OEP

13. táblás gyakorlat

11. előadás

Objektumok viselkedésének leírása
Segíti / támogatja a következő folyamatokat:

- tervezés, megvalósítás
- dokumentálás
- tesztelés

Viselkedési nézetek

- Az UML az objektumok dinamikus viselkedésének jellemzésére számos nézetet vezetett be. Ezek közül az alábbiakkal ismerkedünk meg:
  - Használati eset (*use case*) diagram
  - Kommunikációs (*communication*) diagram
  - Szekvencia (*sequence*) diagram
  - Állapotgép (*state machine*) diagram

Objektum viselkedésének tanulmányozására, leírására szolgál az **állapotgép diagram**

11. előadás

Objektum életciklusa

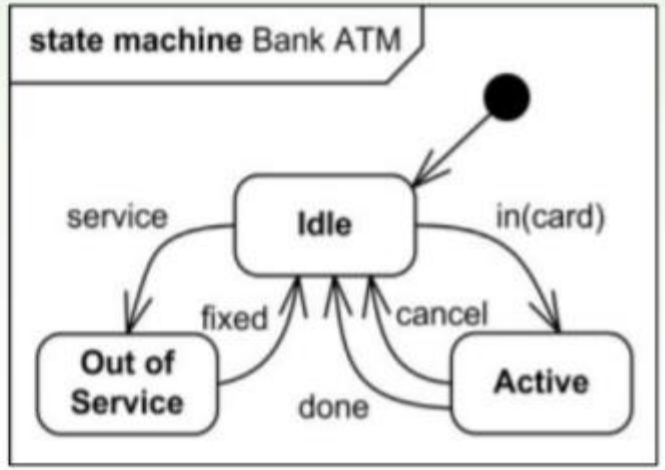
- Az objektum életciklusa során:
  - létrejön: az objektum speciális műveletével, a konstruktorral,
  - működik: más objektumokkal kommunikál, azaz szinkron vagy aszinkron módon üzeneteket küldenek egymásnak, és ennek során megváltozhatnak az adatai,
  - megsemmisül (egy másik speciális művelettel, a destruktorral).
- Egy objektumnak különféle állapotai (*state*) vannak: egy állapot (fizikai állapot) az objektum adatai által felvett értékek együttese, amely az életciklus során változik.
- A könnyebb áttekinthetőség kedvéért azonban gyakran egy állapotnak az objektum több különböző, de közös tulajdonságú fizikai állapotainak összességét tekintjük (logikai állapot).

11. előadás

Állapotgép

- Az állapotgép diagram egy objektum életciklusát ábrázolja. Megmutatja, hogyan változnak egy objektum állapotai az objektumnak küldött üzenetek (metódus hívások vagy szignálok) hatására.
- Az állapotgép egy működést ír le, amelyben mindig egy állapot lehet csak aktív.

- Az állapotgép egy irányított gráf, amelynek csomópontjai az objektum logikai állapotait, irányított élei pedig az állapotok közötti átmeneteket mutatják.
- Mind az állapotokhoz, mind az átmenetekhez tartozhatnak végrehajtandó tevékenységek.



11. előadás

Állapotok

- Az állapotokat lekerekített sarkú téglalap jelöli:

<állapot neve>

anonim is lehet

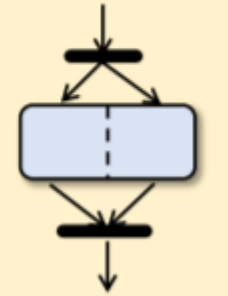
Pszeudo állapotok:

- kezdő állapot
- végállapot

Hierarchikus állapotgépekben:

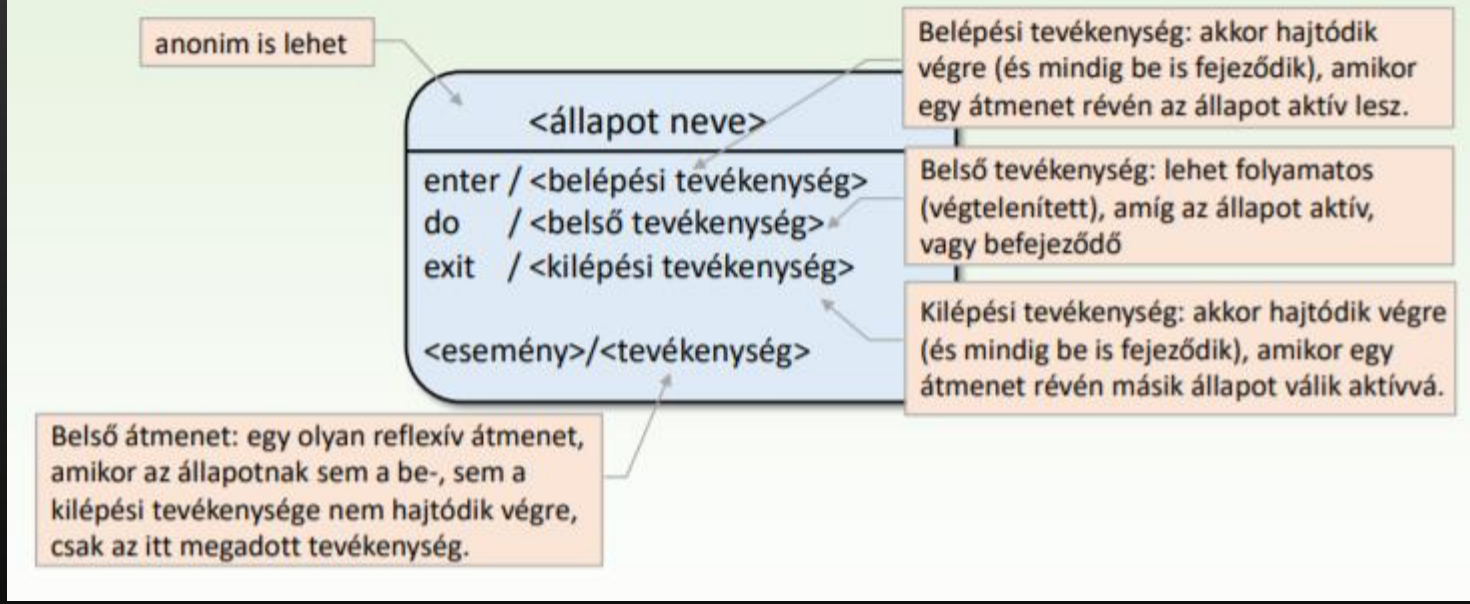
- belépés (entry)
- kilépés (exit)
- megszüntetés
- shallow history (H)
- deep history (H*)

- elágazás (choice) [feltétel] [feltétel]
- csomópont (junction)
- szétágazás (fork)
- összefutás (join)



11. előadás

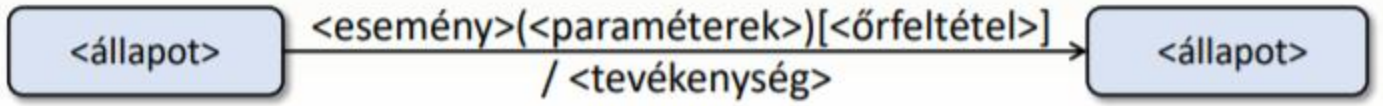
Állapot belső tevékenységei



11. előadás

Állapot-átmenetek jelölése

- Az állapot-átmenetet az alábbi elemek (bármelyik hiányozhat) jellemzik:
  - átmenetet előidéző esemény (*event, trigger*) a paramétereivel
    - lehet az adott objektum egy metódusának a hívása
    - vagy az objektumnak küldött szignál
  - átmenetet megengedő őrfeltétel (*guard*), amely
    - vagy az esemény paramétereitől függő logikai állítás (*when*)
    - vagy egy időhöz kötött várakozási feltétel (*after*)
  - átmenethez rendelt tevékenység (az objektum adattagjaival és a kiváltó esemény paramétereivel operáló program)
- Egy átmenet lehet reflexív.

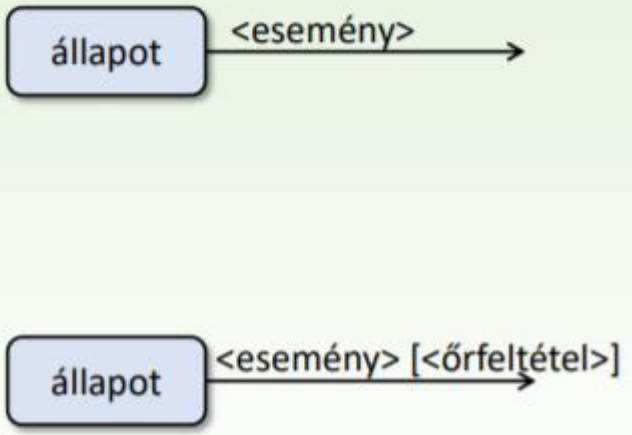


11. előadás

Állapot-átmenetek szemantikája I.

Amikor az átmenet egy esemény hatására jön létre:

- Őrfeltétel hiányában, amikor az esemény bekövetkezik, akkor az állapot belső tevékenysége megszakad, és az átmenet megvalósul.
  Ugyanazon állapotból nem vezethet ki két él ugyanazon eseménnyel.
- Őrfeltétel mellett az átmenet (a belső tevékenység megszakításával) csak akkor valósul meg, ha az esemény bekövetkezésekor az őrfeltétel igaz. Egyébként az esemény hatástalan. (nincs várakozás)
  Egy esemény ugyanazon állapotból kivezető több élhez is tartozhat, feltéve, hogy diszjunkt őrfeltételekkel van rendelkezik.

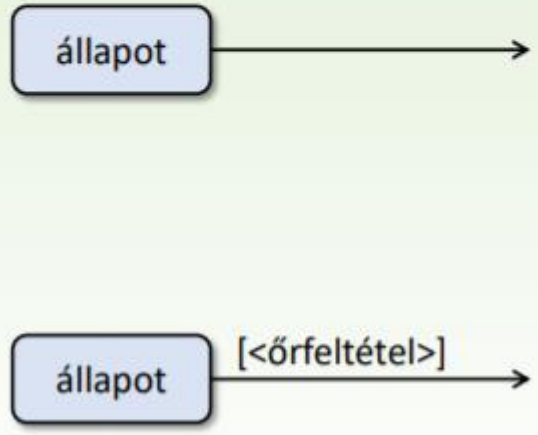


11. előadás

Állapot-átmenetek szemantikája II.

Amikor az átmenetet nem esemény váltja ki:

- Őrfeltétel hiányában az átmenet az állapot belső tevékenységének befejeződésekor valósul meg.
  Ha nincs belső tevékenység, akkor azonnal.
- Őrfeltétel esetén az átmenet bekövetkezése (az esetleges belső tevékenység befejeződése után) addig várakozik, amíg az őrfeltétel igaz nem lesz, feltéve, hogy várakozás közben más állapot-átmenetre nem kerül sor.

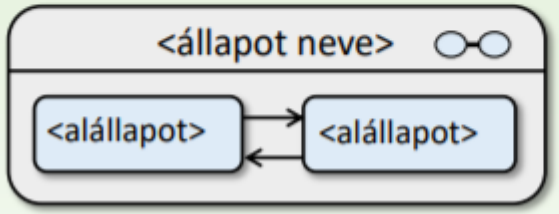


11. előadás

Hierarchikus állapotok

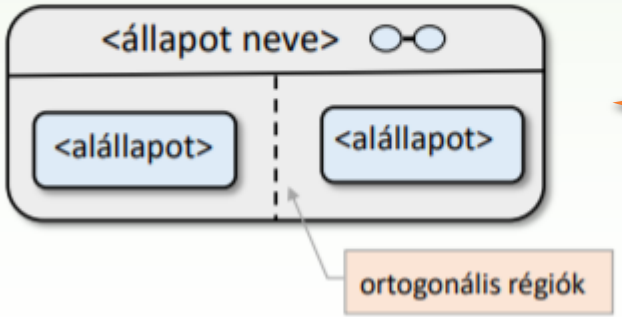
Összetettebb feladatok esetén túl bonyolult ábra keletkezne.

Szekvenciális:



Állapotcsoportot foglal magába, egy „kisebb” állapotgép diagramot tartalmaz.

Párhuzamos:



Több, egymással párhuzamosan működő állapotgéppel írható le. Például ha kompozícióval több objektumot is magába foglal.

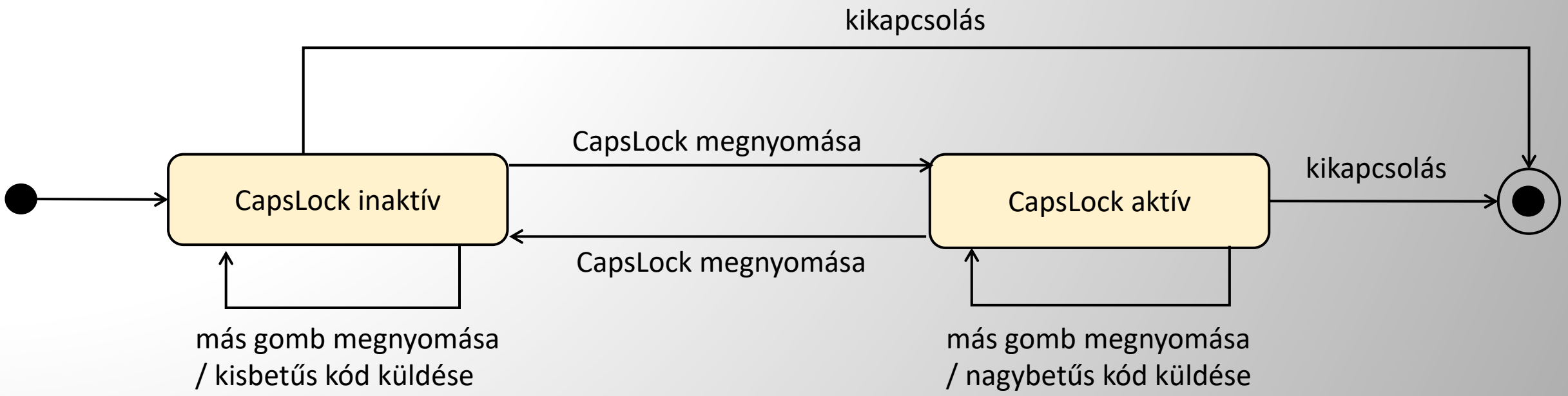
Példák

- Billentyűzet
- Közlekedési lámpa
- Videómagnó
- Verem
- Bankautomata
- Áru

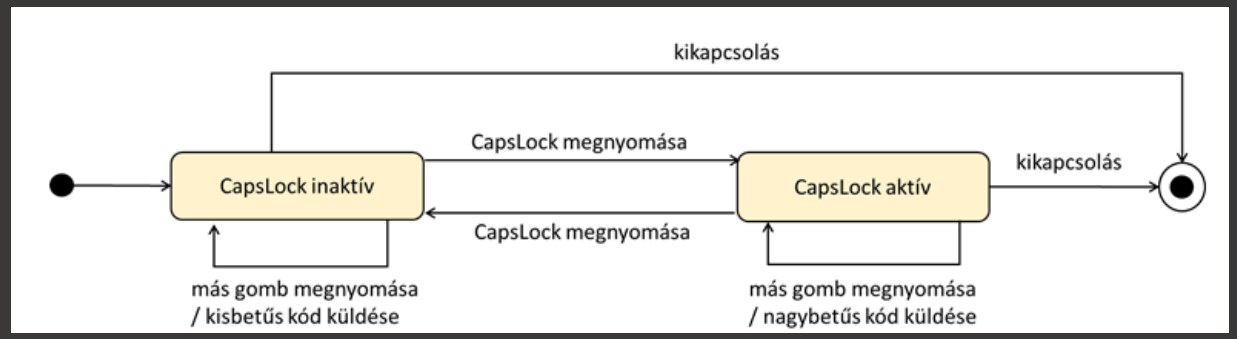
1. Egyszerűsített billentyűzet modellezése:
   ha a CapsLock aktivált, akkor minden más billentyű lenyomásra nagybetűs karaktereket,
   különben kisbetűs karaktereket kapunk.

Legyen két állapot: CapsLock aktív, illetve inaktív.

Legyen háromféle művelet: CapsLock lenyomása, más gomb lenyomása, kikapcsolás



Melyik állapot átmenet táblázat helyes?



A

	inaktív	**aktív**
CapsLock	aktív	inaktív
más nyomógomb	aktív **/** nagybetűs kód	inaktív **/** kisbetűs kód
kikapcsolás	exit	exit

B

	inaktív	**aktív**
CapsLock	aktív	inaktív
más nyomógomb	inaktív **/** kisbetűs kód	aktív **/** nagybetűs kód

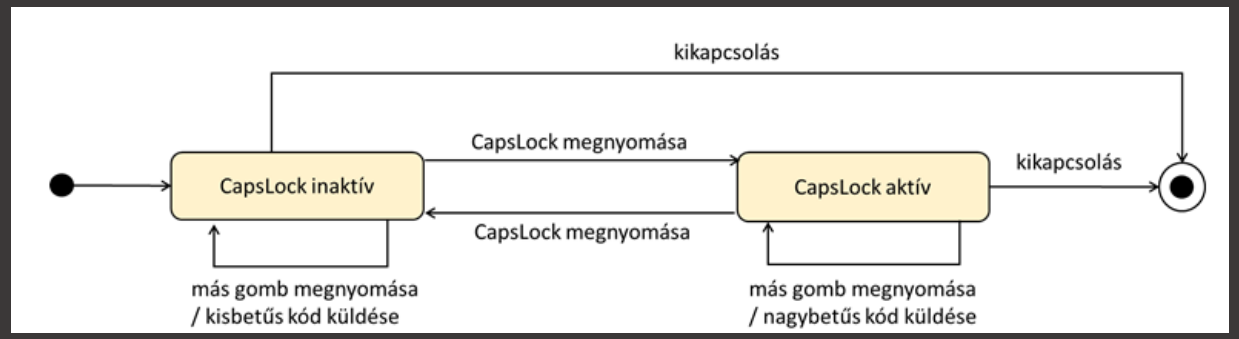
C

	inaktív	**aktív**
CapsLock	aktív	inaktív
más nyomógomb	inaktív	aktív
kikapcsolás	exit	exit

D

	inaktív	**aktív**
CapsLock	aktív	inaktív
más nyomógomb	inaktív **/** kisbetűs kód	aktív **/** nagybetűs kód
kikapcsolás	exit	exit

Hogyan valósítható meg a mellékelt állapotgép diagram?



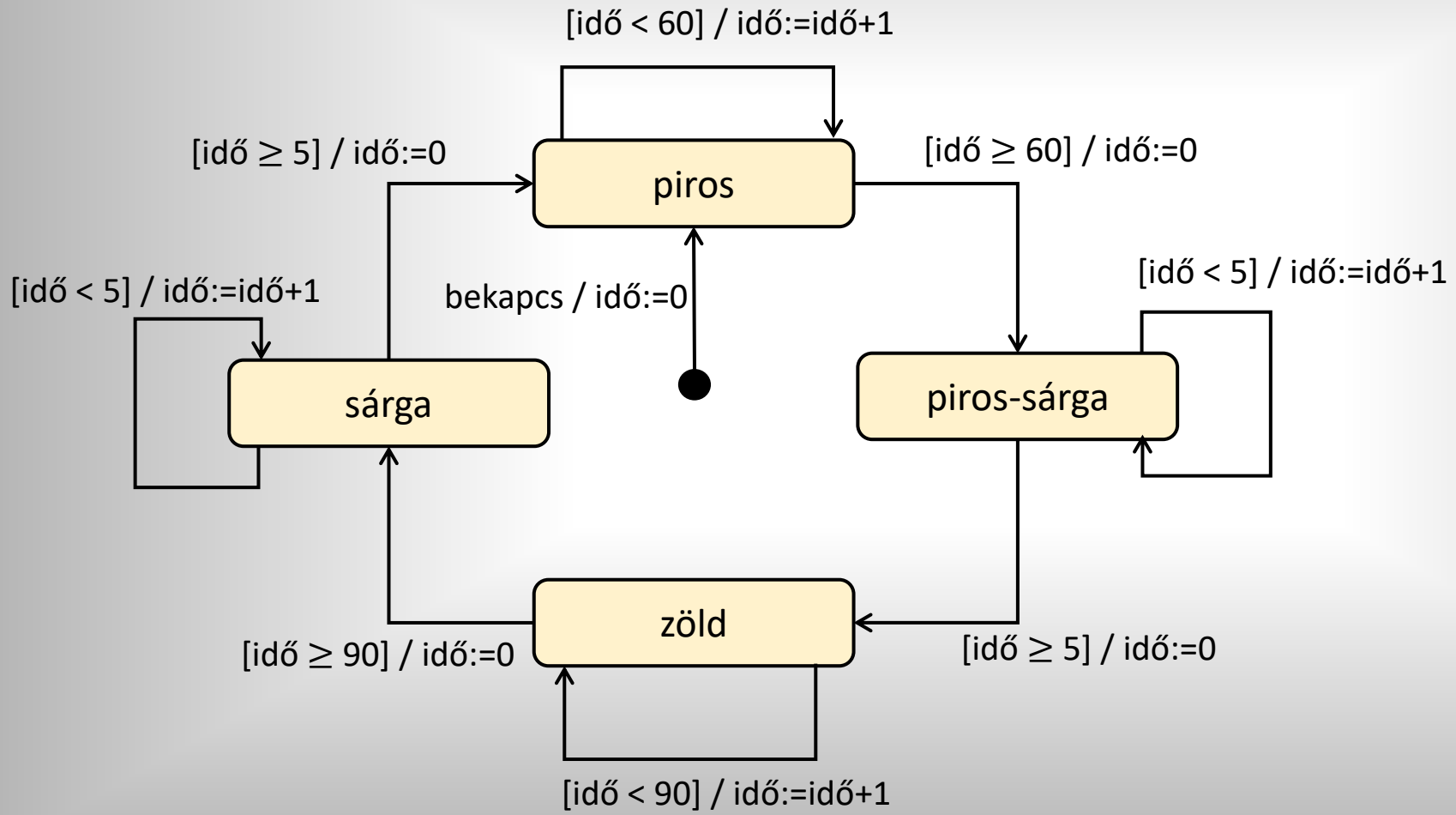
A

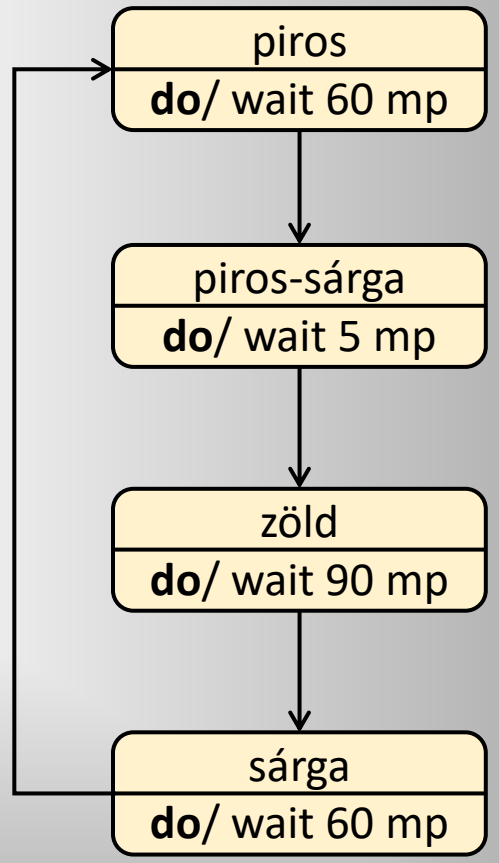
```
state := inaktív
while  signal ≠ kikapcsolás loop
     if signal = CapsLock then
          switch state do
               case inaktív: state := aktív
               case aktív:   state := inaktív
          endswitch
     else
          switch state do
               case inaktív: signal kisbetűs kódjának küldése
               case aktív:   signal nagybetűs kódjának küldése
          endswitch
     endif
endloop
```

B

```
state := inaktív
while state ≠ kikapcsolás loop
     if signal = CapsLock then
          if     state = aktív   then state := inaktív
          elseif state = inaktív then state := aktív
          endif
     else
          if     state = aktív   then signal nagybetűs kódjának küldése
          elseif state = inaktív then signal kisbetűs kódjának küldése
          endif
     endif
endloop
```

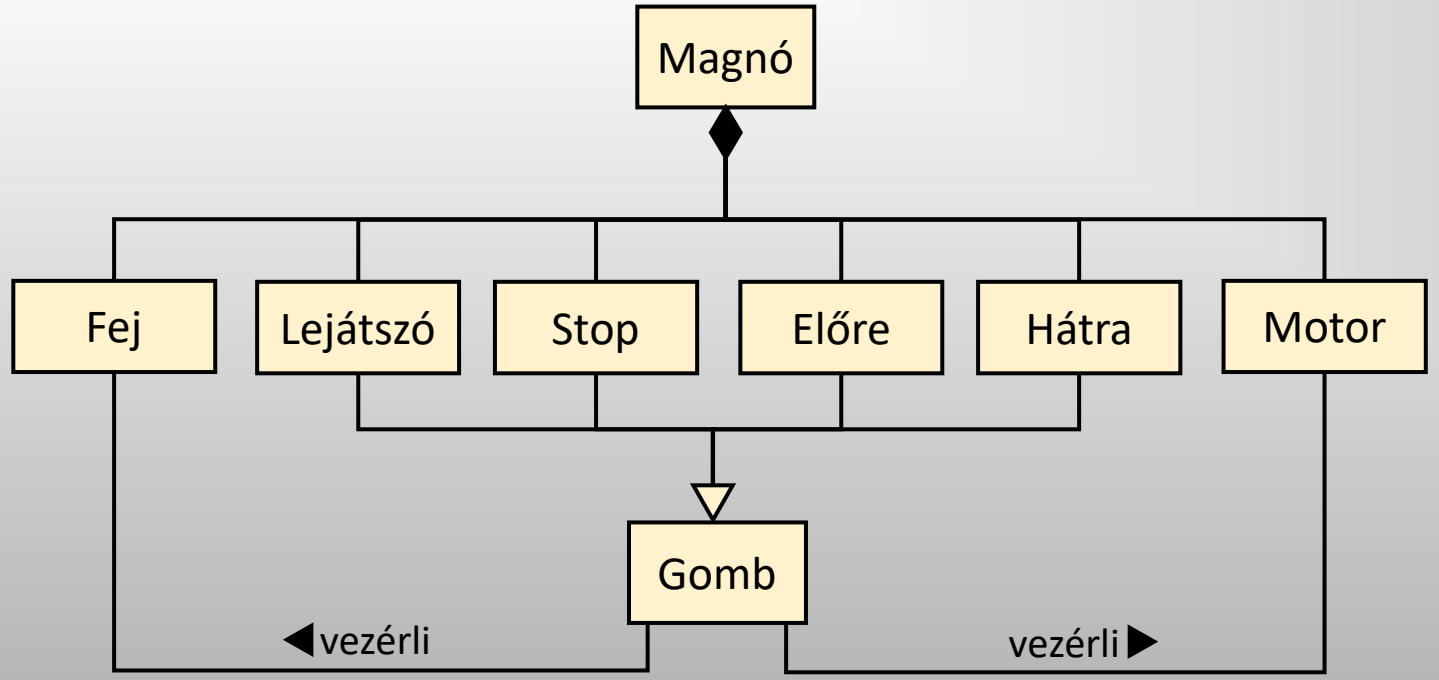
2. Egy közlekedési lámpán piros, piros-sárga, zöld, sárga fények vannak. A lámpa 60 másodpercig piros és 90 másodpercig zöld színű. Az átmeneti állapotok 5 másodpercig tartanak: pirosról a zöldre a piros-sárgán keresztül, zöldről a pirosra a sárgán keresztül. Kezdetben a lámpa piros.

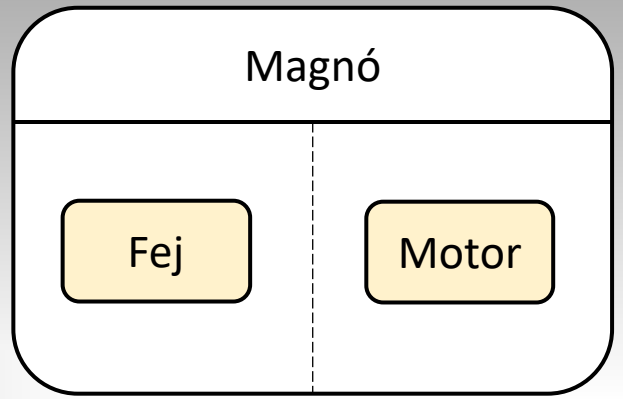


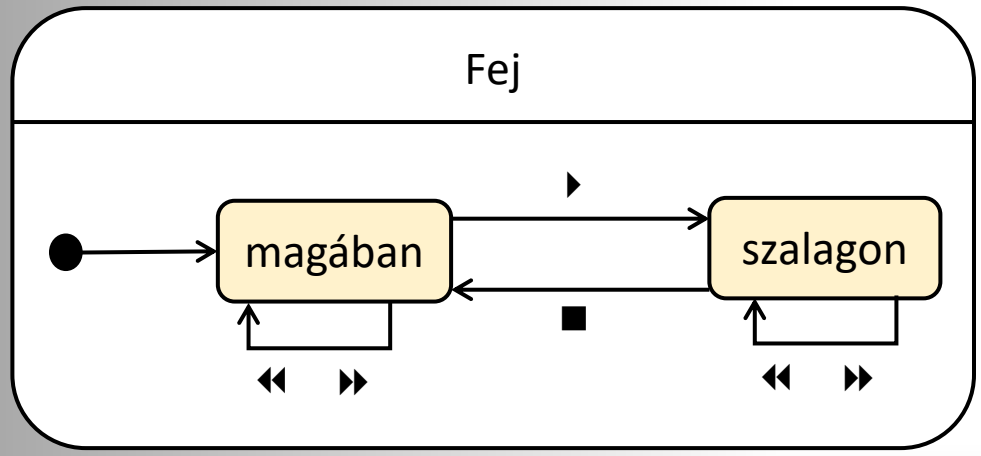


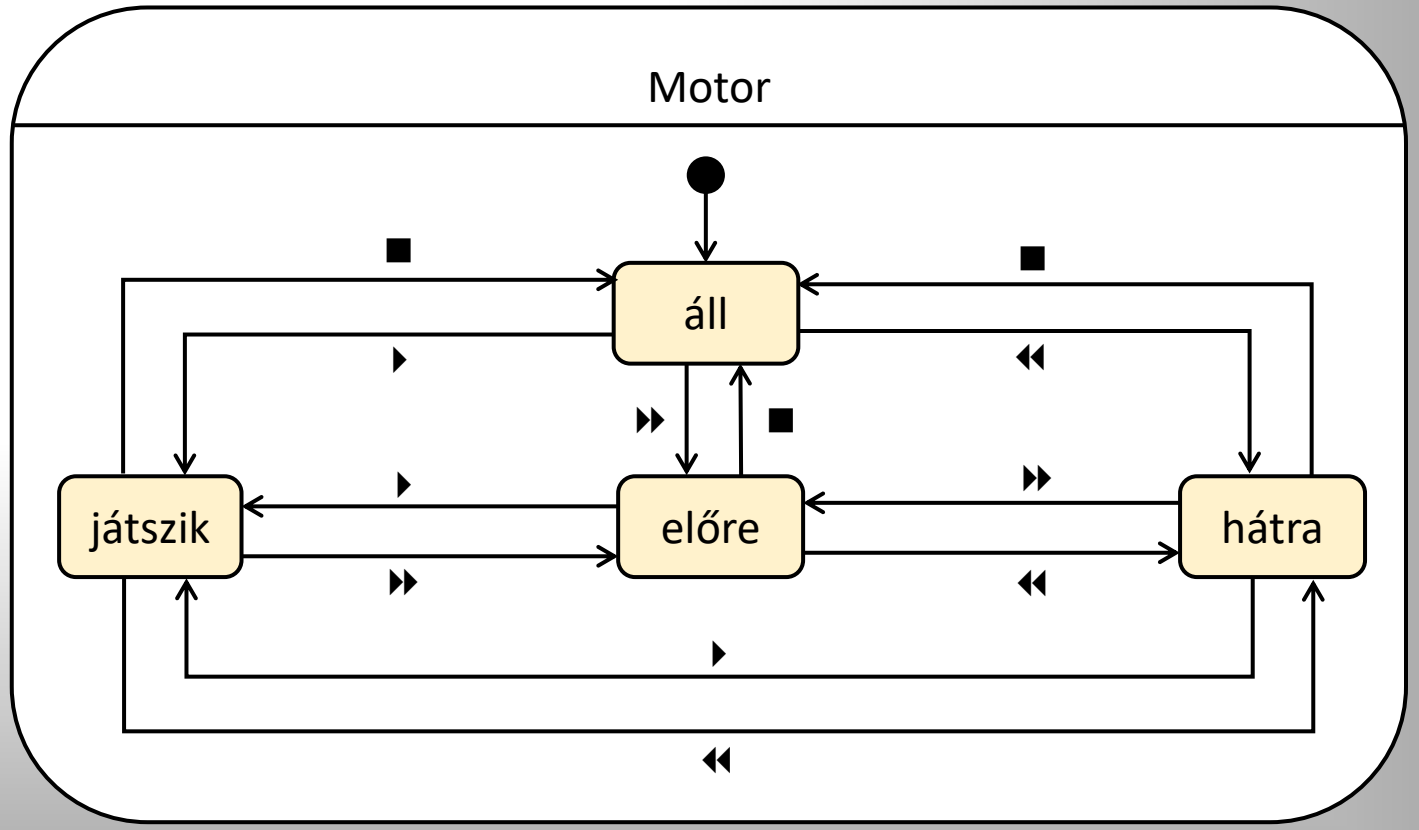
3. Készítsük el egy videomagnó osztálydiagramját és állapotgépét! A magnóban található egy olvasó fej és egy motor, amelyeket négy gomb segítségével vezérelhetünk. A gombokat elegendő megérinteni a vezérlés során.
A négy gomb és vezérlési szerepük a következő:
- ■ (állj) : leállítja a motort, és a fejet felemeli a szalagról, ha azon volt.;
- ▸ (lejátszás) : lejátszó sebességbe helyezi a motort, és a fejet a szalagra helyezi.
- ▸▸ (előre) : a motor előre csévéli a szalagot
- ◂◂ (hátra) : a motor hátra csévéli a szalagot

Előre, illetve hátracsévélés alatt a fej a szalagon lehet: ez a gyorskeresés funkció

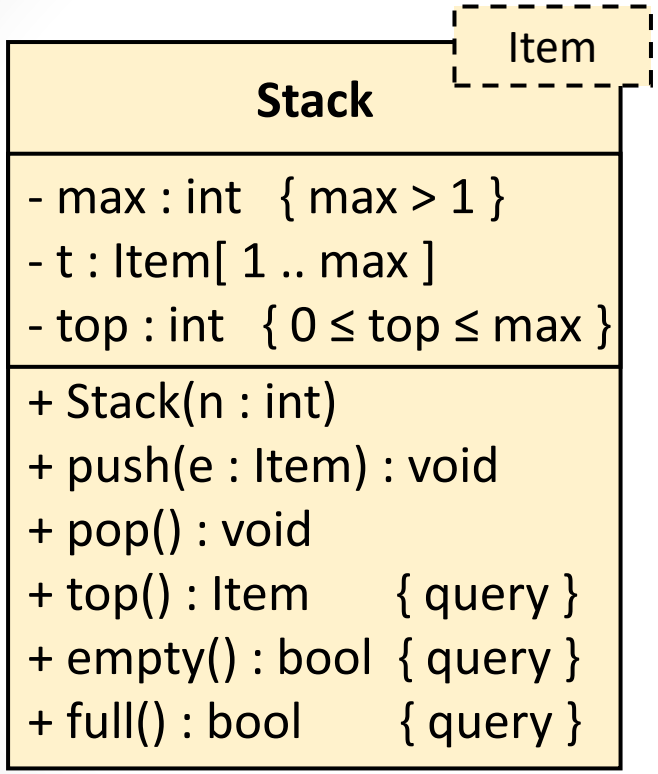



Magnó
Fej
Motor


Fej
magában
szalagon

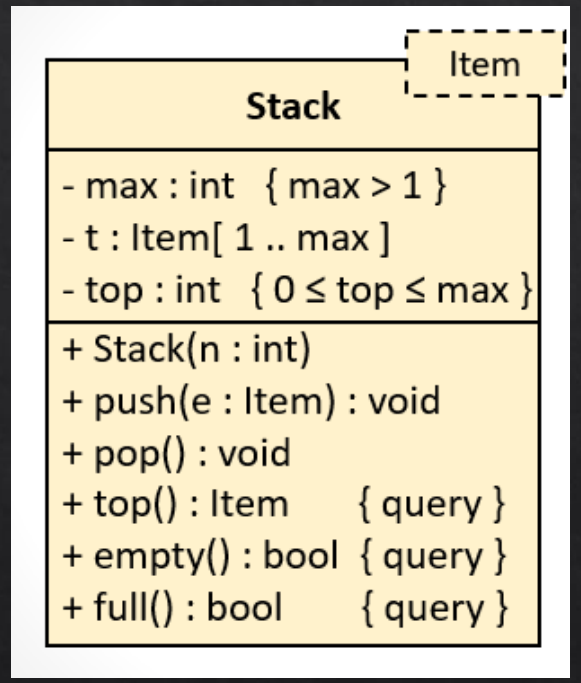

Motor
áll
játszik
előre
hátra

4. Vegyünk egy korlátos vermet (push(), pop() műveletekkel), amelyet egy (t:array[1..max] of Item) tömb, és egy (top:int) index reprezentál.

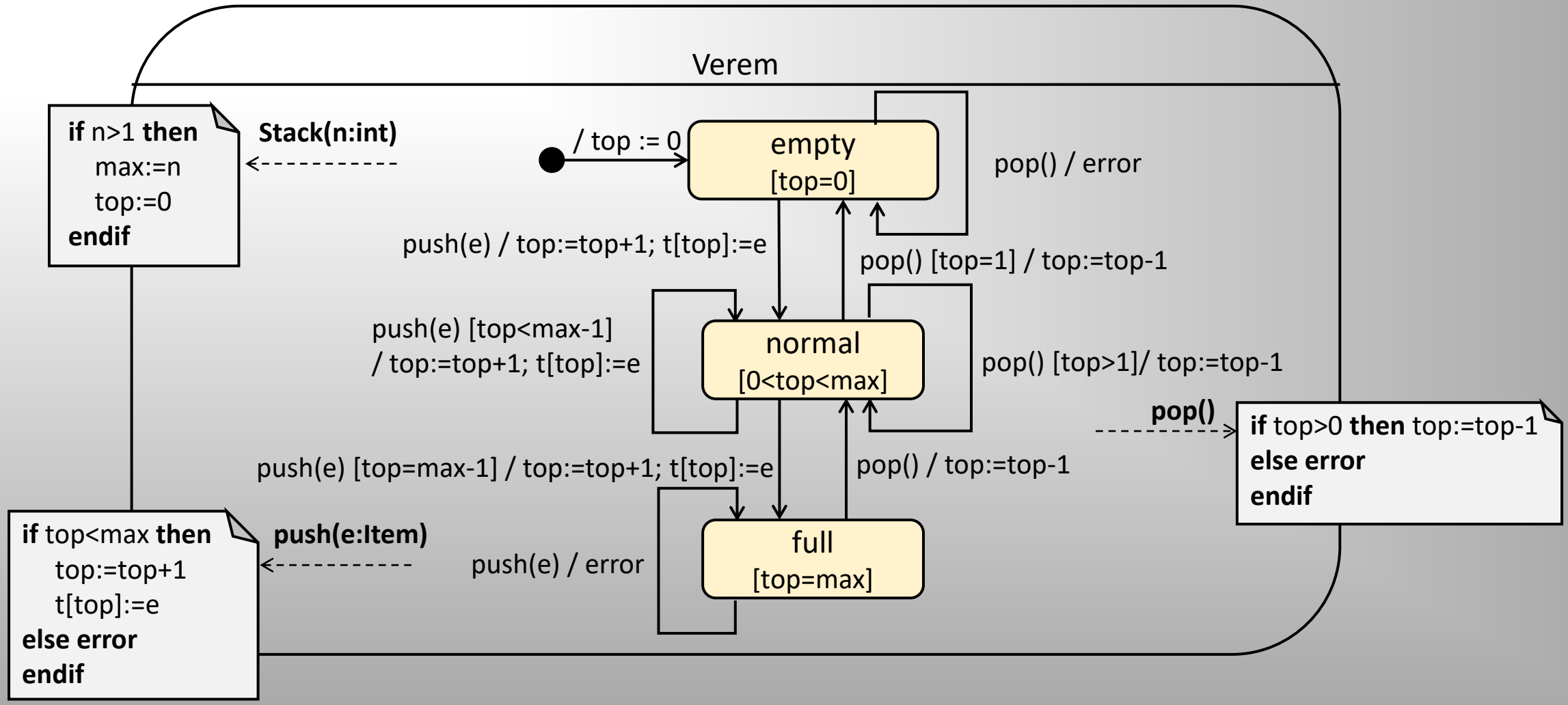


Melyek a fizikai-, és melyek a logikai állapotai a korlátos veremnek?

1. A verem fizikai állapota, hogy üres, egy elemű, ..., tele; és logikai állapotai, hogy üres, teli, és köztes feltöltöttségű.
2. A fizikai állapot annyiféle van, ahányféleképpen felvehetnek értéket a verem max, t, és top adattagjai. A logikai állapotok attól függnek, hogy a műveletek működése milyen esetekre bontható fel. Ennek megfelelően beszélhetünk üres, tele, és köztes feltöltöttségű vermekről.
3. Külön fizikai állapotnak számít ugyanazon verem reprezentációja, ha a t tömbben a top-adik elem (ez a verem tetején levő elem) előtt eltérőek az elemek, vagy különbözik a max-ban tárolt tömb méret, de ezek a fizikai állapotok ugyanazon logikai állapotnak felelnek meg.
4. Ebben az esetben ez a két fogalom egybeesik.



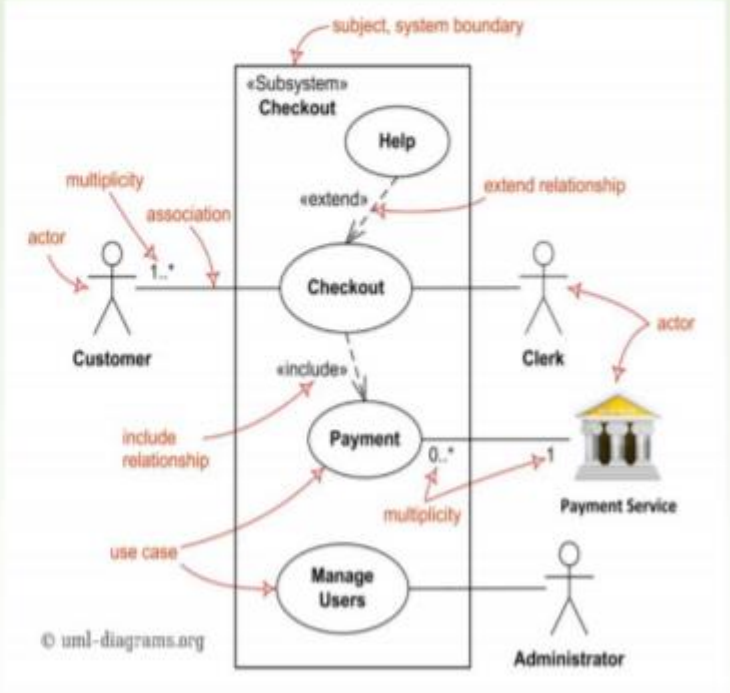
4. Vegyünk egy korlátos vermet (push(), pop() műveletekkel), amelyet egy (t:array[1..max] of Item) tömb, és egy (top:int) index reprezentál.
Három állapotot vezetünk be: „empty” (top=0), „normal” (0<top<max), „full” (top=max), amelyek között a verem műveletek (push(), pop()) hatására következik be átmenet. Ezzel azt is kikötöttük, hogy max>1.
Kezdetben (kezdeti átmenet) top:=0.



11. előadás

Használati eset diagram

- Megmutatja, hogy a tervezett rendszer
  - milyen funkciókat lát el, milyen szolgáltatásokat biztosít,
  - kik számára nyújtja a szolgáltatásokat
  - milyen követelményeket vár el a környezetétől

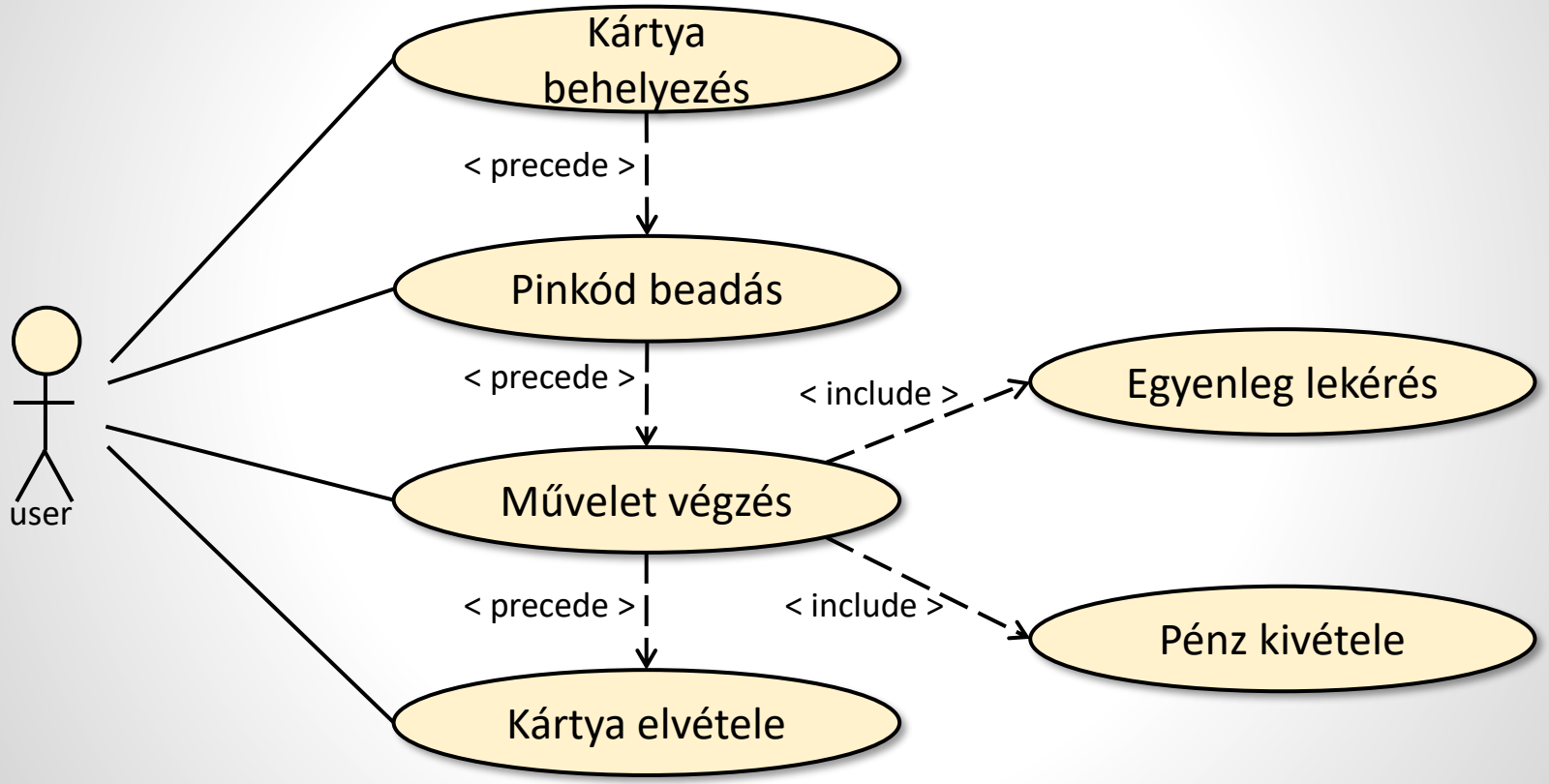


11. előadás

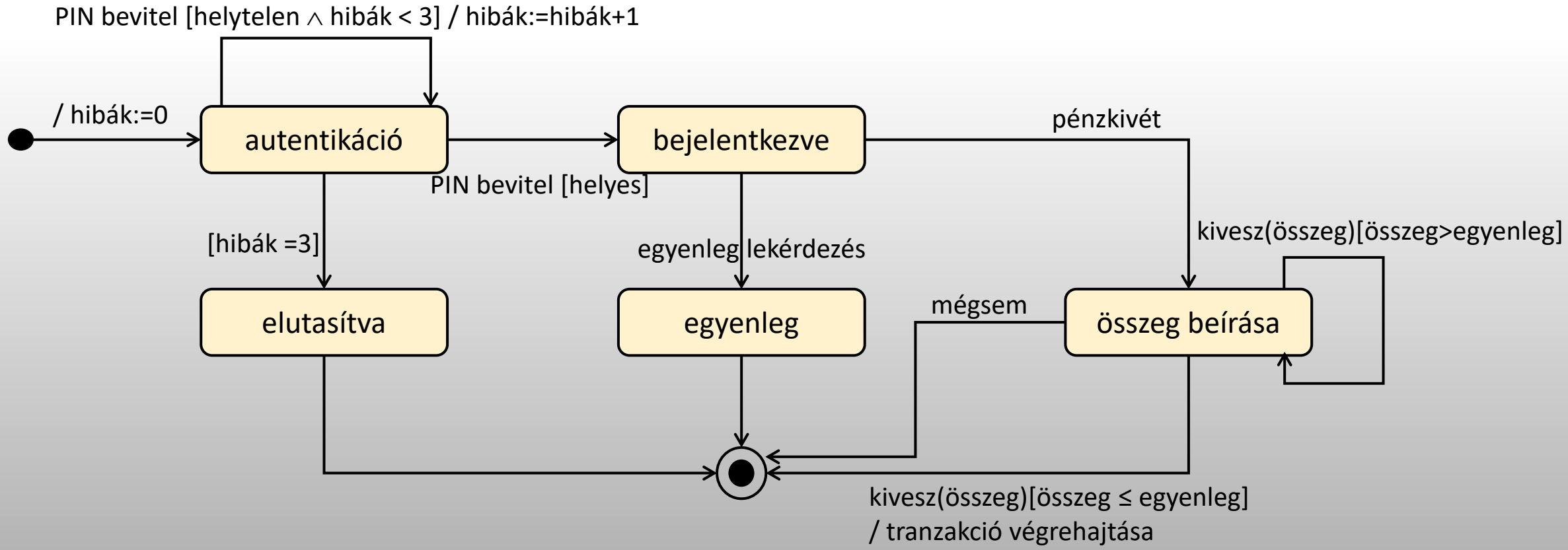
Használati eset diagram elemei

- Használati esetek: a rendszernek egy külső megfigyelő szempontjából azonosítható fő funkciói, szolgáltatásai
- Aktorok: felhasználók vagy külső szereplők, akik a funkciókat használják
- Használati esetek rákövetkezési sorrendje.
  - precede: felhasználó által közvetlenül kezdeményezhető két tevékenység között biztosítandó sorrendet jelöli
  - invoke: egy felhasználói tevékenység és az azt követő, de közvetlenül nem előidézhető tevékenység közötti kapcsolatot jelöli
- Használati esetek közötti tartalmazási viszony.
  - include: egy felhasználó tevékenységnek egy jól elkülöníthető, önállóan is kezdeményezhető része, amely nélkül azonban a tartalmazó tevékenység nem teljes (absztrakt).
  - extend: egy felhasználói tevékenységet opcionálisán kiegészítő másik tevékenység, amely önmagában is teljes (soha nem absztrakt)
- Egyéb:
  - Használati esetek vagy az aktorok közötti származtatás.
  - Multiplicitás is megadható az aktoroknál.

5. Egy bankautomata a következő módon működik. Azzal indul, hogy az ügyfél behelyezi a kártyáját, majd beviszi a pin kódot, amivel háromszor próbálkozhat (harmadik sikertelen kísérlet után a tranzakció elutasítva). Ha sikeres a pin kód megadása, akkor le lehet kérdezni az egyenleget, vagy ki lehet venni pénzt az automatából. Ha a megadott összeg kisebb vagy egyenlő, mint az egyenleg, akkor sikeres a pénz kivét, különben nem.



6. Egy bankautomata a következő módon működik. Azzal indul, hogy az ügyfél behelyezi a kártyáját, majd beviszi a pin kódot, amivel háromszor próbálkozhat (harmadik sikertelen kísérlet után a tranzakció elutasítva). Ha sikeres a pin kód megadása, akkor le lehet kérdezni az egyenleget, vagy ki lehet venni pénzt az automatából. Ha a megadott összeg kisebb vagy egyenlő, mint az egyenleg, akkor sikeres a pénz kivét, különben nem.



6. Egy szavatossági időhöz kötött friss (nem tartós) áru életciklusa. Az áru rendelkezik szavatossági idővel.
   Amikor megérkezik a boltba kiteszik a polcra és árulni kezdik.
   Amikor a lejárati idő közeleg, árleszállítják, reklámozzák.
   Ha lejárt, leveszik a polcról és megsemmisítik.