Feladatgyűjtemény

Tartalomjegyzék

1. Struktúra alapú modellezés **1**
1.1. Struktúra modellezése gráffal . . . 1
1.2. Tulajdonságmodellezés . . . 1
1.3. Típusok modellezése . . . 2
1.4. Űrlap . . . 2

2. Állapot alapú modellezés, kooperáló viselkedésmodellek **3**
2.1. Közlekedési lámpa . . . 3
2.2. Háromrétegű architektúra . . . 3
2.3. Érintőképernyős billentyűzet . . . 3
2.4. „Mit ír ki?” . . . 4
2.5. Ablaktörlő . . . 4
2.6. Összetett rendszer modellezése . . . 4

3. Folyamat alapú modellezés **5**
3.1. Folyamat lefutása . . . 5
3.2. Vezérlési folyam (forráskód alapján) . . . 5
3.3. Folyamatmodell szöveges specifikáció alapján . . . 6

4. Modellek ellenőrzése **6**
4.1. Folyamat statikus analízise . . . 6
4.2. Dinamikus analízis teszteléssel . . . 7

5. Teljesítménymodellezés **8**
5.1. Zárthelyi megtekintése . . . 8
5.2. Diszk teljesítménye . . . 8
5.3. Kétrétegű architektúra . . . 8
5.4. Sziget közlekedési hálózata (∗ korábbi zárthelyi feladat) . . . 9
5.5. Tudásbázis (∗) . . . 9
5.6. Közösségi oldal . . . 9
5.7. Szerver teljesítménye – teljesítménymodellezés . . . 9

6. Adatelemzés **10**
6.1. Szerver teljesítménye – felderítő adatelemzése . . . 10
6.2. Képgaléria – adatelemzés . . . 10
6.3. Szenzorhálózat (korábbi zh feladat) – adatelemzés . . . 11
6.4. Szenzorhálózat (korábbi zh feladat) – teljesítményelemzés (∗) . . . 11

7. Követelménymodellezés **12**
7.1. Vasúti biztosítóberendezés követelményelemzése . . . 12
7.2. Rendszermodellezés tárgy követelményei . . . 13

8. Kísérlettervezés* **14**
8.1. Kísérlettervezés . . . 15
8.2. Kísérlet kiértékelése . . . 15

1. Struktúra alapú modellezés

Közösségi fuvarszolgáltatás

Közösségi fuvarszolgáltatást tervezünk, ahol bárki meghirdetheti a közeljövőben tervezett autós utazásait; mások pedig a rendszerünkön keresztül értesülhetnek erről, és utasként csatlakozhatnak (akár

csak egy rövidebb szakaszon is), ha beszállnak az üzemanyagköltségbe. Az egyes fuvarokat különböző fuvarszakaszokra osztjuk. Nem feltétlenül végig az autó gazdája fog vezetni, ez megbeszélhető, azonban minden fuvarszakasz során jelen kell lennie.
A szolgáltatásunk eddig zárt tesztüzemben futott, a fuvarokat ad-hoc szerveztük, és az adatokat nem rögzítettük szisztematikus módon. Hamarosan szeretnénk nyilvánosan is elindítani a szolgáltatást. A webes felületen az utazásszervezéssel kapcsolatos információkat elérhetővé kell tenni, ezért valamilyen módon nyilván kell ezeket tartanunk.

1.1. Struktúra modellezése gráffal

Szeretnénk a rendszer mögöttes adatmodelljét megtervezni. Ehhez az eddigi fuvarok tapasztalatai alapján összeállítottunk néhány tipikus forgatókönyvet.

a) Anna Szombathelyről autóutat tervez Győr és Budapest érintésével Debrecenbe. Balázs Győrből indít fuvart Budapestre, majd onnan tovább Kecskemétre. Alkossunk gráfmodellt a szövegben megadott viszonyok alapján!
b) Dani győri, és nincs kocsija. Milyen gráfelméleti művelet ad választ arra, hogy a felajánlott fuvarokba utasként becsatlakozva mely városokba lehet Győrből eljutni? (Feltehetjük, hogy az autósok az utazás időpontját tekintve rugalmasak.)
c) Csilla úgy döntött, hogy Anna autójával fog utazni Szombathelytől Budapestig; Dani Győrből Kecskemétre kért fuvart. Mivel Anna előző este sokáig dolgozott, az indulás után aludna, ezért úgy beszélték meg, hogy Budapestig Csilla vezet. Balázs végig maga vezet. Alakítsuk át a gráfot olyan módon, hogy kifejezze ezt a tudást!
d) Balázsnak nem ez az első közösségi fuvarja; korábban Kecskemétről hirdetett utazást Győrbe Budapesten keresztül. Módosítsuk a gráfot olyan módon, hogy megjelenjen benne ez az információ is!
e) Milyen művelettel kaphatunk a teljes tudást reprezentáló gráfból egy egyszerűsített nézetet, amelyik csak a hirdetőket, az utazásaikat, és az utazásokat alkotó szakaszokat mutatja?

1.2. Tulajdonságmodellezés

Az eddigi fuvarok során összegyűjtöttünk néhány adatot. Ezeket az alábbi táblázat tartalmazza.

ért. száma	ért. összege	kategória	név	jelszó	rendszám	dohányos	A/C	díjfizetés	jog.sz.
6	24	kocsi			`ABC-123`		nincs		
17	71	személy	Anna	qwe		nem			`KL2048`
16	49	személy	Balázs	pass		igen			`MN4096`
14	45	személy	Csilla	12345		igen		kártya	
1	5	személy	Dani	barát		nem		utalás	
0	0	kocsi			`DEF-456`		van		
7	31	személy	Eszter	2501		nem		kártya	
2	8	személy	Feri	almafa		nem		Bitcoin	

1. táblázat. A `Fuvarok` táblázat.

Egy-egy fuvar után 1–5 skálán lehet értékelni, hogy az útitársak és az autó mennyire járultak hozzá a kellemes utazáshoz. A fenti táblázat többek között a begyűjtött értékeléseket is mutatja.

a) Amikor valaki a lehetséges fuvarok közül válogat, az útitársak és az autó értékelése mellett a légkondicionáló megléte, ill. a dohányzás is fontos szempont lehet. A döntéshez azonban nem kell, és nem is szabad a felhasználóknak megismernie a sofőrök jelszavát és jogosítványszámát, valamint hogy a többi útitárs a fuvar megegyezéses díját milyen módszerrel rendezi a cég felé. Milyen (tulajdonságmodellezésnél megismert) művelettel kapható meg az ehhez a nézethez szükséges információ?
b) A rangsorolás az értékelések pontszámának nem az összege, hanem az átlaga alapján történik. Milyen művelettel bővíthető ki a tulajdonságmodell ezen számítás eredményével?

c) Eszter és Feri együtt keres fuvart. Eszter lehetőleg 4 pont feletti értékelésű autót szeretne. Feri csak légkondicionálóval rendelkező kocsik közül hajlandó válogatni. Milyen műveletekkel kaphatóak meg a választási lehetőségeik?

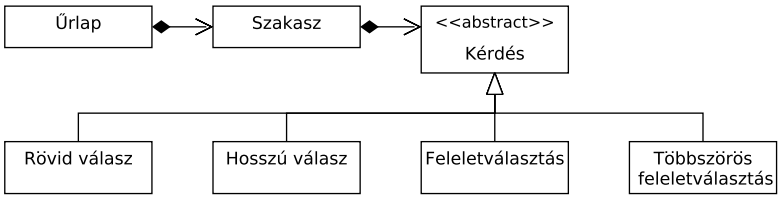
1.3. Típusok modellezése

A szolgáltatáshoz szeretnénk adatbázist tervezni. Ehhez fontos, hogy megkülönböztessük a típusokat a rendszerünkben és keressünk validációs szabályokat.

a) Milyen alapvető elem- és kapcsolattípusokat sugallnak a gráfmodellben látható megadott viszonyok? Ábrázoljuk típusgráffal!
b) Milyen típusokba sorolhatóak a táblázatban szereplő elemek a rajtuk értelmezett jellemzőik köre és a kapcsolataik alapján?
c) Definiáljunk egy típushierarchiát az előző pont alapján!
d) (*Kiegészítő feladat*) A típusgráf, a típushierarchia és a jellemzők értelmezési tartománya alapján rajzoljunk metamodellt! Milyen további megkötésekkel (jólformáltsági kényszerekkel) egészíthetjük ki?

1.4. Űrlap

Az alábbi metamodell alapján készítsünk egy űrlapot, amelynek segítségével az utasok egy-egy utazást követően visszajelzést adhatnak a sofőrről. Nem szeretnék túl sok időt elvenni az utastól, ezért a legtöbb információt eldöntendő, illetve feleletválasztós kérdések formájában gyűjtjük be. Az utasnak lehetősége van arra is, hogy saját szavaival összefoglalja tapasztalatait egy rövid szöveges vélemény keretében.



a) Milyen információra van szükségünk a sofőr azonosításához?
b) Gyűjtsünk össze pár kérdést, csoportosítsuk őket, majd adjuk meg az elkészült űrlapnak egy modelljét. (Ez már példánymodell lesz a későbbiekben.)
c) Top-down vagy bottom-up tervezést alkalmaztunk?
d) (*) Ha az utas az ötös skálán való értékelésnél hármasnál rosszabbat ad a sofőrre, akkor mindenképpen szeretnénk szöveges véleményt. Hogyan tudnánk ezt a modellben megfogalmazni (és melyikben)?

Kiegészítő feladat: megvalósítás programmal

a) Készítsünk olyan adatstruktúrát (tetszőleges programozási nyelven), amely egy ilyen fuvarszervező információtartalmának reprezentálására szolgál!
b) Egészítsük ki olyan eljárással (metódussal) a programot, amely képes felsorolni, hogy egy megadott városból hova juthatunk el (átszállás nélkül) a meghirdetett fuvarokkal!
c) Készítsük el az előző eljárás okosabb változatát, amelyik igény szerint elkerüli azokat a fuvarokat, ahol legalább egy szakaszon dohányzó útitársunk lenne!

2. Állapot alapú modellezés, kooperáló viselkedésmodellek

2.1. Közlekedési lámpa

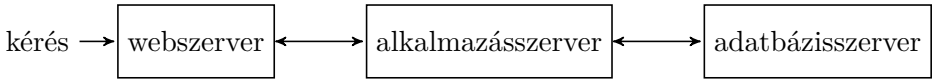
Közlekedési lámpát vezérlő elektronikát tervezünk.

a) Készítsük el egy egyszerű piros–sárga–zöld közlekedési lámpa olyan állapotterét, amely kellően finom ahhoz, hogy a lámpák vezérlését ez alapján lehessen végezni! Győződjünk meg arról, hogy az állapottér kizárólagos és teljes!

b) A három égőnek külön-külön mi az állapottere? Milyen absztrakciós viszony áll fent a lámpa és az egyes égők állapottere közt? Hogy viszonyul a lámpa állapottere a három égő állapotterét reprezentáló három állapotváltozó direkt szorzatához?
c) Mik az érvényes állapotátmeneti szabályok? Készítsük el az (egyszerű) állapotgráfot!
d) Hogyan fejezhető ki ugyanez a működés tömörebben hierarchikus állapotokkal?
e) Amikor a lámpa elektromos fogyasztását vizsgáljuk, csak az érdekel, hogy a három égőből hány ég egyszerre. Absztraháljuk az állapotgépet úgy, hogy az állapotokat csak a fogyasztásuk különböztesse meg!
f) A piros jelzés végén van egy olyan időszak, amikor a merőleges gyalogosátkelő zöld lámpája már villog. Finomítsuk úgy az (absztrakció előtti) állapotgráfot, hogy ez az állapot elkülöníthető legyen!
g) Egy út mentén 10 jelzőlámpa található, egyenként 4 állapottal. Legfeljebb hány állapota lehet a teljes rendszernek? Várhatóan kell-e minden állapotot kezelnünk?

2.2. Háromrétegű architektúra

Egy informatikai rendszert szeretnénk modellezni, melyet *háromrétegű architektúra* valósít meg az alábbiak szerint:



A használt fogalmak:
- **Adatbázisszerver**: hosszútávon tároljuk az adatokat
- **Alkalmazásszerver**: az üzleti logikáért felelős alkalmazást futtatja
- **Webszerver**: megjelenítésért felelős, generálja a HTML oldalakat

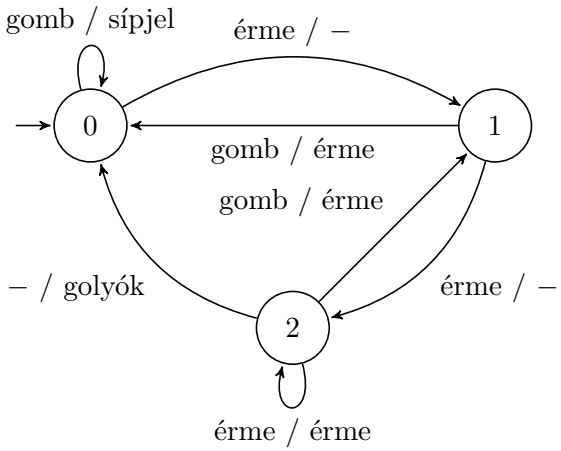
Háromrétegű kiszolgáló infrastruktúránk viselkedésének modellezésére megfelelő állapotterek-e az alábbiak?
a) { Webszerver dolgozik, Alkalmazásszerver dolgozik, Adatbázisszerver dolgozik }
b) { Leállítva, Tétlen üzemel, Aktívan dolgozik }
c) $\mathbb{N}$ (mint a pillanatnyilag feldolgozás alatt álló kérések száma)
d) $\mathbb{Z}$ (mint a pillanatnyilag feldolgozás alatt álló kérések száma)
e) { A kérés feldolgozása még nem kezdődött el, A szerverek épp dolgoznak a kéréssel, A kérés kiszolgálása befejeződött }
f) { Igaz } $*$

2.3. Érintőképernyős billentyűzet

Modellezzük állapotgéppel egy mobiltelefon érintőképernyőjére tervezett, az előadáson is bemutatott virtuális billentyűzetet! A billentyűzeten egyszerre vagy a kisbetűk, vagy a nagybetűk, vagy a számok és fontosabb szimbólumok, vagy ritkább szimbólumok láthatóak. Az elsődleges üzemmódváltó gomb a betűk és a számok/szimbólumok beírása között vált, a másodlagos üzemmódváltó pedig ezen kategóriákon belül. Létezik továbbá egy olyan nagybetűs állapot is, amely egy betű leütése után automatikusan kisbetűsre vált. Vegyük figyelembe a bal felső gombot (`q/Q/1/=`), ill. a két üzemmódváltó gombot mint bemenetet, és a szövegmezőbe begépelt karaktereket mint kimenetet!

2.4. „Mit ír ki?”

Tekintsük az M állapotgépet!
Megjegyzés: a „$-$” jel a bemeneti jel pozíciójában spontán állapotátmenetet jelöl, a kimenet helyén pedig kimenet hiányát, egyik esetben sem a digitális áramkörtervezésben használt *don't care* szimbólum.

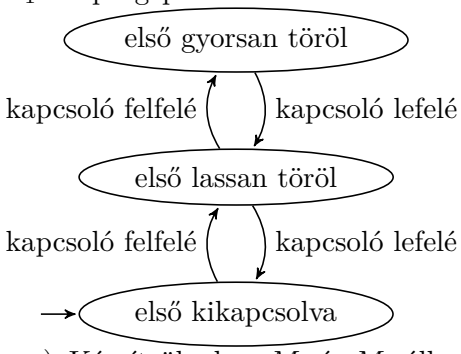


a) Milyen valós rendszer lehet az M állapotgép mögött, hogyan működik?
b) Determinisztikus-e ez a viselkedésmodell? Hozzávehető-e, ill. elhagyható-e egyetlen állapotátmeneti szabály, hogy ez megváltozzon?
c) Absztraháljuk az M állapotgépet a $\{\{0\}, \{1, 2\}\}$ állapotpartíció szerint!
d) Hol és milyen jellegű nemdeterminizmus figyelhető meg az így kapott absztrakt modellen?

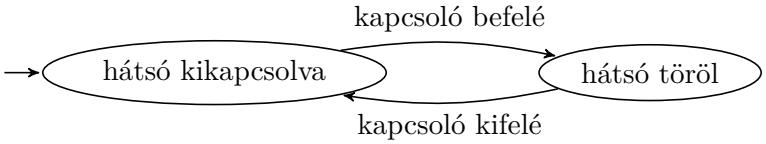
2.5. Ablaktörlő

Egy autóban az első ablaktörlőnek három állapota van (*első kikapcsolva*, *első lassan töröl*, *első gyorsan töröl*), a hátsó ablaktörlőnek kettő (*hátsó kikapcsolva*, *hátsó töröl*). Az első ablaktörlő működését az M_1 állapotgép, a hátsóét az M_2 állapotgép modellezi.

M_1 állapotgép:



M_2 állapotgép:



a) Készítsük el az M_1 és M_2 állapotgépek aszinkron szorzatát!
b) Hány állapota és átmenete van az így kapott modellnek?
c) (Kiegészítő feladat) Kifejezhető-e a kapott állapotgépen, ill. a komponensre vetített modellek segítségével olyan kocsi, ahol a hátsó ablaktörlő csak akkor kapcsolható be, ha megy az első is?

2.6. Összetett rendszer modellezése

Felhő alapú adattárolást modellezünk (ld. Dropbox, Google Drive, Tresorit), egyetlen állományra szorítkozva. Az állománynak a szerveren és a kliensnél (pl. laptop) is elérhető egy-egy replikája, kezdetben azonos tartalommal. A fájl módosításai *szinkronizálás* során továbbítódnak a példányok között. Ha szinkronizáció előtt mindkét példányt módosítják, akkor *ütközés* lép fel, amelyet a felhasználónak kell feloldania a kliensen.

Lokálisan a kliensnél, illetve (pl. másik kliens tevékenységének hatására) módosulhat a szerveren is. Felhasználói utasításra, valamint időről időre spontán módon a kliens és a szerver szinkronizálhat; ilyenkor az esetleges módosítás eljut a másik példányhoz is, és újra *azonos* lesz a két másolat. Ha a legutóbbi szinkronizáció óta egymástól függetlenül mindkét replikát módosították, akkor viszont konfliktushelyzet (ütközés) áll fenn. Ilyenkor a kliens a saját és a szerverről letöltött változatot összehasonlítja, és a felhasználóra bízza az ütközés feloldását.

a) Modellezzük először a kliens (részleges) működését állapotgéppel! A kliens kezdetben *azonos* állapotú (a lokális fájlmásolat egyezik azzal, ami a szerveren a legutóbbi szinkronizációkor volt / lett), ám *írás* bemenet hatására a *piszkos* állapotba kerül (és további *írás* hatására is ottmarad). Az *elvetés* bemenet hatására tetszőleges állapotból újra *azonos* állapotba kerül.

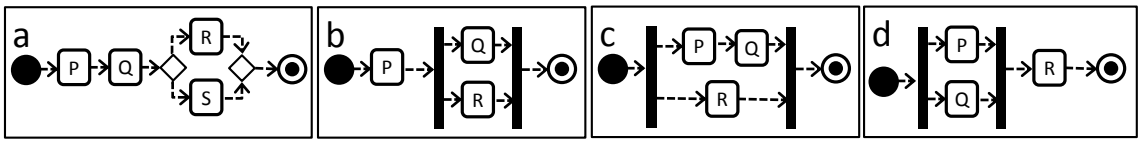
b) A szerver lehetséges állapotai (csupán az adott klienssel való szinkronizációt vizsgálva) az *azonos* és a *frissült*. Előfordulhat, hogy a megfelelő írási jog birtokában egy másik felhasználó (vagy ugyanazon felhasználó egy másik kliens, pl. a telefonja segítségével) frissíti a szerveren található állományt. A szerver kezdetben *azonos* állapotban van.
c) Ha a szerver *azonos* állapotban van, akkor a kliens a *szinkronizálj* bemenet hatására feltölti az esetleges lokális módosításokat a szerverre, és szintén *azonos* állapotba kerül. A *szinkronizálj* bemenetet a szerver is megkapja. Hol kooperál a két állapotgép?
d) Ha a szerver *frissült* állapotban van, akkor a kliensnek adott *szinkronizálj* bemenet hatására a szerver *azonos* állapotba kerül; a kliens pedig *azonos* állapotból nem mozdul, de *piszkos* állapotból *ütközés* állapotba megy. Mit jelent ez? Mi történjen az ütközés állapotban? Hol kooperál a két állapotgép?
e) A kliens időnként magától is szinkronizál a szerverrel, felhasználói bemenet nélkül. Mit jelent ez? Hol kooperál a két állapotgép?
f) Fejtsük ki a teljes összetett állapotgépet a vegyes szorzatban részt vevő két állapotgép alapján.
g) (Kiegészítő feladat.) Ebben a modellben a szerver és a kliens közvetlenül figyelembe tudják venni egymás belső állapotát, és a szinkronizáció is pillanatszerűen végbemegy közöttük. Egy valódi elosztott rendszerben azonban üzenetváltással kell a kliens és a szerver közötti kommunikációt megvalósítani; a küldés és a válasz megérkezte között pedig huzamosabb idő eltelhet. Gondoljuk végig, hogy lehetne finomítani a modellt, hogy ezeket a részleteket is tükrözze!

3. Folyamat alapú modellezés

3.1. Folyamat lefutása

Egy folyamat végrehajtása során az összes lépést megfigyeltük. A következő eseménysor bekövetkeztét észleltük:

> Folyamat indul, P elkezdődik, P befejeződik, Q elkezdődik, R elkezdődik, Q befejeződik, R befejeződik, Folyamat befejeződik.



Az a, b, c, d folyamatmodellek közül melyek lehetnek helyes modelljei a rendszernek?

3.2. Vezérlési folyam (forráskód alapján)

Tekintsük az alábbi C nyelvű függvényt.

```
1  unsigned long long fibonacci(int n)
2  {
3     if (n <= 0) {
4        return 0;
5     } else if (n == 1) {
6        return 1;
7     } else {
8        unsigned long long a = fibonacci(n - 1);
9        unsigned long long b = fibonacci(n - 2);
10       return a + b;
11    }
12 }
```

a) Milyen vezérlési folyamot határoz meg a függvény?
b) Ellenőrizzük, hogy jólstrukturált-e ez a folyamat!
c) Azonosítsuk az adatfüggőségeket (adatáramlást) a tevékenységek között!
d) Ha a programozási nyelv vagy a futtatókörnyezet megengedi, hol van lehetőség párhuzamosításra?
e) (Kiegészítő feladat.) Mi biztosítja azt, hogy a függvény előbb-utóbb terminál?

3.3. Folyamatmodell szöveges specifikáció alapján

Egy nagy szoftveralapítvány kódtára (pl. Git) számos nyílt forráskódú szoftver fejlesztésének ad otthont. A megbízható belsős fejlesztőkön kívül külsősök is gyakran küldenek be hibajavításokat vagy újonnan megvalósított képességeket. Oda kell figyelni arra, hogy a kiadott szoftverben csak jogszerűen (pl. munkaadó beleegyezésével) bekerült forráskód szerepeljen.

a) Ha egy fejlesztő hozzá szeretne járulni egy projekthez az általa készített forráskóddal, akkor a saját státuszától függő lépéseket kell tennie. Belsős fejlesztők közvetlenül írhatnak a kódtár adott projekt részére fenntartott területére. Külsős fejlesztőknek először átvizsgálásra (*code review*) be kell nyújtaniuk a kódjukat; ezután egy belső fejlesztőnek ellenőriznie kell azt, és utána vagy elutasítania, vagy elfogadnia. Ha a kívülről érkező kód egy bizonyos küszöbértéknél rövidebb (pl. néhány soros hibajavítás), akkor az elfogadás után a készítőjének már csak egy rövid hozzájárulási nyilatkozatot kell tennie, hogy beolvasztható legyen a kódtárba.
   A nagyobb lélegzetű külső hozzájárulások (pl. egy teljesen új modul beépítése) esetében azonban az elfogadást követően az alapítvány jogi osztálya egy külön adminisztratív eljárásban tisztázza a változtatások szellemi tulajdonának jogállását, és csak ennek sikeres lezárása után olvaszthatja be a belső fejlesztő a kódot. Frissen indított, első hivatalos kiadásuk előtt álló projekteknél itt tesznek egy kivételt: az elfogadott külső hozzájárulás kódtárba beolvasztásával nem kell megvárni ezt az adminisztratív eljárást. Készítsünk folyamatmodellt az itt leírt tevékenységekből!
b) A szoftver fejlesztési projektje abból áll, hogy újabb és újabb módosításokat végeznek a forráskódon, amíg a projekt vezetése úgy nem látja, hogy a szoftver kellően stabil egy hivatalos kiadáshoz (*release*). Amikor eljött ez a pont, akkor közzétesznek egy új stabil verziót a szoftverből, majd ismét a fejlesztésen a sor, és így tovább. Készítsünk folyamatmodellt az itt leírt tevékenységekből!
c) (Kiegészítő feladat.) Ellenőrizzük, hogy jólstrukturáltak-e a folyamataink!
d) (Kiegészítő feladat.) Milyen viszonyban állnak egymással az a) és b) feladatokban elkészített folyamatmodellek?

Kiegészítő feladat: Adatfolyamháló

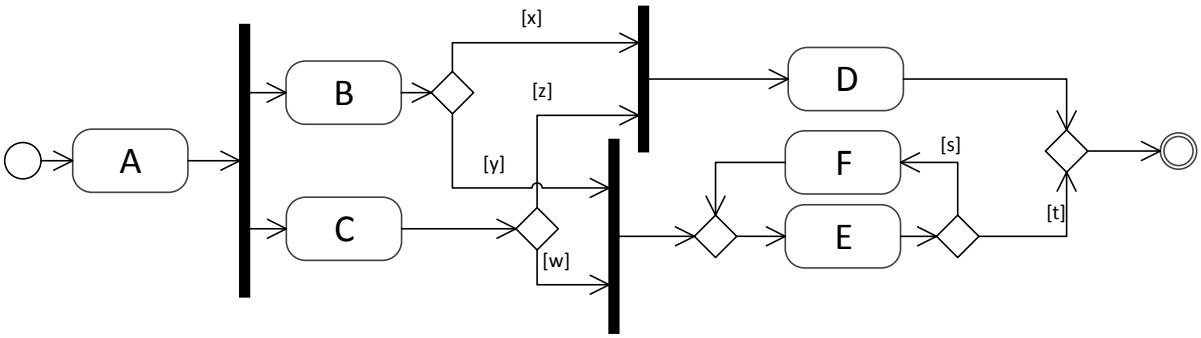
Elakadásjelző háromszögeket előállító gyárunkat adatfolyamhálóval modellezzük. A háló kezdetben két csomópontot tartalmaz. Az első csomópont egy gép, amely fényvisszaverő oldallapokat állít elő, és a futószalagra helyezi őket. A második csomópont az összeszerelő gép, amely a futószalagról felveszi a lapokat; ezen kívül időnként egy összeszerelt háromszöget bocsájt ki az egész háló kimenetén.

a) Készítsük el a feladat adatfolyamháló modelljét. Szorítkozzunk állapotgép jellegű csomópontokra.
b) Finomítsuk a modellt a következőképp: az első gép időnként deformált oldallapokat gyárt.
c) Finomítsuk tovább a modellt a következőképp: az összeszerelő gép az eredeti funkcionalitás elé kapcsolva tartalmaz egy bevizsgáló berendezést is, amely képes kidobni a deformált lapokat (az ép lapokat továbbengedve).
d) Végül finomítsuk tovább a modellt a következőképp: az összeszerelő gép (a selejtes lapok kiszűrése után) mindig bevár három fényvisszaverő lapot, és belőlük szerel össze egy háromszöget.

4. Modellek ellenőrzése

4.1. Folyamat statikus analízise

Ellenőrizzük az alábbi folyamatmodellt.



a) Milyen feltételek mellett teljesen specifikált a folyamat?
b) Milyen feltételek mellett determinisztikus is a folyamat?
c) Milyen feltételek mellett holtpontmentes is a folyamat?
d) Milyen további feltételek mellett termináló a folyamat?
e) Jólstrukturált-e a folyamat? Ha nem, hogyan lehetne azzá tenni? Segít-e ez a problémákon?

4.2. Dinamikus analízis teszteléssel

Az `f()` függvénnyel szemben a következő *követelmény*einek vannak:
R1 Az `f()` függvénynek minden végrehajtása során legalább egyszer outputot kell kiadnia.
R2 Az `f()` függvénynek tetszőleges inputsorozat esetén terminálnia kell.
R3 Az `f()` függvény végrehajtása során kiadott legutolsó output értéke kötelezően 0.
A függvény egy lehetséges megvalósítását adja meg az alábbi C nyelvű kódrészlet:

```c
int readInput();
void writeOutput(int out);

void f() {
  int x = readInput();
  int y = readInput();
  int z = x + y;
  writeOutput(x * y);
  while (x > 0 && y > 0) {
    if (1 == readInput() % 2) {
      y--;
      z--;
    } else {
      x--;
      y++;
    }
    writeOutput(z + x * y * y - x - y);
  }
}
```

A következő lépések során ellenőrizzük a függvény működését!
a) Ábrázoljuk folyamatmodellként `f()` vezérlési folyamát!
b) Miért lehetünk biztosak az R1 teljesülésében?
c) Miért lehetünk biztosak az R2 teljesülésében?
d) (Kiegészítő feladat.) Építsünk olyan állapotgépet, amely az `f()` függvénnyel ekvivalens módon működik. Modellezzük a `readInput()` hívásokat input csatornaként, valamint a `writeOutput()` hívást output csatornaként. Az `f()` függvény terminálását modellezzük úgy, hogy az automata ad egy speciális outputot, és átmegy egy nyelő (kimenő átmenet nélküli) állapotba.
e) Az R3 követelményt teszteléssel ellenőrizzük. A $t_1 = \langle 2, 3, 5, 7, 11, 13, ... \rangle$ input szekvencia a tesztesetünk. Detektálunk-e hibát?
f) Számítsunk *utasításszintű tesztfedést* a programkódon, vagyis hogy az utasítások mekkora hányadát járja be a tesztelt függvény a teszteset végrehajtása során! Hogy jelenik meg ez a mérőszám a vezérlési folyamon?
g) Az R3 követelményhez a $t_2 = \langle 1, 2, 4, 1, 2, 4, ... \rangle$ input szekvencia a második tesztesetünk. Detektál-e hibát ez a teszteset? Mekkora a két tesztből álló tesztkészlet együttes utasításfedése?
h) (Kiegészítő feladat.) Milyen tesztfedettségi metrika számítható a korábban megépített állapotgép alapján?

i) Készítsünk olyan *tesztorákulum* állapotgépet, amely `f()` input és output szekvenciái és terminálása alapján el tudja dönteni, hogy az adott lefutás során az R3 követelmény sérült-e! Hogy viselkedik az orákulum a fenti tesztinputra?
j) Adjunk meg egy tesztesetet, amely kimutat egy hibát a programban! Milyen elv alapján sejthettük volna meg, hogy a korábban összeállított tesztkészletünk kiegészítésre szorul?
k) (Kiegészítő feladat.) Vegyük hozzá a tesztkészlethez a $t_3 = \langle 0, 1, 2, 3, 4, 5, ... \rangle$ és $t_4 = \langle 1, 2, 3, 4, 5, 6, ... \rangle$ input szekvenciákat mint további teszteseteket! Detektálunk-e hibát? Hogyan változnak a tesztfedési számok?
l) (Kiegészítő feladat.) Határozzuk meg, hogy pontosan milyen input szekvenciák esetén sérül R3, és javasoljunk hibajavítást!

5. Teljesítménymodellezés

5.1. Zárthelyi megtekintése

A zárthelyik megtekintése során a hallgatóknak lehetőségük van reklamálni esetleges javítási hibák miatt. Sikeres reklamáció esetén a pontszámuk módosításra kerül. Az első nagyfeladatból (F1) óránként 10 darabot képes átnézni egy javító, a második nagyfeladatból (F2) pedig 20 darabot. Mindkét feladathoz tartozik 1-1 javító, akik az adott feladatot javították. A továbbiakban készítsünk minden kérdéshez egy-egy folyamatmodellt és határozzuk meg, hogy óránként hány hallgató dolgozatát sikerül átnézni az egyes esetekben!

a) A hallgatók először az F1, majd az F2 feladatot nézetik át a javítóval.
b) A leleményes hallgatók a két feladatot külön-külön már egyszerre két javítónak adják oda, mivel külön lapra voltak írva. Mit nyerünk a párhuzamosítással?
c) A nagy tömeg miatt a hallgatók csak az egyik feladatukat nézetik át, mégpedig azt, amelyiknek a javítója éppen szabad.
d) Híre ment, hogy a második feladat javítója sokkal kevésbé szigorú, így a hallgatók 80%-a inkább kivárja ennél a javítónál a sort. A maradék 20% a másik javítónál reklamál az első feladattal kapcsolatban.
e) A hallgatók 10%-ának a reklamáció után már csak 1-2 pont kellene a jobb jegyhez, ezért újra és újra megpróbálkoznak a reklamációval. Feltételezhetjük, hogy a hallgatók az a) részben leírt reklamációs stratégiát használják.
f) Mi történne másként, ha bármelyik javító bármelyik feladatot hajlandó átnézni (de az egyes feladatok átnézése változatlan ideig tart), és így szeretnék a hallgatók az eredeti tervnek megfelelően először az F1, majd az F2 feladatot átnézetni a javítóval?

5.2. Diszk teljesítménye

Egy diszk 50 kérést szolgál ki másodpercenként. Minden kérés kiszolgálása 0,005 másodpercet vesz igénybe. A rendszerben nincs átlapolódás.

a) Mekkora a maximálisan kiszolgálható terhelés (érkezési ráta)?
b) Mekkora a kihasználtság?

5.3. Kétrétegű architektúra

Adott egy webszerver (WS) és két fürtözött adatbázisszerver (DB1, DB2). A két adatbázis szerver közt súlyozott round robin terheléselosztás alapján választunk, 1:2 arányban. Minden felhasználói kérés kiszolgálása során mindkét fajta erőforrást használjuk. A csúcsidőszakban 30 percig monitorozzuk a rendszert, ezalatt 9000 kérést szolgál ki. A szerveken mért foglaltsági idők: WS – 1350 s CPU idő; DB1 – 810 s, DB2 – 1320 s diszk IO idő.

a) Készítsünk folyamatmodellt a kérések feldolgozásáról a szöveg alapján!
b) Mekkora az egyes szerverek jelenlegi átbocsátása?
c) Mennyi időt töltenek egy-egy hozzájuk beérkezett kérés kiszolgálásával a szerverek?
d) Mekkora a rendszer maximális áteresztőképessége?
e) Miért nem egyféle foglaltsági időt vettünk figyelembe a két erőforrástípusnál?
f) Hol csal még így is a modell?

5.4. Sziget közlekedési hálózata (∗ korábbi zárthelyi feladat)

Egy sziget lakói minden reggel munkába menet átkelnek a szigetet ölelő tavon. Észak felé híd vezet, dél felé autóskomp. Az irányonként egysávos híd 200 m hosszú, és 60 km/h sebességgel szabad rajta haladni, a követési távolság (hátsó lámpától hátsó lámpáig 30 m) betartása mellett. A négy komphajó egyenként 15 percenként teszi meg a sziget-szárazföld-sziget kört, és így óránként négyen együtt legfeljebb 800 autót tudnak átvinni a szárazföldre.

a) Mekkora a híd átbocsátóképessége (észak felé)?
b) Hány autó fér el egy kompban?
c) A reggeli csúcsforgalomban mekkora a szigetet elhagyó két útvonal együttes átbocsátóképessége?
d) Ha délben a szárazföldi főutat baleset miatt lezárták, és a szigeten keresztül (a hídon, majd a kompon átkelve) terelik a forgalmat, mekkora a terelőútvonal átbocsátóképessége?
e) Valamelyik reggel 7:00 és 8:30 között 900 autó hagyta el a szigetet komppal. Mennyi volt ebben az időszakban a kompok átbocsátása és kihasználtsága?
f) A fenti mérésben átlagosan hány autó állt sorba egyszerre a parton, ha az autók jól időzítve, átlagosan fél perccel a beszállásuk előtt érkeztek kompkikötőhöz?

5.5. Tudásbázis (∗)

Vállalatunk nyilvános szakmai tudástára egymásra is hivatkozó szócikkeket kínál a cég termékeit világszerte használó ügyfeleknek. Egyetlen szócikk lekérésének kiszolgálásához a szervert átlagosan 60 ms-ig veszi igénybe. A szócikk megtekintése után az olvasó csak az esetek 30%-ában hagyja el az oldalt, többnyire ugyanis egy újabb szócikkre mutató hivatkozásra kattint.

a) Egy olvasó összes tudásszomjának kielégítéséhez átlagosan mekkora szerveridő szükséges?
b) Tekintsük úgy, hogy az egyes kérések a szerveren nem párhuzamosíthatóak. Óránként hány egyedi látogatót képes kiszolgálni a szerver?

5.6. Közösségi oldal

Internetes közösségi oldalt működtetünk. Az utóbbi időben számottevően népszerűbb lett az oldal, de ezáltal a válaszidő is kellemetlenül megnőtt. Az üzleti cél, hogy csúcsidőszakban egyszerre 1500 felhasználót átlagosan négy másodperces válaszidővel szolgáljon ki a honlap.

a) Minimálisan mekkorára kell tervezni a kiszolgáló infrastruktúra átbocsátóképességét, ha az azon kívüli késleltetés (hálózati forgalom, HTML megjelenítés a kliensoldalon) egy másodpercnek becsülhető?
b) Az újratervezett weboldalon a mérések szerint egyetlen kérés kiszolgálása átlagosan 20 ms CPU-időt igényel a webszerveren, és 12,5 ms erejéig foglal le egy adatbázisszervert. Jelenleg 15 webszerver fogadja a kéréseket és az adatbázis 5 kiszolgálóra van replikálva. Lineáris skálázhatóságot feltételezve, milyen számítógépből és mennyit kell még legalább venni a fenti cél eléréséhez?
c) (∗) A kibővített rendszerben mekkora lesz az egyes szervertípusok kihasználtsági aránya? Ha az a cél, hogy még a csúcsidőszakban is legfeljebb 50%-os legyen a kihasználtság, meddig kellene még bővíteni a rendszert?
d) Tekintsünk csak 2 db webszervert és 3 db adatbázis szervert. Készítsünk állapot alapú modell(eke)t, amely(ek) az infrastruktúra erőforrásait modellezi(k) az elérhetőségeik (szabad/foglalt) szerint. Milyen tervezői döntésekkel szembesülünk? Mik az egyes lehetőségek előnyei és hátrányai?

5.7. Szerver teljesítménye – teljesítménymodellezés

Egy szerveren az alábbi teljesítményjellemzőket mértük:

Mintavétel időpontja [ms]	500	600	700	800	900
Utolsó 100ms alatt feldolgozott kérések száma [darab]	11	12	21	18	20
Utolsó 100ms átlagos kiszolgálási ideje [ms]	15	20	21	25	27
Utolsó 100ms CPU kihasználtság [%]	12	13	16	17	19
Utolsó 100ms HDD I/O kihasználtság [%]	55	63	87	61	73

a) A rendelkezésre álló adatok alapján a szerver melyik erőforrása tűnik a szűk keresztmetszetnek?
b) Az első mintavétel idején mekkora az átbocsátási ráta értéke? Az 5 mintavétel alapján mekkora az átbocsátási ráta átlaga?
c) Ezen 5 mérés alapján milyen becslést tudunk adni az egyszerre kiszolgálás alatt lévő kérések átlagos számára?

6. Adatelemzés

6.1. Szerver teljesítménye – felderítő adatelemzése

Egy szerveren az alábbi teljesítményjellemzőket mértük:

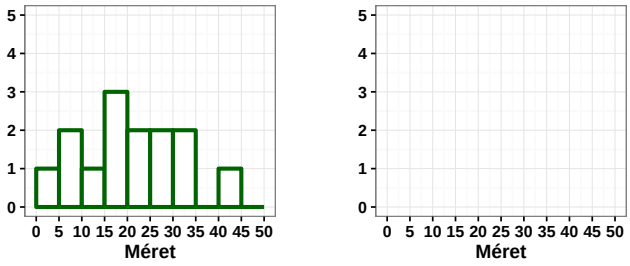
Mintavétel időpontja [ms]	500	600	700	800	900
Utolsó 100ms alatt feldolgozott kérések száma [darab]	11	12	21	18	20
Utolsó 100ms átlagos kiszolgálási ideje [ms]	15	20	21	25	27
Utolsó 100ms CPU kihasználtság [%]	12	13	16	17	19
Utolsó 100ms HDD I/O kihasználtság [%]	55	63	87	61	73

a) Ábrázoljuk a feldolgozott kérések számát és a CPU kihasználtságot pontfelhő (scatterplot) diagramon! Értelmezzük a diagramot!
b) Az első mintavétel idején mekkora az átbocsátási ráta értéke? Az 5 mintavétel alapján mekkora az átbocsátási ráta tapasztalati átlaga és mediánja? Mi tartozik a 40%-os kvantilisbe?
c) Vajon mely mért jellemzők között sejthető ok-okozati viszony?

6.2. Képgaléria – adatelemzés

Online képgalériánkban a felhasználók keresés alapján megjeleníthetnek a keresőkifejezésre illeszkedő képeket.

a) Az alábbi hisztogramon ábrázoltuk az albumok méretének eloszlását. Mivel a tárhely hatékony szervezéséhez elég azt tudnunk, hogy hány 10 alatti, 10 és 20 közötti stb. képet tartalmazó albumunk van, az alábbihoz képest kétszeres oszlopszélességű hisztogramot szeretnénk (szintén a 0 mérettől kezdve felszámítva az oszlopokat). Rajzoljuk meg az ábrát!



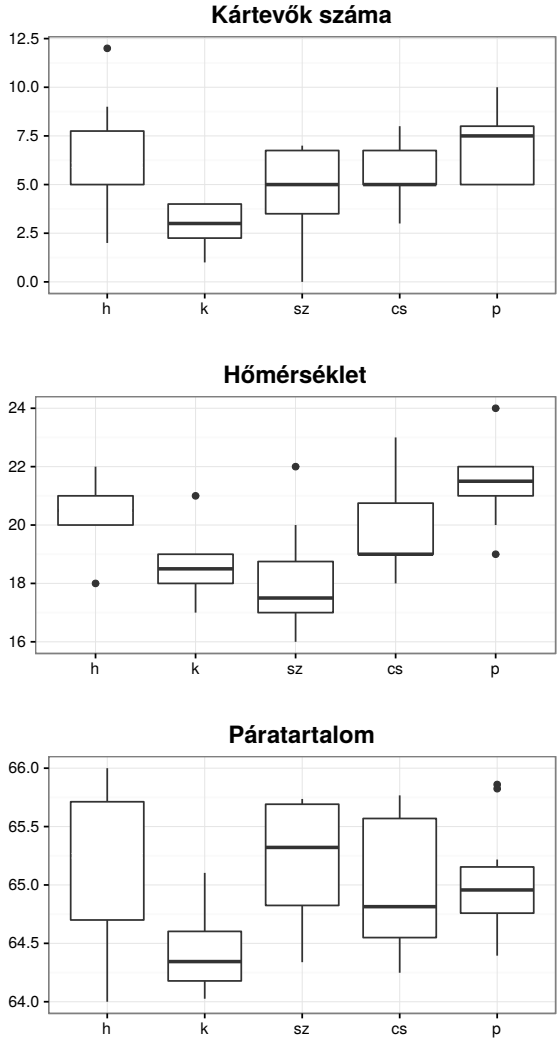
b) Pont-pont diagramon (scatterploton) ábrázoltuk 5 kiválasztott album méretét illetve megtekintési számát a megosztási számmal összehasonlításban. Igaz-e, hogy minél nagyobb az album, annál többen tekintik meg? Válaszolja meg a kérdést egy harmadik pont-pont diagramon, amely a megtekintések számát a méret függvényében ábrázolja!

c) Az albumok jellemző népszerűségét szeretnénk meghatározni, emiatt a pont-pont diagram alapján kiszámoltuk a megtekintési számok átlagát és mediánját. Általánosságban megtehető-e ez egy pont-pont diagram alapján? Mennyivel változnak ezen középértékek, ha feltöltünk egy új albumot, amelyet 40-en tekintenek meg?

6.3. Szenzorhálózat (korábbi zh feladat) – adatelemzés

Adott egy mezőgazdasági szenzorhálózat, amellyel a szabadföldes, üvegházi, ill. fóliasátras területeink állapotát követjük nyomon a mért értékek (hőmérséklet, páratartalom, fényerősség, szélsebesség, detektált kártevők stb.) alapján.

Dátum	Hőm. [°C]	Pára. [%]	Kártevők [db]
2015. 05. 04. 08:00	18	66,00	3
2015. 05. 04. 09:00	20	65,75	6
2015. 05. 04. 10:00	20	65,75	8
2015. 05. 04. 11:00	20	65,50	9
2015. 05. 04. 12:00	20	65,50	5
2015. 05. 04. 13:00	21	65,00	12
2015. 05. 04. 14:00	21	64,70	5
2015. 05. 04. 15:00	21	64,70	6
2015. 05. 04. 16:00	21	64,60	7
2015. 05. 04. 17:00	22	64,00	2



a) Sajnos a május 4. hétfői középértékek (medián) lemaradtak az ábráról, rajzoljuk őket be a táblázatban található adatok alapján!
b) Értelmezze a diagramokat: mely változó(k) első kvartilisei mutat(nak) szigorúan monoton változást az idő folyamán?
c) (Kiegészítő feladat.) Szeretnénk párhuzamos koordináta diagramon összevetni a hétfői hőmérsékleti értékeket a detektált kártevők számával.

6.4. Szenzorhálózat (korábbi zh feladat) – teljesítményelemzés (∗)

(A 6.3. feladathoz kapcsolódó teljesítményelemzési feladatok.) A különböző típusú szenzorok a helyüktől számított 100 méteres körzetben lévő területekről szolgáltatnak adatokat. A szenzorok mérési eredményeiket időbélyeggel ellátva, rádiós kommunikációs hálózaton továbbítják a központnak. A központi számítógép processzora feldolgozza a kéréseket, majd archiválási célzattal kiírja őket egy tárolóegységre. A gazdaságunk összesen 4500 szenzort telepített, amelyek percenként egy-egy mérési eredményről adnak jelentést. A rendszer sikerrel kiszolgálja a terhelést. A rádiós kommunikációs hálózat 100 mérési eredményt képes másodpercenként továbbítani. A központi számítógép CPU idejének 75%-a tétlenül múlik. A tárolóegységet 8 ms-ig foglalja le minden egyes kérés kiírása.

a) Másodpercenként hány mérési adat a rendszer jelenlegi átbocsátása?
b) Mekkora a hálózat, CPU, ill. tároló átbocsátása, átbocsátóképessége és kihasználtsága?
c) A mérési pontosság javításához hány szenzort helyezhetünk még üzembe ugyanezen a területen az infrastruktúra fejlesztése nélkül? Feltételezzünk lineáris skálázódást!

d) A rádióhálózat ügyes kódolással biztosítja, hogy egyszerre több szenzor is sugározhasson mérési eredményeket. Átlagosan hány szenzor rádiója sugároz egyszerre (vagyis hányszoros az átlapolódás) a hálózaton jelenleg, ill. a hálózat maximális terheltsége esetén, ha egy mérési eredmény sugárzása 40 ms-ig tart?

7. Követelménymodellezés

7.1. Vasúti biztosítóberendezés követelményelemzése

Vasúti biztosítóberendezést tervezünk. A rendszer elsődleges célja a vonatok összeütközésének megakadályozása. A megfelelő rendszer kifejlesztésének kulcsa a jó minőségű követelményspecifikáció, ugyanis a követelmények alapján kell majd teszteseteket és egyéb ellenőrző vizsgálatokat kidolgoznunk.

2. táblázat. A vasúti biztosítórendszer követelményei (részlet)

R1	**Biztonság**	A felügyelt pályarendszeren tartózkodó vonatok nem ütközhetnek össze.
R2	**Működés**	A vonatoknak biztosítani kell, hogy elérhessék az úticéljukat.
R3	**Optimalitás**	Minimalizálni kell a vonatok menetidejét.
R4	**Pályaszakaszok felügyelete**	A pályarendszert szakaszokra kell osztani, ezeken egyszerre egy vonat tartózkodhat.
R5	**Szakaszokra bontás**	A pályarendszert szakaszokra kell bontani.
R6	**Foglaltság**	Egy szakaszon egyszerre egy vonat tartózkodhat.
R7	**Foglaltság érzékelése**	Valamilyen módon érzékelni kell, hogy egy szakaszon áll-e vonat, vagy nem.
R8	**Hibatűrés**	A komponensek meghibásodására fel kell készülni.
R9	**Foglaltságjelző szenzorok**	A foglaltságot többféle, redundánsan kialakított szenzorral kell érzékelni.
R10	**Sínekbe épített szenzor**	A sínekbe mindegyik szakaszon szenzorokat kell telepíteni, amik jelzik, hogy a szakaszon áll-e vonat, vagy sem.
R11	**Kamerás rendszer**	Ahol lehetséges, kamerákat kell telepíteni a szakaszok megfigyelésére.
R12	**Helyzetmeghatározás**	A vonatoknak folyamatosan jelezni kell a helyzetüket a központi vezérlő felé.
R13	**GPS alrendszer**	A vonatokat GPS alrendszerrel kell felszerelni.
R14	**Vezeték nélküli kapcsolat**	Biztosítani kell, hogy a vonatok vezeték nélküli hálózaton jelezhessék a helyzetüket a központi vezérlő felé.
R15	**Vonatok vezérlése**	Meg kell tudni akadályozni, hogy a foglalt szakaszra másik vonat is ráhajthasson.
R16	**Vonat leállítása**	A központi rendszer azonnal leállíthatja a vonatot.
R17	**Mozdonytípusok támogatása**	A rendszernek támogatnia kell minden, a sínpáron közlekedni képes mozdonytípust.
R18	**Mozdony nem módosítható**	Nem alkalmazható olyan megoldás, amihez a mozdonyok vezérlését meg kellene változtatni.

a) Gyűjtsük össze azokat a szereplőket, akik egy ilyen rendszer kifejlesztése kapcsán érintettek, vagyis követelményeket támaszthatnak a leendő rendszerrel szemben (ún. *stakeholderek*)!

b) A stakeholderek felderítése után összegyűjtöttük az általuk támasztott követelményeket is, ennek egy részletét tartalmazza a 2. táblázat. Rajzoljuk fel a követelmények közötti függőségi viszonyokat egy gráf segítségével! A gráfban A-ból B-be mutató irányított éllel jelezzük, ha *(1)* az A követelmény a B követelmény része (*kompozíció*), *(2)* az A követelmény finomítja (részletezi) a B követelményt (*refines* kapcsolat), illetve *(3)* az A követelmény származtatható a B követelményből (*derive* kapcsolat). Ne foglalkozzunk azzal, hogy két követelmény közül ezen viszonyok melyike áll fent; most csak a kapcsolat megléte a fontos.

c) A felsoroltak közül melyek funcionális követelmények, illetve milyen típusúak az extrafunkcionális követelmények (biztonságosság, teljesítmény, megbízhatóság stb.)?
d) Vizsgáljuk meg, hogy konzisztens-e a bemutatott követelményrendszer! Ha nem az, akkor mutassunk példát ellentmondásra!
e) A fentiekből adjunk példát közvetlenül ellenőrizhető követelményre!

7.2. Rendszermodellezés tárgy követelményei

Az alábbiakban felsoroljuk a Rendszermodellezés tárgy menetével kapcsolatos követelményeket. Figyelem: a felsorolás (a legtöbb valós specifikációhoz hasonlóan) nem feltétlenül logikus sorrendben szedi össze a követelményeket és nem biztos, hogy teljes/konzisztens. A könnyebb hivatkozás kedvéért a követelményeket sorszámmal láttuk el.

R1 A tárgy elvégzésének feltétele az aláírás megszerzése.
R2 A félév 1 regisztrációs hétből, 14 oktatási hétből és 1 pótlási hétből áll.
R3 Az aláíráshoz a hallgatónak az összes zárthelyi dolgozatot és a házi feladatot elégséges szintre teljesítenie kell.
R4 A tárgyból egy házi feladat van.
R5 A házi feladat leadási határideje a 12. hét.
R6 A házi feladat mellé plusz pontszám szerezhető szorgalmi feladat és a bemelegítő feladat leadásával.
R7 A házi feladat a pótlási héten pótolható.
R8 A zárthelyi dolgozatok közül csak egy pótolható.
R9 A tárgyból két zárthelyi dolgozat van.
R10 A tárgy csak akkor vehető fel, ha a hallgató teljesítette a tanrend szerinti előkövetelményeket.
R11 A tárgy sikeres elvégzése után az összes ráépülő tárgy felvehető.
R12 A házi feladat kiadásának ideje a 3. hét.
R13 Az 1. zárthelyi ideje a 8. hét.
R14 A 2. zárthelyi ideje a 14. hét.
R15 A házi feladat a pótlási héten pótolható.
R16 Minden pótlás különeljárásidíj-köteles.
R17 A tárgy 14 előadásból és 6 gyakorlatból áll.
R18 A gyakorlatokon történő részvétellel jutalompont szerezhető.
R19 A tárgyra kapott jegy a zárthelyik pontszáma, a házi feladat pontszáma és a jutalompontok összegének függvénye.
R20 A gyakorlatok opcionális beugrófeladattal indulnak, melyekért jutalompont jár.
R21 A zárthelyi kötelezően beugróval indul, melynek nem teljesítése a zárthelyi nem teljesítését vonja maga után.

a) Teljes-e a fenti követelményrendszer? (Ha nem, hogyan változtatna rajta?)
b) Konzisztens-e a fenti követelményrendszer? (Ha nem, hogyan változtatna rajta?)
c) Rajzoljon folyamatmodellt, ami az egyéni hallgató szemszögéből mutatja be a tárgy menetét!
d) Mennyiben térhet el egy folyamatmodell, amely az oktató szemszögéből mutatja be a tárgy menetét?
e) Ha feltételezzük, hogy a tárgy összes, követelmények teljesítésével kapcsolatos lépése (házi feladat kiadás, házi feladat beadás, értékelés stb.) munkafolyamat alapon történik, akkor hány különböző munkafolyamat sablon hány különböző példánya fut jelenleg a rendszerben?
f) Ha a Neptun szemszögéből nézzük, mik a tárgy elvégzésének lehetséges kimenetei? (Egy hallgatói munkafolyamat futásának eredményei?) Ha az előtanulmányi követelmények kiértékelésének szempontjából nézzük, mik a lehetséges kimenetek? Milyen viszony áll fenn ezek közt? Mindez mennyiben változna, ha a tárgy vizsgás tárgy lenne?
g) Hogyan ellenőrizhető, hogy a tárgy végrehajtásának folyamata a Tanulmányi és Vizsgaszabályzatnak (TVSZ) megfelel-e?

8. Kísérlettervezés*

Adatelemzés – elméleti ismeretek

Valószínűségszámítási alapfogalmak

- Valószínűségi változó (*random variable*): X
- Várható érték, átlag (*expected value*, *average*, *mean*):

$$\mu = \mathbb{E}X = \sum_{i=1}^{n} p_i x_i$$

- Szórásnégyzet (*variance*):

$$\sigma^2 = \mathbb{E}\left(X - \mu\right)^2 = \sum_{i=1}^{n} p_i (x_i - \mu)^2$$

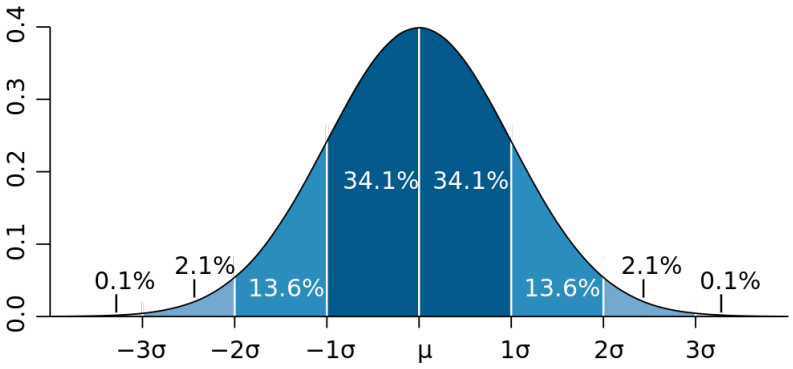
- Szórás (*standard deviation*):

$$\sigma = \sqrt{\mathbb{E}\left(X - \mu\right)^2} = \sqrt{\sum_{i=1}^{n} p_i (x_i - \mu)^2}$$

Konfidencia – normális eloszlás

A normális eloszlású változó

- az esetek 68%-ában legfeljebb 1σ messze kerül μ-től,
- az esetek 95%-ában legfeljebb 2σ messze kerül μ-től,
- az esetek 99,7%-ában legfeljebb 3σ messze kerül μ-től.



1. ábra. Konfidenciaintervallumok

Statisztikai alapfogalmak

- Megfigyelések: t darab, $x_1, \ldots, x_t$
- Tapasztalati átlag (*sample mean*):

$$m = \bar{x} = \frac{x_1 + \cdots + x_t}{t}$$

- Korrigált tapasztalati szórás (*unbiased sample standard deviation*):

$$s = \sqrt{\frac{(x_1 - m)^2 + \cdots + (x_t - m)^2}{t - 1}} = \sqrt{\frac{\sum_{i=1}^{t} (x_i - m)^2}{t - 1}}$$

  Figyeljük meg, hogy a korrigált tapasztalati értékeknél t helyett $(t-1)$-gyel osztunk. Ennek oka, hogy t-vel osztva a kapott érték általában alábecsli a teljes populáció szórását. Belátható, hogy $(t-1)$-gyel osztva a valódi szórást jobban közelítő értéket kapunk. Ezt nevezzük Bessel-féle korrekciónak (`https://en.wikipedia.org/wiki/Bessel's_correction`).

Kísérlettervezés

A centrális határeloszlás-tételből (*central limit theorem*) következik, hogy tetszőleges eloszlású jellemző (véges μ várható értékkel és σ szórással) tapasztalati átlaga (m valószínűségi változó) $t \to \infty$ esetén normális eloszlású, μ várható értékkel és $\sigma_m = \frac{\sigma}{\sqrt{t}}$ szórással (ahol σ a megfigyelés alatt álló valószínűségi változó szórása).
Ökölszabály: ismert szórásnál $t > 30$ után kezd elfogadható lenni ez a közelítés.
A fentiek miatt a megfigyeléseink kiértékeléséhez használhatjuk a normális eloszláshoz tartozó konfidenciaintervallumokat. Eszerint m 68%-os valószínűséggel maximum σ_m távolságra esik μ-től. Tehát (ha m-et és σ_m-et ismerjük, és μ-t szeretnénk ez alapján becsülni) μ értékéről kijelenthetjük, hogy

- 68% valószínűséggel $m - \sigma_m$ és $m + \sigma_m$ közé esik, valamint ehhez hasonlóan
- 95% valószínűséggel $m - 2\sigma_m$ és $m + 2\sigma_m$ közé esik,
- és 99,7% valószínűséggel $m - 3\sigma_m$ és $m + 3\sigma_m$ közé esik.

Sokszor azonban σ nem ismert (a mérésekre azért van szükség, hogy becsléseket kaphassunk μ és σ értékére), ekkor elfogadható a $\sigma_m \approx \frac{s}{\sqrt{t}}$ (ahol s a korrigált tapasztalati szórás), amennyiben $t \geq 100$.

8.1. Kísérlettervezés

Egy modellezett folyamat átbocsátóképességére szimuláció alapján szeretnénk egy közelítő értéket és hozzá tartozó konfidencia-intervallumot meghatározni.

a) Hány szimuláció mérési eredményeiből számoljunk átlagot?
b) Az így elvégzett mérési eredmények tapasztalati közepe 500 kérés/s; a tapasztalati szórás 10%. Szeretnénk, hogy 95% konfidencia mellett egy legfeljebb 40 kérés/s széles intervallumba essen az átbocsátóképesség. Hány mérést végezzünk még?

8.2. Kísérlet kiértékelése

Infrastruktúránk méretezését megnehezíti, hogy egy adott feladattípus végrehajtási ideje a körülmények függvényében ingadozik, például lapozás, memória szemétgyűjtés, memória cache találatok stb. változékonysága folytán. Ezért összeállítottunk egy valós munkaterhelést jól jellemző benchmarkot, és ennek többszöri lefuttatása során a futási időket átlagolva szeretnénk meghatározni a rendszer átlagos teljesítményét.

a) Az első tíz futtatás eredményei: 37 s, 34 s, 35 s, 39 s, 57 s, 41 s, 36 s, 35 s, 61 s, 35 s. Mennyi ez alapján a rövid kísérlet alapján a tapasztalati átlag és tapasztalati szórás?
b) Nagyobb léptékben futtatva a kísérletet, a benchmark 10 000 futtatása átlagban 44,3 másodpercig tartott, 11,6 másodperc tapasztalati szórással. Mennyire lehetünk biztosak a kapott eredmény pontosságában?