    - a kiválasztások változásával reaktívan analízisek, illetve statisztikai modellek felállításai újrafuthatnak
    - megvalósítás: Mondrian szórásdiagramra illeszthető regressziós egyenes
      * teljes adatkészlet azonosítása, feltüntetése
      * paraméterek lekérdezéssel megtekinthetőek

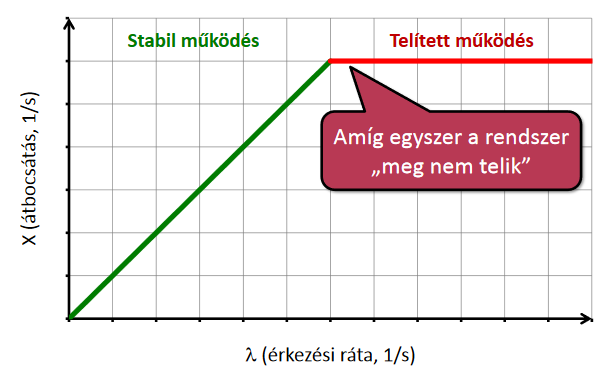
# **7-8. előadás: Teljesítménymodellezés**



[**BACK**](#_top)

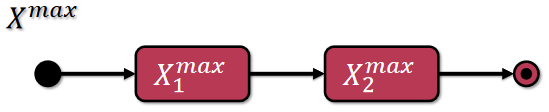
1. **Teljesítménymodellezés**
   * rendszer vizsgálata, mely felhasználókat kiszolgál véges erőforrásokat használva
     + vizsgálat: feldolgozási idő (*válaszidő*) fókuszál
   * *átbocsátás*: egységnyi idő alatt feldolgozott tranzakciók száma
     + erőforrások kihasználtsága a *rendszer egyensúlyi állapotá*ban
   * *tranzakció*
     + teljes rendszer felé érkező felhasználói kérések
     + jelölés:
   * *kérések*
     + alrendszernek továbbított feladatrészek
     + jelölés:
   * *átlapolódás* van, ha
     + van várakozási sor a rendszerben
     + van több feldolgozóegység
     + ha nincs átlapolódás → minden pillanatban legfeljebb egy tranzakció van a rendszerben
2. **Rendszerszintű tulajdonságok**
   * Definíció
     + ***érkezési ráta***
       - egységnyi idő alatt érkező kérések
       - jelölés:
       - mértékegysége:
     + ***átbocsátás***
       - egységnyi idő alatt feldolgozott kérések
       - jelölés: , mint „throughput”
       - mértékegysége:



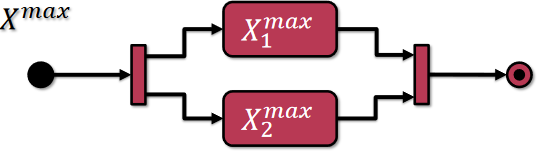


* + - ***válaszidő***
      * felhasználói kérések által a rendszer határain belül töltött átlagos idő
      * jelölés: , mint „round-trip time”
      * mértékegysége:
    - ***rendszerben lévő kérések átlagos száma***
      * átlapolódás mértéke
      * jelölés:
      * mértékegysége:
    - ***rendszerben egyensúlyi állapot***
      * ha
      * vagyis: egységnyi idő alatt ugyanannyi új felhasználói kérés érkezik a rendszerben, mint amennyit ezalatt feldolgozott
      * átlagos értékek **CSAK EKKOR** használhatóak

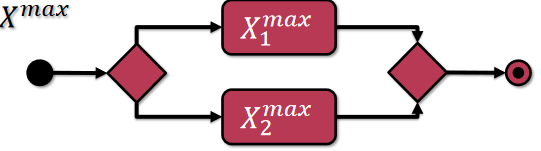
1. **Folyamatmodellek elemzése**
   * Definíció
     + ***szekvenciális komponálás***



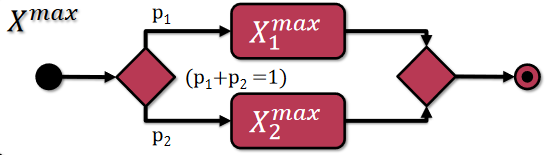
* + - * + egyik tevékenység hiába gyors, másik feltorlódhat
        + pl. Okmányiroda
      * *szűk keresztmetszet*
        + minimumhelyet adó tevékenység
    - ***párhuzamos komponálás***



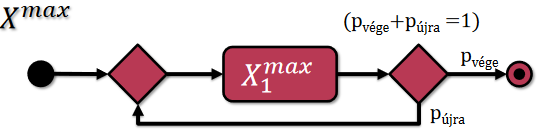
* + - * + egyik tevékenység hiába gyors, be kell várniuk egymást
        + pl. ZH javítás
      * szűk keresztmetszet
    - ***komponálás szabad választással***



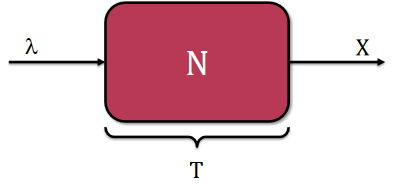
* + - * + tokenek mindkét irányba mehetnek
        + ha egyik telítésben van, másik fogadhat tokent
        + pl. pénztárak
    - ***komponálás kötött arányú választással***



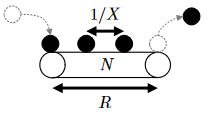
* + - * + tokenek valószínűséggel választják az tevékenységet
        + teljes folyamatból , jut az tevékenységre
        + pl. felhasználó viselkedése egy weblapon vásárlásnál
      * szűk keresztmetszet
    - ***komponálás ciklussal***



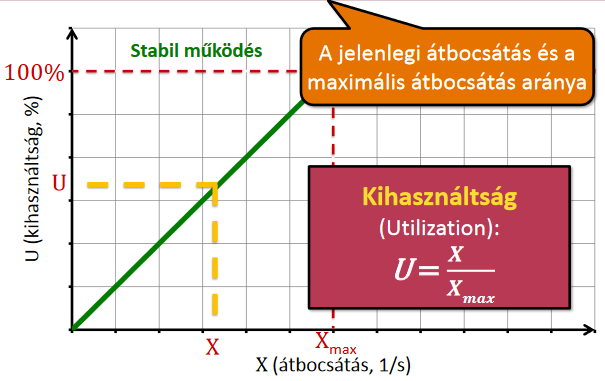
* + - * pl. felhasználó a rendszerben, eséllyel kilép, új kérés
    - ***vizitációs szám***
      * folyamat a végrehajtása során átlagosan hányszor fut le az adott tevékenység/alfolyamat
        + választás esetén maga a döntési valószínűség
        + ciklus esetén a várható iterációk száma
      * *átbocsátóképesség a vizitációs szám ismeretében*
      * *végrehajtási idő*
    - ***Little-törvény***
      * érkezési ráta
      * átbocsátás
      * rendszerben töltött idő
      * rendszerben lévő tokenek száma
      * tehát: rendszerben tartózkodó kérések átlagos száma egyenlő az átbocsátás és az átlagos rendszerben töltött idő szorzatával



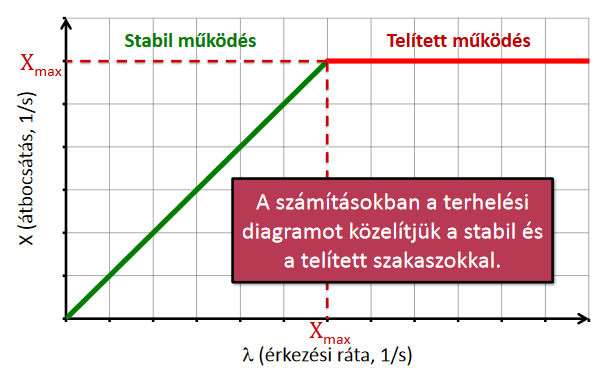
* + - * például: futószalag
        + ideig tart végighaladni rajta
        + időnként ráteszünk egy új elemet
        + az első elem levételének pillanatában



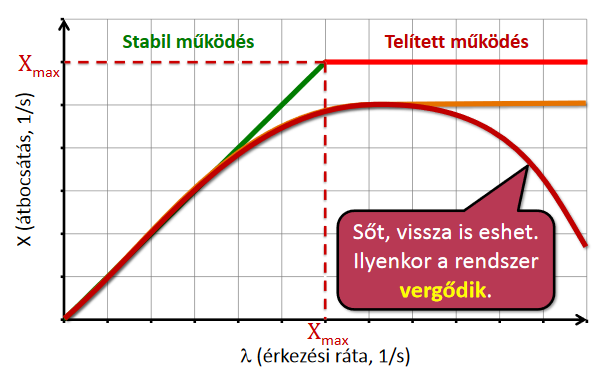
* + Definíció
    - ***látogatások átlagos száma***
      * megadja, hogy egy tranzakció átlagosan hány kérést generál az alrendszer (erőforrás) felé
      * alrendszerek, erőforrások teljesítményjellemzőit ezzel tudjuk átváltani rendszer jellemzőire és vissza
      * mértékegysége: (kérés/tranzakció)
      * kiszámítás:
    - ***szolgáltatásigény***
      * egy tranzakció átlagosan mennyi ideig használja az adott alrendszert/erőforrást
      * rendszerszintű
      * jelölés: , mint „service-demand”
      * mértékegysége:
    - ***erőforrásigény***
      * egy kérés átlagosan mennyi ideig használja az adott alrendszert/erőforrást
      * alrendszerszintű
      * jelölés: , mint „resource demand”
      * mértékegysége:
    - ***kihasználtság***
      * megmutatja, hogy a globális teljesítménykorlátoktól milyen távol működik a rendszer
      * jelölés: , mint „utilization”



* + - ***közelítő vs. valós terhelési függvény***

* + - ***vergődés***
      * rendszer átbocsátása visszaesik a telített működés során

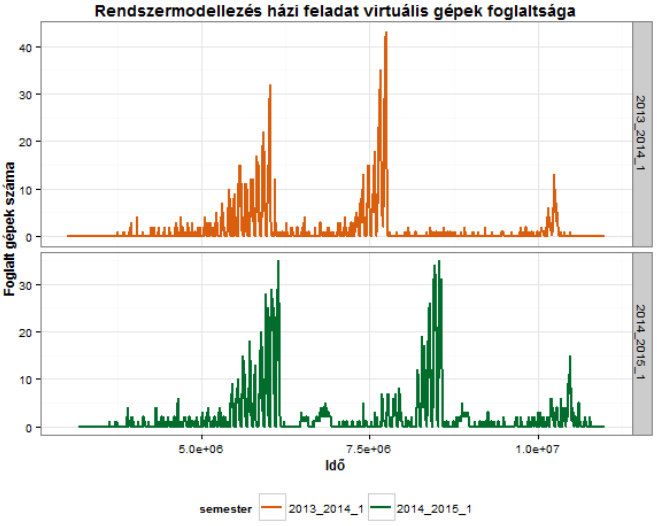


* + - ***kizárólagos erőforrás törvény***
      * ha kérések közül egyszerre legfeljebb egy futhat
        + pl. egyetlen szerver futtatja őket, azonos változót írnak
        + többi kérés sorban áll
        + átlagos végrehajtási idő mellett:
    - ***kizárólagos erőforrás kihasználtsága***
      * átbocsátás és az átbocsátóképesség aránya
    - ***Forced Flow törvény*** 
      * erőforrás, mint alrendszer átbocsátás kiszámítása a teljes rendszer átbocsátásából
      * ebből a **Little-törvény**:
    - ***kihasználtság törvény***

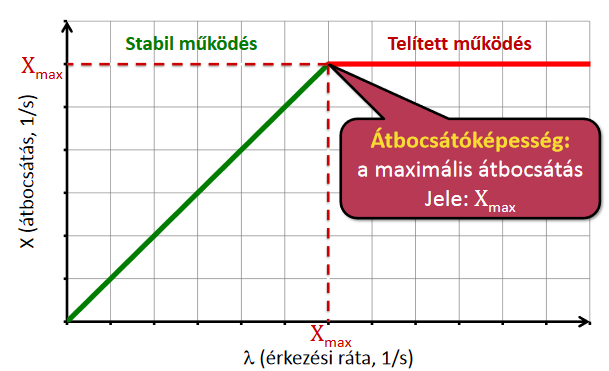
erőforrás legfeljebb ennyi kérést képes kiszolgálni

* + - * maximum darab kérés végrehajtása alatt
      * ha , akkor értéke közvetlenül megadja a kihasználtság értékét
  + Példa
    - * erőforrás: disc
      * 1 kérés 0,0225 mp-ig tart
      * mekkora a kihasználtság?
      * sorban állás is van a disc előtt
      * disc:
      * kérések átlagos száma a rendszerben:
      * átlagos rendszerben tartózkodási idő
      * átlagos sorban állási idő
      * átlagos sorban állási idő?
      * kérések átlagos száma a sorban?
  + Definíció
    - ***Zipf törvénye***
      * az szó előfordulási gyakorisága
      * a korpuszra jellemző érték
      * egyszerűsítve
        + (*frequency*) gyakoriság
        + (*popularity*) a szöveg „rangja” (csökkenő sorrendben)
    - ***Zipf törvénye pl. Web dokumentumokra***
      * hivatkozások (elérések)
      * rang ( leggyakoribb)
      * pozitív konstans
  + Példa
    - * weboldalak aloldalainak látogatottsága
      * slágerlisták
      * városok populációja rangsoruk szerint
    - ***terhelés változása***
      * átlagos értékekkel számoltunk
      * a rendszer viselkedését a terhelés (*intenzitás*) függvényében néztük
      * DE: valójában nem (feltétlenül) előre kiszámíthatóan változik a terhelés
      * valójában:
        + a rendszer viselkedése *időben* változik
        + ennek műszaki hatásai vannak

váltás feladatok közt, erőforrásfoglalás stb. (pl. OS)



* + - ***átbocsátóképesség***
      * erőforráskészlet **felső határ**t szab az elvégezhető munka mennyiségének, az átbocsátásnak
      * esetén



* + - ***szűk keresztmetszet***
      * egyes erőforrásokat a teljes rendszer különböző mértékben használja
      * **egyikük** korlátozza ténylegesen a rendszer tényleges átbocsátóképességét
      * átbocsátóképességgel felírva a Little-törvényt, megkapjuk , átlapolódás maximális mértékét a rendszerben amikor :
    - ***szolgáltatásigény törvénye***
      * adott erőforrásra vonatkozó szolgáltatásigény meghatározása
      * egyetlen erőforráspéldány esetén

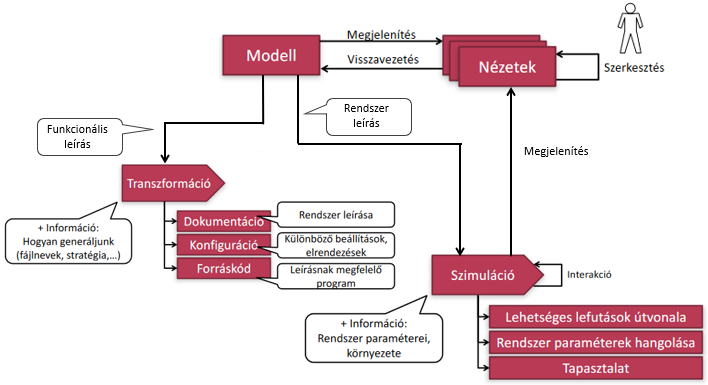
1. **Átlagos mértékek számítása mérési eredményekből**
   * Definíció
     + ***mérési idő***
       - jelölés: , mint „time”
       - mértékegysége:
     + ***tranzakciók száma***
       - mérési idő alatt elvégzett tranzakciók száma
       - jelölés: , mint „count”
       - mértékegysége: vagy alrendszereknél
     + ***tranzakciók száma***
       - egyes erőforrások foglalási ideje a mért időtartamon belül
       - jelölés: , mint „busy time”
       - mértékegysége:
   * egy egypéldányos, átlapolódásmentes erőforrás átlagos kihasználtsága a mérés ideje alatt
   * mérésidő alatt átlagos átbocsátás
   * tranzakciók átlagos szolgáltatásigénye

# **9. előadás: Modellező eszközök, kódgenerálás**

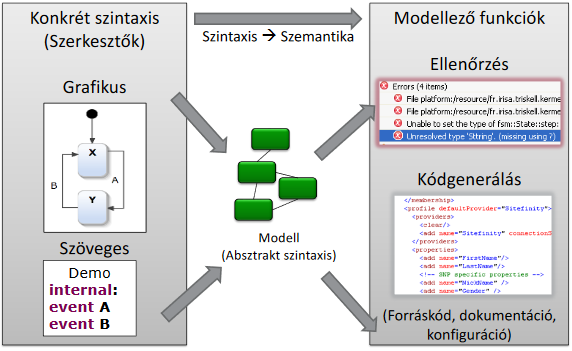


[**BACK**](#_top)

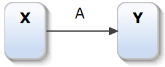
1. **Modellező eszközök funkciói**
   * szöveges
   * grafikus
   * egyéb szerkesztőfelület



1. **Modellező eszközök**
   * Yakindu modellezési funkciói



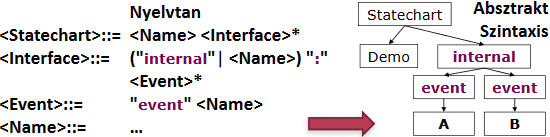
* + Definíció
    - ***absztrakt szintaxis***
      * szerkesztés alatt álló rendszermodell strukturális modellje
      * modellező program kezeli
  + Példa
    - Yakindu modell



* + - absztrakt szintaxis



* + - ***konkrét szintaxis***
      * cél: konkrét megjelenítés
      * *szöveges szintaxis* (programozási nyelv)
        + feladata: szöveg ↔ modell, nyelvtani szabályok alapján



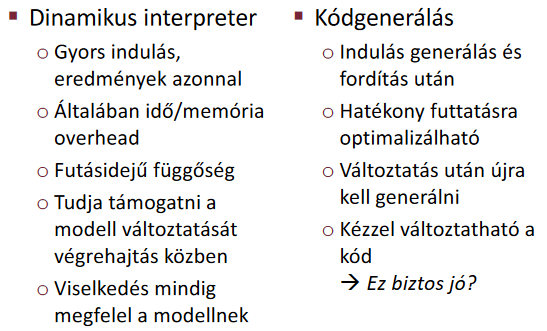
* + - * *grafikus szintaxis* (diagram)
        + feladata: diagram ↔ modell, nézeti modell szabályok alapján
        + könnyebben átlátható, nehezebben írható

feltétel a modellben teljesül → diagram létrejön

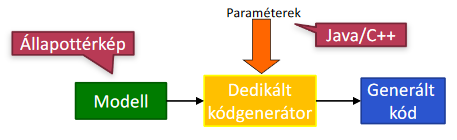
diagram változik → modell változik

* + - ***modellek validálása: szintaxisellenőrzés***
      * ***szintaktikai ellenőrzés***
        + modellező eszközök összekötik a logikailag egymásra épülő modellelemeket
      * *szintaxisvezérelt szerkesztő*
        + szerkesztés közben hiba   
          (**Couldn’t resolve reference**)
        + fejlett szerkesztőeszközök (pl. „tab” -os lehetőségek felkínálása)
        + kód és diagram együtt
        + programozás: szerkesztés közben **hibás**
        + modellezés: szerkesztés közben **helyes**
      * ***strukturált szerkesztő***
        + modell gráf vizsgálata
        + hibaminták keresése szerkesztés közben
      * Példa
        + elérhetetlen állapot *(Node is not reachable.)*
        + hiányzó kezdőállapot
        + holtpont
        + változó értékadások stb.

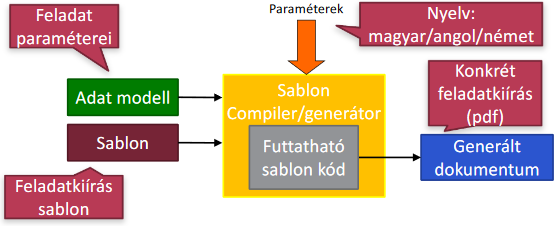
1. **Kódgenerálás**
   * **motiváció**: fejlesztési idő rövidítése
   * Példa
     + sakkjátékosok kezdeti gondolkodási ideje
       - kiinduló érték
       - idő változtatása
         * nem lehetséges
         * mindkét játékosnak egyszerre
         * játékosokra külön-külön
   * **előnyei**
     + helyes kódot generál 🡪 tanúsítvány
     + gombnyomásra generálódik a kód minden változtatás után azonnal
     + optimalizálható átláthatóságra, hatékonyságra
   * nehéz jól megírni (fentiek miatt)
   * **feladata**
     + modellnek megfelelő viselkedésű program automatikus előállítása
     + több megoldás létezik (tervezői döntések)
       - *interpretált*: modellt beolvassuk és végrehajtjuk
       - *generatív*: forráskód szintézise
       - programozási nyelvek: Java, C, C++
       - *optimalizálás*: memória vs. CPU, megfigyelhetőség vs. teljesítmény
     + dinamikus interpreter vs. kódgenerálás:



* + **típusok**
    - ***dedikált kódgenerátor*** (pl. Yakindu)
      * modellező eszköz része
      * tanúsítványozhatóság
      * alacsony testreszabhatóság

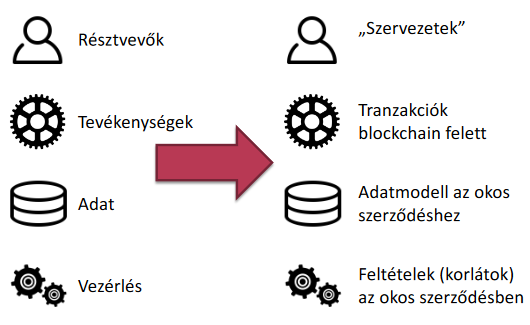


* + - ***sablon-alapú kódgenerátor***
      * gyorsabb fejlesztés
      * komplex változások az életciklus során
        + sablon és a modell függetlenül változik

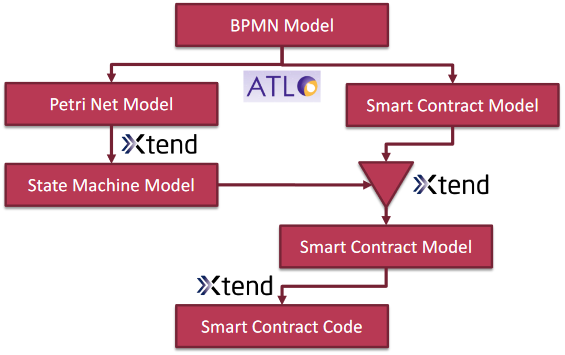


* + **funkciók**
    - hatékonyság vs. újrafelhasználhatóság, érthetőség
    - nyomonkövethetőség
      * adott kódrészlet mely modellelem miatt van ott
      * *inkrementalitás* támogatása: csak ott változzon a kód, ahol a modell
      * hatékony újragenerálás
  + **generált kód illesztése**
    - csak egy része az alkalmazásnak
    - kézzel nem módosítjuk
    - generált és kézzel írt kód külön file-ba/mappába
    - illesztés: *ragasztó kód* (glue code)
      * generált kód hívása
      * leszármazás generált kódból

1. **Üzleti folyamatok végrehajtása blockchain alapon**
   * **röviden**
     + elosztott, letagadhatatlan, többrésztvevős tranzakciók
   * **felhasználási területek**
     + (kriptovaluták)
     + árucsere
     + szállítmányok követése
   * **motiváció**
     + *modell alapú fejlesztés*
       - magasabb absztrakciós szint
       - produktivitás
         * modellek (szerződésminták) újrahasznosítása
         * tervezés egyszerűsítése
       - automatizált fejlesztés
       - minőségbiztosítás
         * modell validáció, modellellenőrzés, modell alapú tesztelés
     + „*blockchainesítés*”
       - meglevő megoldások (részben) blockchainre vitele
   * (BPMN elements in blockchain platforms)



* + **célok**
    - üzleti folyamatok alapján okos szerződések generálása
      * nem feltétlen a végrehajtó környezet szintjén
      * nehezen ellenőrizhető
      * kód és specifikáció összerendelése
    - a fejlesztés részleges automatizálása
      * „boilerplate” kód generálása
      * hibalehetőségek számának csökkentése
        + üzleti logika váza
        + meglévő lépések becsatolása (?)
  + **transzformációs folyamat**

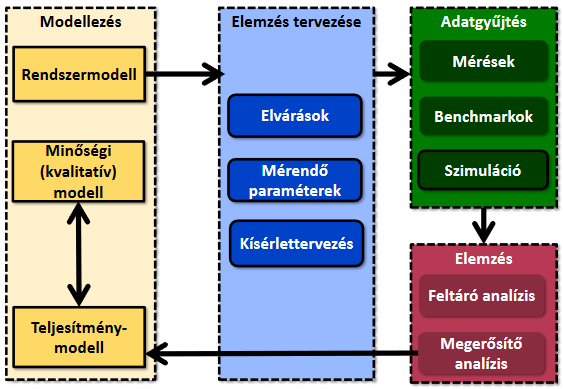


# **10. előadás: Modellek paraméterezése: regresszió, benchmarkok**



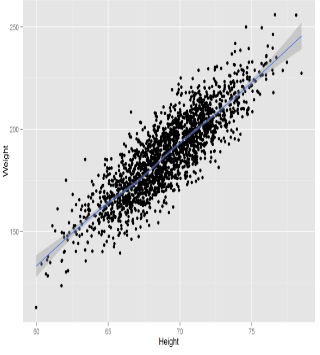
[**BACK**](#_top)

1. **Rendszermodelltől teljesítménymodellig**



* + **adatok hihetőség**
    - érzékenységvizsgálat
      * a modell kimeneti paraméterei mennyire érzékenyek a bementei paramétereinek változására
      * „parameter sweep”: egy paraméter vizsgálata adott tartományban
    - ***ökölszabály***: adatok hihetősége
      * mérés\_bizonytalansága (szórás) ~ mérések\_száma2
      * (kellő mennyiségű adat esetén) → valószínűség-számítás

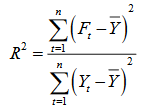
1. **Matematikai becslések: regressziós módszerek**
   * **regresszió (*f függvény)***
     + bemenet: az attribútumok értéke
     + kimenet: megfigyelések legjobb közelítése
     + „ökölszabály”
   * Példa
     + testtömeg/magasság együttes eloszlása



* + ***regressziós módszerek***
    - alapelv
      * véletlen változó
      * közelítés
      * hiba
      * jósolt esemény
      * megfigyelhető változók
    - *átlagos hiba (mean error)*
      * becsült érték
      * mért érték
    - ***lineáris regresszió***
      * egyszerű lineáris függvény illesztése az adatokra
      * *legkisebb négyzetek módszere*
        + keressük azokat az paramétereket, amelyekre
        + cél:
        + legjobban illeszkedő egyenes
        + DE: Anscombe’s quartet

minőségileg különböző adatok

azonos regressziós egyenes

* + - * *korrelációs együttható* (négyzete)
        + változó becsült és tényleges értékének kapcsolata
        + és közti érték

nincs kapcsolat

függvényszerű kapcsolat

(kapcsolat iránya)

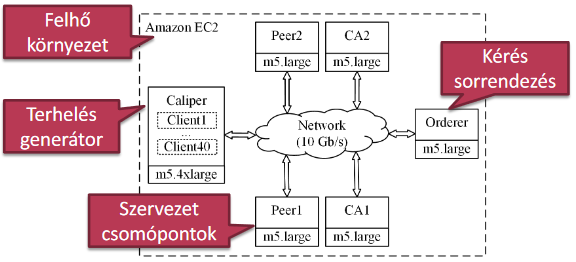
* + - * *nemlineáris módszerek*
        + exponenciális megközelítés
        + Web forgalom növekedéséhez jól illik
        + exponenciális terhelés becslőfüggvény
    - ***mozgó átlagok módszere***
      * rövid távú előrejelzésre jó
      * egyszerre egy értéket ad meg
      * becsült érték az utolsó érték átlaga
      * időpontban mért érték
      * becsült érték
      * tipikusan és között van (becslés hibája ne legyen túl nagy)
    - ***exponenciális csúszóablak***
      * egy értéket ad meg, előző méréseket átlagolva
      * későbbi mérés (és mérési hiba) nagyobb súllyal
      * rövid távú előrejelzésre
      * időpontra becsült érték
      * időpontban mért érték
      * mérési hiba periódusban
      * súlyozás gyakorlatban

1. **Benchmarking**
   * Definíció
     + ***benchmarkolás***
       - egy program (*programok, vagy más műveletek*) futtatása
       - szabványos tesztekkel vagy bemenetekkel
       - egy objektum relatív teljesítményének felmérése érdekében
     + ***ismételhetőség (repeatability)***
       - benchmarkot lehessen egymás után többször is futtatni
       - → mérési eredmények szórása csökkenthető legyen
     + ***reprodukálhatóság (reproducibility)***
       - benchmark legyen hasonló környezetben, hasonló eszközökkel megismételhető
     + ***érthetőség/általánosított felhasználói eset*** 
       - átlag felhasználó számára értelmezhető legyen az eredmény
     + ***relevancia (relevance)***
       - benchmarkban megvalósított terhelési profil hasonlít arra a valós terhelésre, amely alatt a rendszer teljesítményéről információt szeretnénk kapni
       - biztosítsuk, hogy
         * azt mérjük, amit kell
         * terhelésgenerálás jellege közelítse a valódi terhelést
         * minimalizáljuk a zavaró tényezőket

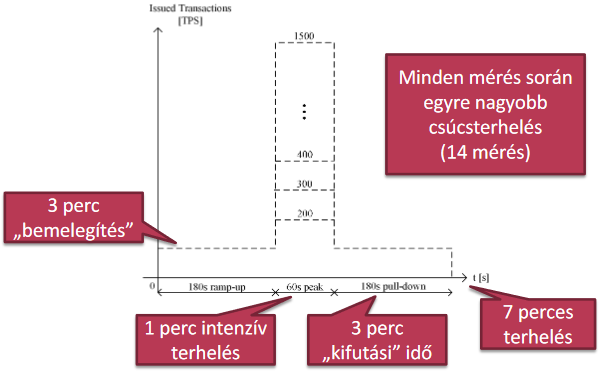
pl. gyorsítótárak ürítése, többi futó folyamat leállítása

* + - ***TPC-C***
      * *OLTP benchmark*
        + OLTP (Online Transaction Processing)
        + szoftverek és hardverek OLTP teljesítményét méri
      * *komplex benchmark*
        + több tranzakció típus
        + összetett adatbázis séma
        + széles skálán mozgó adathalmaz méret
      * *mért metrikák*
        + áteresztőképesség
        + ár-teljesítmény arány ( hatékonyság)
      * **mintaadatbázis**
        + nagykereskedelmi beszállító cég rendelései
        + 9 különböző tábla
      * **5 féle tranzakció típus**
        + rendelés, fizetés, szállítás, rendelés státusza, raktár állapota
        + frissítés, beszúrás, törlés, megszakítás
      * **ACID tranzakciók**
        + Atomicity, Consistency, Isolation, Durability
      * **változatos felhasználói kérések szimulációja**
        + tranzakció típusok kezdeményezése adott valószínűséggel
      * **időkorlát az egyes tranzakció típusokra**

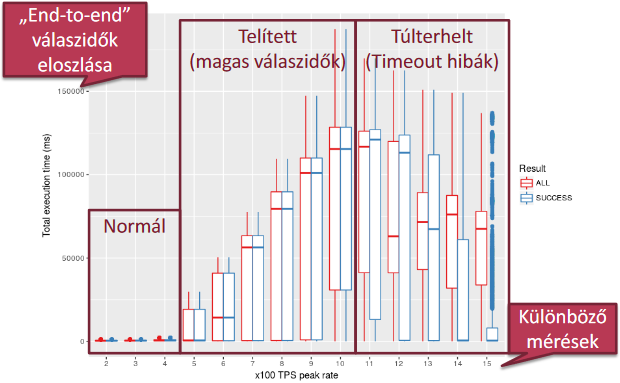
1. **Teljesítménytesztelés**
   * Definíció
     + ***Hyperledger Fabric***
       - konzorciális elosztott főkönyv technológia (DLT)
       - komplex infrastruktúra
         * szervezetek, főkönyvek, telepített okos szerződések



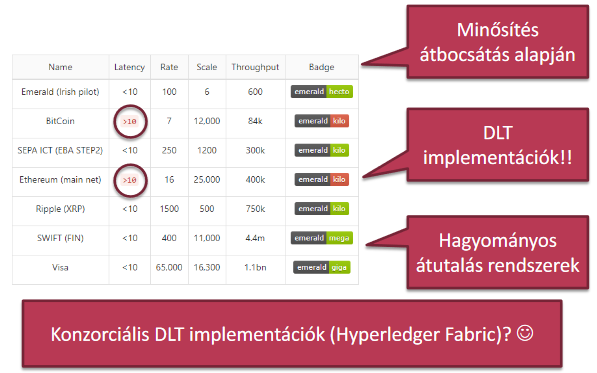
* + - ***terhelésprofil***



* + - ***válaszidők eloszlása***



1. **Benchmarkolás**
   * **egy jó benchmark ismérvei**
     + *gyártó-független*
       - különböző gyártók termékeinek összehasonlítása
     + *jól definiált cél*
       - valós felhasználási eseteket takar
     + *megismételhető*
       - a rendszer bármikor determinisztikusan újramérhető
     + *aktívan karbantartott*
       - követi az új technológia trendeket, felhasználási esetek evolúcióját
     + *nyílt szabványokon alapul*
       - szabadon elérhető, szakemberek által átvizsgált, az ipar által elfogadott
     + *egyszerű, általános metrikákat mér*
       - eltérő megvalósítású, azonos célú rendszerek összehasonlítása
   * ***Emerald SEPA ICT benchmark***
     + SEPA ICT: Single Payment Area, Instant Credit Transfer – Gyors Nemzetközi Átutalások
     + **Emerald alapelvek**
       - nem lehet egyszeres hibapont
       - a csomópontok földrajzilag elosztottak
       - egy minimum válaszidőt tartania kell
       - szabványos átutalás csomagok használata
       - minden csomópont küld és fogad utasításokat
     + **Emerald metrikák**
       - *késleltetés (latency)*
         * tranzakció nyugtázásáig eltelt idő
       - *ráta (rate)*
         * másodpercenként nyugtázott tranzakciók
       - *skálatfaktor (scale)*
         * tranzakciókat kezdeményező csomópontok száma
       - *átbocsátás (througput)*
         * átlagos ráta skálafaktor
     + **Emerald eredmények**



* + ***Cassandra Benchmark***

