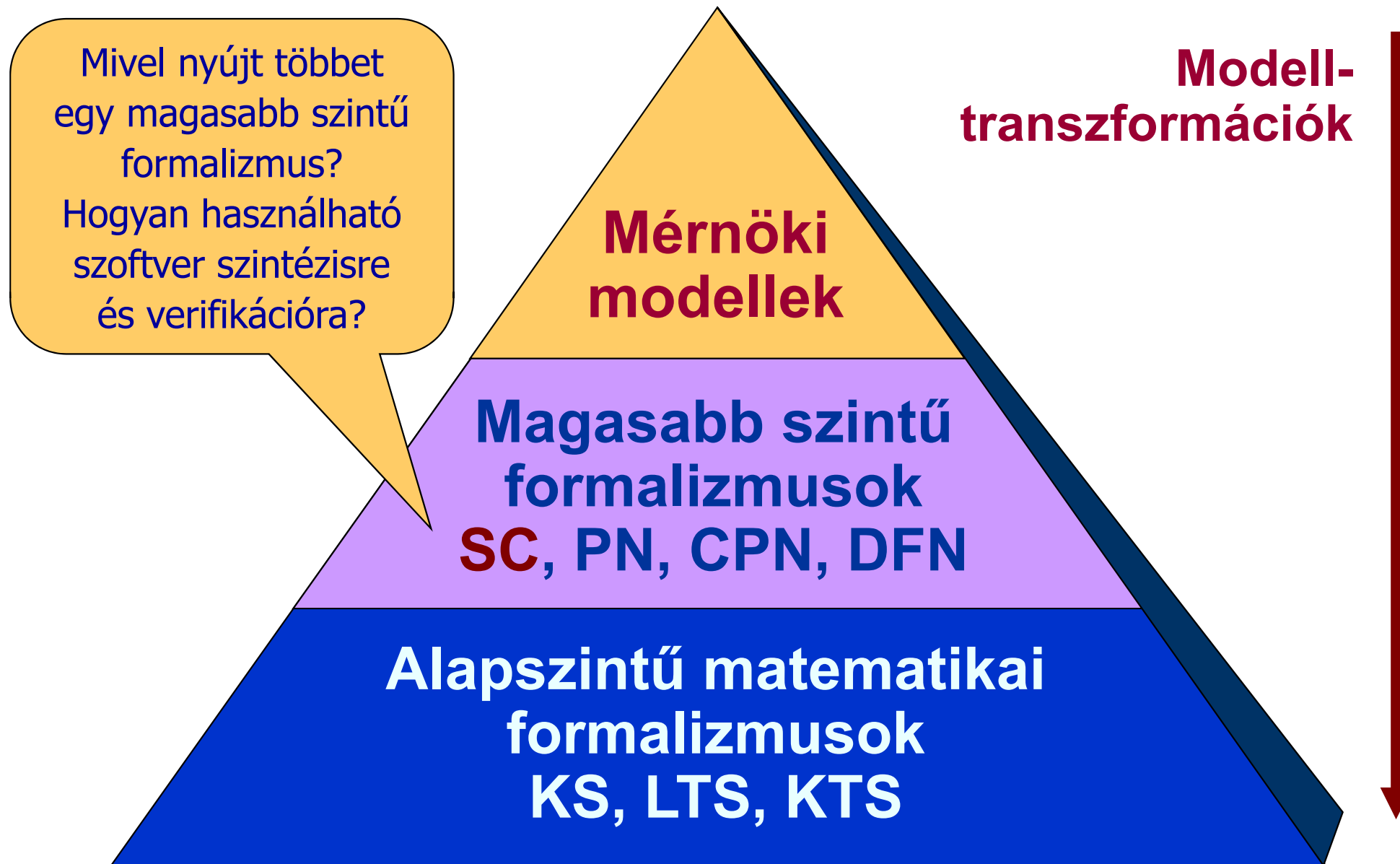


Magasabb szintű formalizmus: Állapottérképek (statecharts)

dr. Majzik István
BME Méréstechnika és Információs
Rendszerek Tanszék

Modellek a formális ellenőrzéshez



Tartalomjegyzék

- Alapelemek
- Az állapottérkép szintaxisa
 - UML 2 statechart diagram
- Az állapottérkép szemantikája
 - UML 2 State Machine szemantika
- A modell használata

Mi az állapottérképek célja?

- **Állapot alapú, eseményvezérelt rendszerek viselkedésének megadására alkalmasak**
 - Egy állapotgép viselkedésének leírása
 - **Reaktív viselkedés:**
Külső események hatására történő állapotváltást ír le
 - Pl. bejövő üzenet, jelzés, hívás, ...
 - **Akciók:** az állapotátmenetekhez rendelt tevékenységek, történések
 - Pl. értékadás, kimenő üzenet, ...
- **Szokásos használat:**
 - **Beágyazott rendszerek:** bejövő események feldolgozása (pl. robot vezérlése, vagyonvédelmi rendszer, ...)
 - **Protokollok:** üzenetek feldolgozása

Alapfogalmak

- **Állapot, aktív állapot**
 - Adott feltételek teljesülése (pl. művelet végrehajtható)
 - Állapotváltozók adott állapota
- **Állapotátmenet**
 - Állapot változása
 - **Trigger esemény** válthatja ki
 - Trigger nélküli átmenet: „önmagától” következik be
 - **Örfeltétel** rendelhető hozzá
 - Állapotátmenet csak akkor történhet meg, ha az örfeltétel igaz
 - **Akció** rendelhető hozzá
 - Állapotátmenethez rendelt tevékenység, történés
- **Esemény**
 - Aszinkron történés, paraméterei is lehetnek
 - Önálló elem, eseményosztály példánya
 - Öröklés: esemény attribútumok bővítése

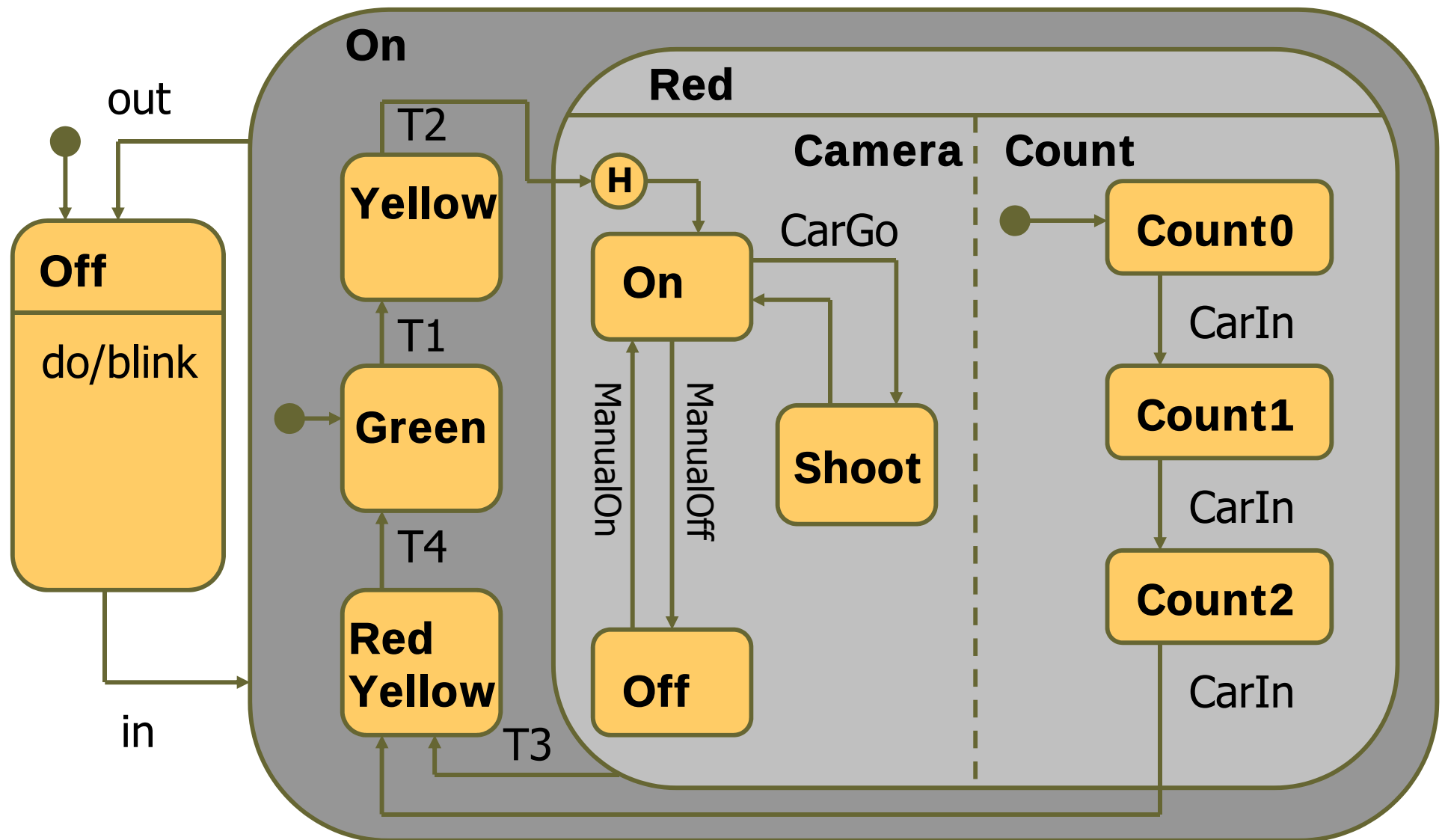
Új igények

- **Állapotok finomítása: Állapothierarchia**
 - Szuperállapot: alállapotokra bontható
- **Konkurens viselkedés leírása**
 - Nem akarunk sorrendi kötöttséget megadni (egyidejűleg, vagy tetszőleges sorrendben végzett feldolgozás)
 - Többszálú / elosztott / párhuzamos végrehajtás esetén
- **Összetett állapotátmenetek**
 - Szétváló, egyesülő, feltételtől függő elágazó átmenet
- **Emlékezés: Visszatérés egy korábbi aktív állapotkonfigurációra**
 - Visszatérés adott feldolgozáshoz közbenső esemény után
 - Egy állapotfinomítási szinten vagy mélyebben is

Állapotdiagramok és állapottérképek

- **Állapotdiagram:**
 - Egyszintű, egyszerű állapotok és átmenetek
 - Pl. UPPAAL esetén látott automaták leírása
- **Állapottérkép:** az állapotdiagram kiterjesztése
 - **Állapothierarchia:** állapotok finomítása
 - **Konkurens régiók:** konkurens viselkedés leírása
 - **Összetett átmenetek:** szétváló, egyesülő, feltételes
 - **Emlékezés:** Legutolsó aktív állapotkonfiguráció tárolása
 - Szintaktikai segédelemek
 - Ritkán használt (nem intuitív) kiegészítések
 - Késleltetett esemény
 - Szinkronizációs állapot
 - ...

Egy állapottérkép



Az állapottérképek szintaxisa (UML szerinti szintaxis)

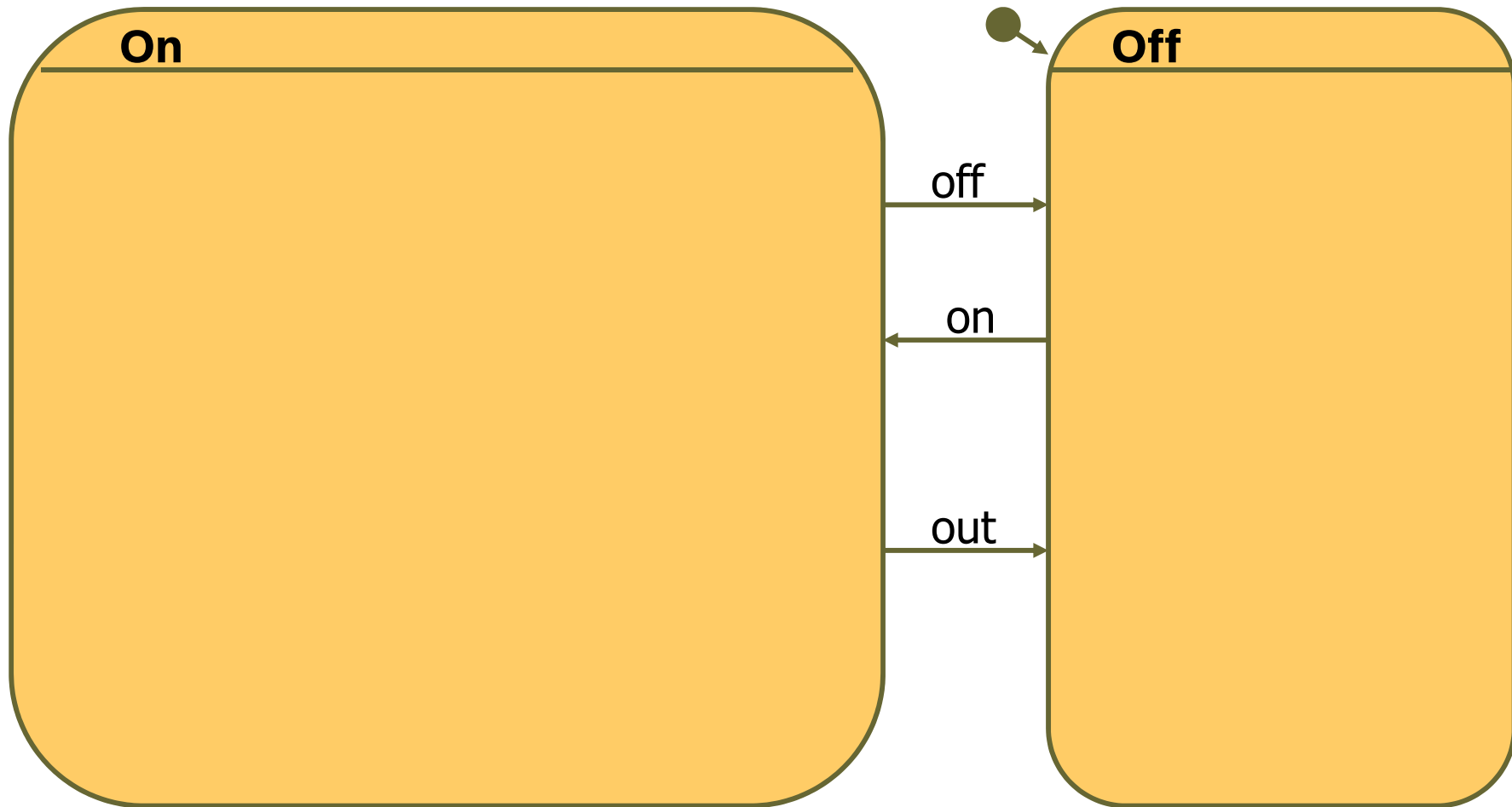
Állapotok: Akciók és állapotfinomítás

- Állapotok: Alapszintű modellelem
- Állapotokhoz kötődő akciók:
 - Belépési akció (**entry** / ...)
 - Kilépési akció (**exit** / ...)
 - Belső akciók (**do** / ...)
- Állapotfinomítás
 - Egyszerű állapot: nincs finomítása
 - OR jellegű finomítás: alárendelt állapotok
 - Ezek közül egyszerre egy állapot lehet aktív
 - AND jellegű finomítás: konkurens régiók
 - Egyidejűleg minden egyes régióban kell legyen aktív állapot!

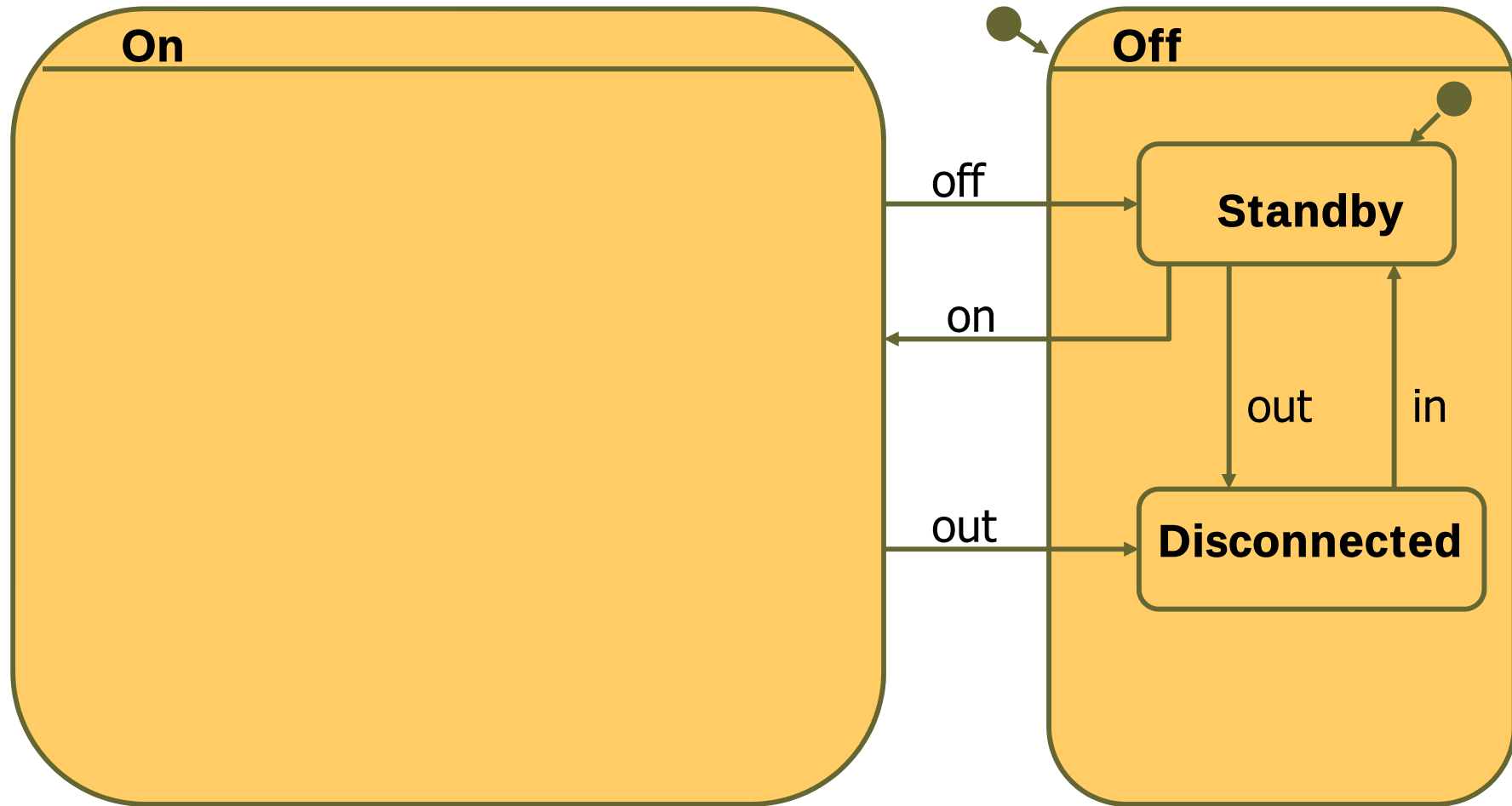
print_job

entry / init()
exit / reset()
do / poll()
job / print()

Példa: Állapotfinomítás

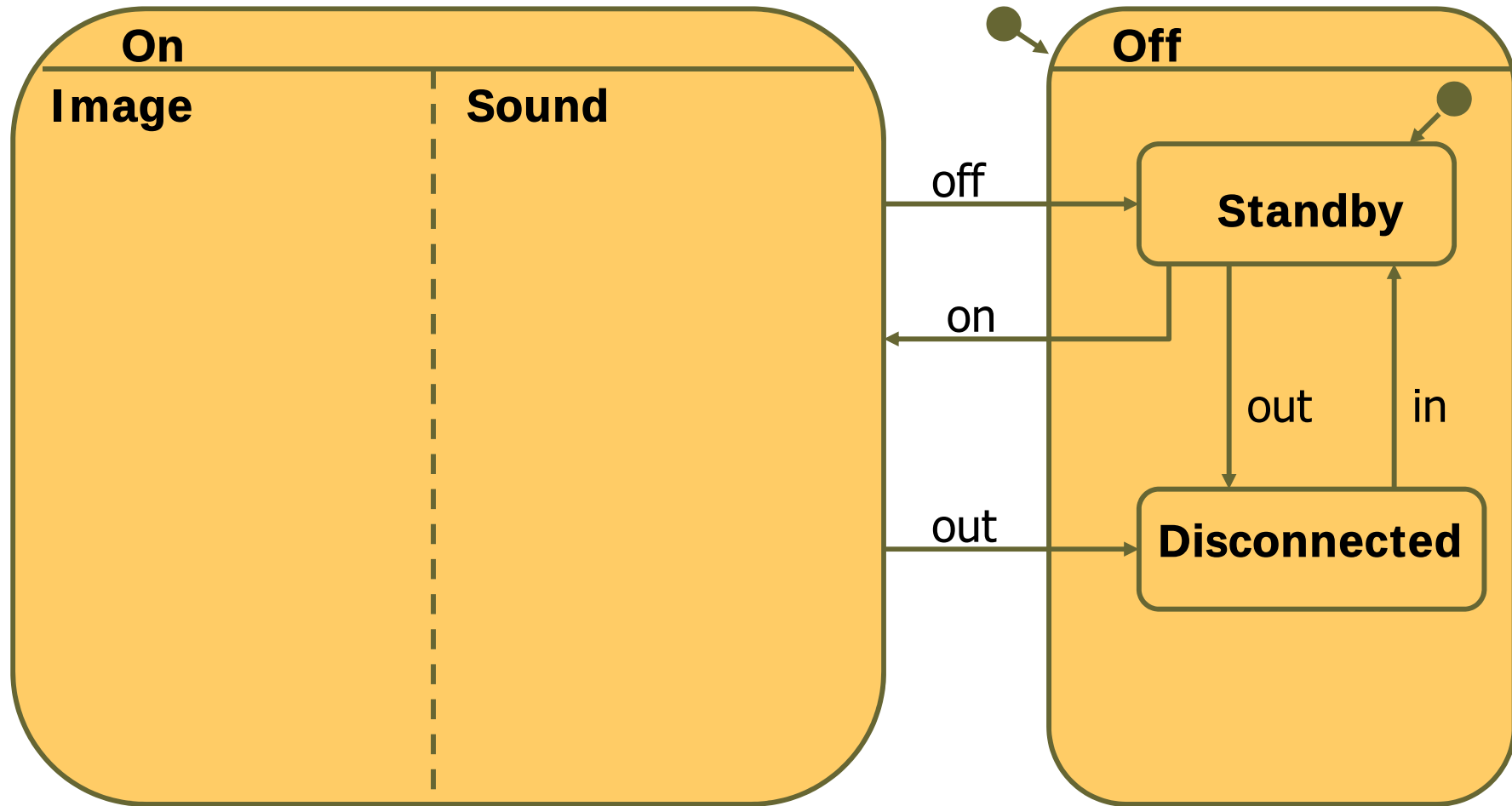


Példa: Állapotfinomítás



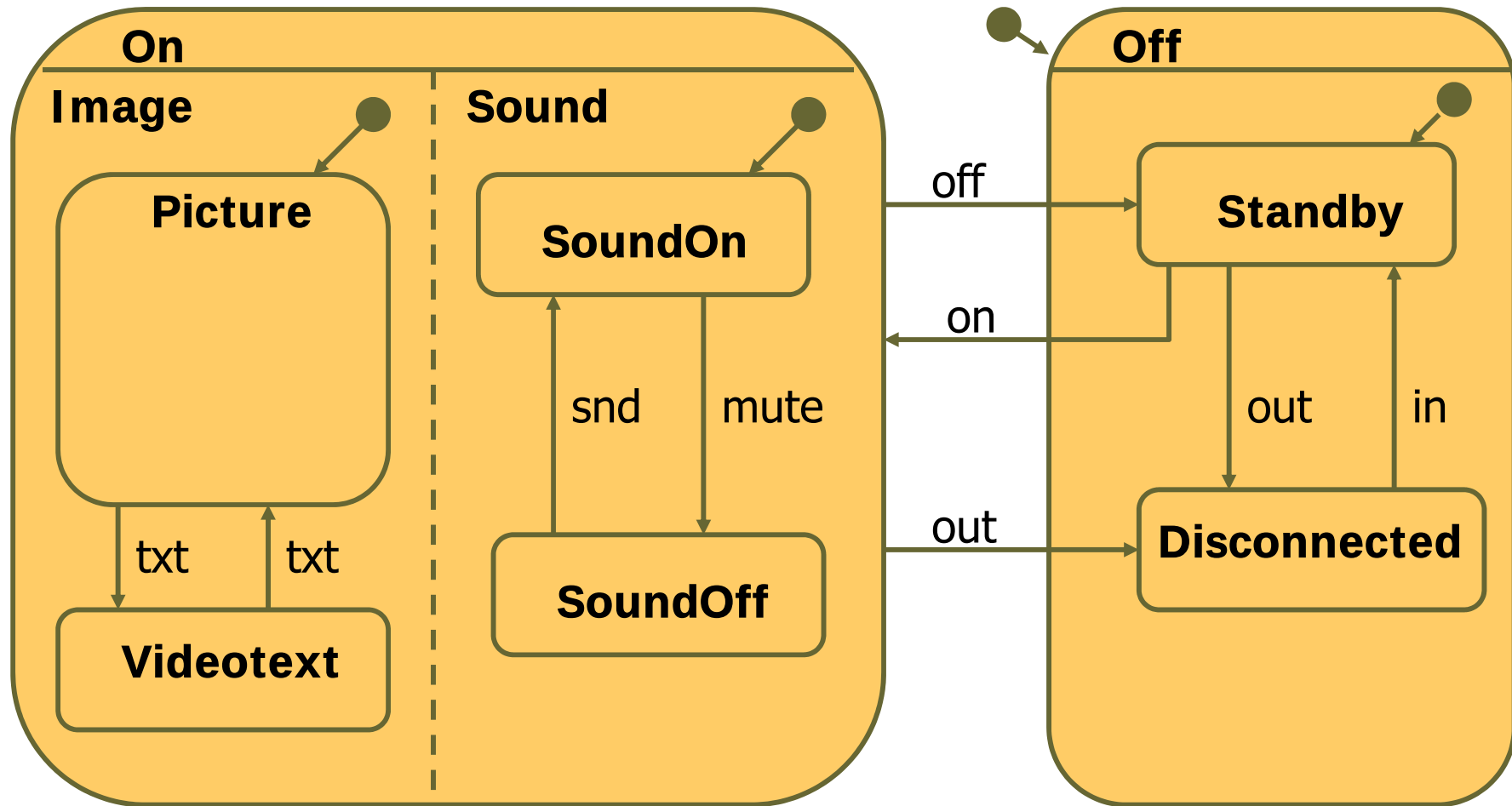
OR jellegű finomítás

Példa: Állapotfinomítás



AND jellegű finomítás

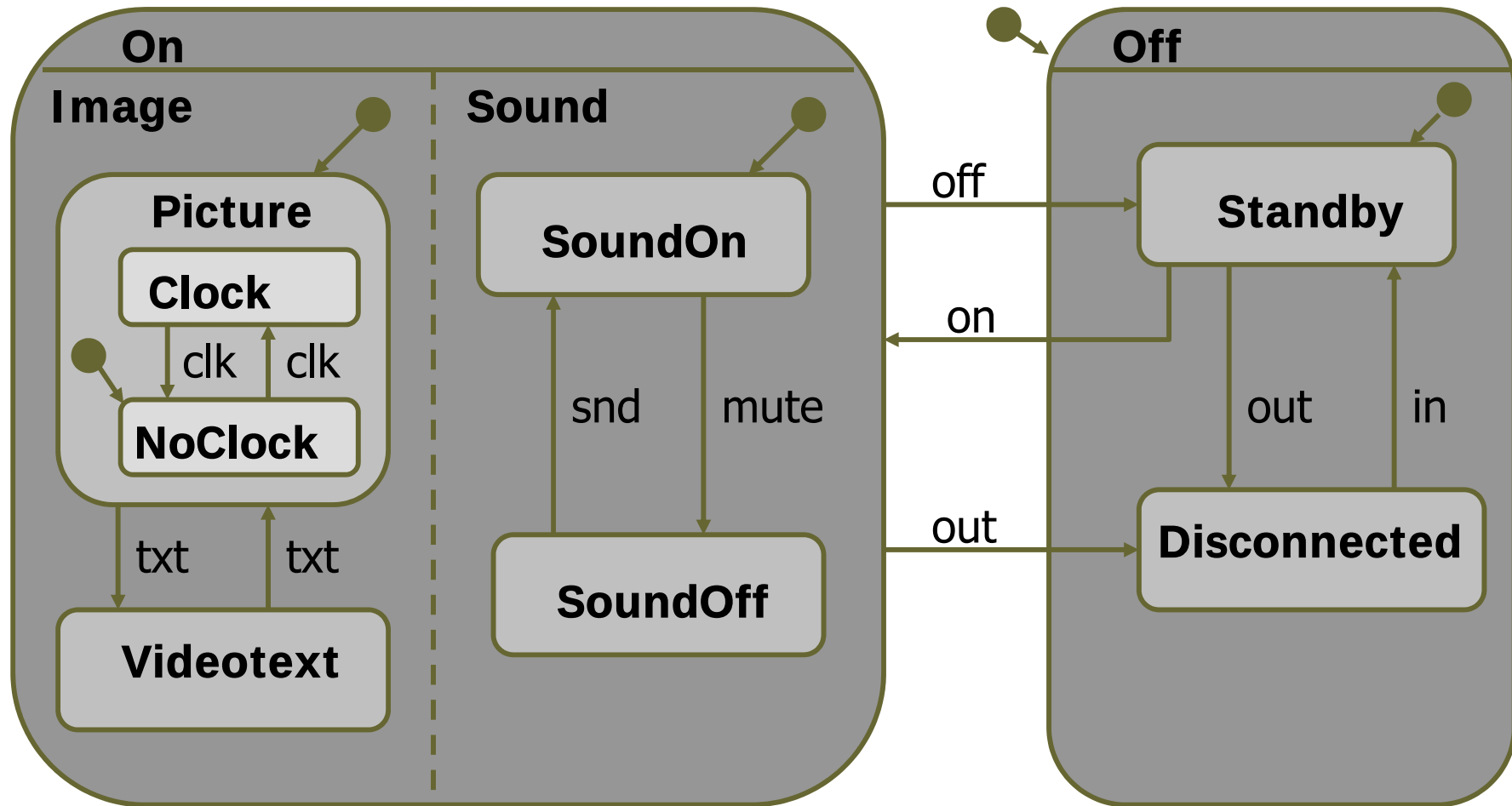
Példa: Állapotfinomítás



AND jellegű finomítás

OR jellegű finomítás

Példa: Állapotfinomítás

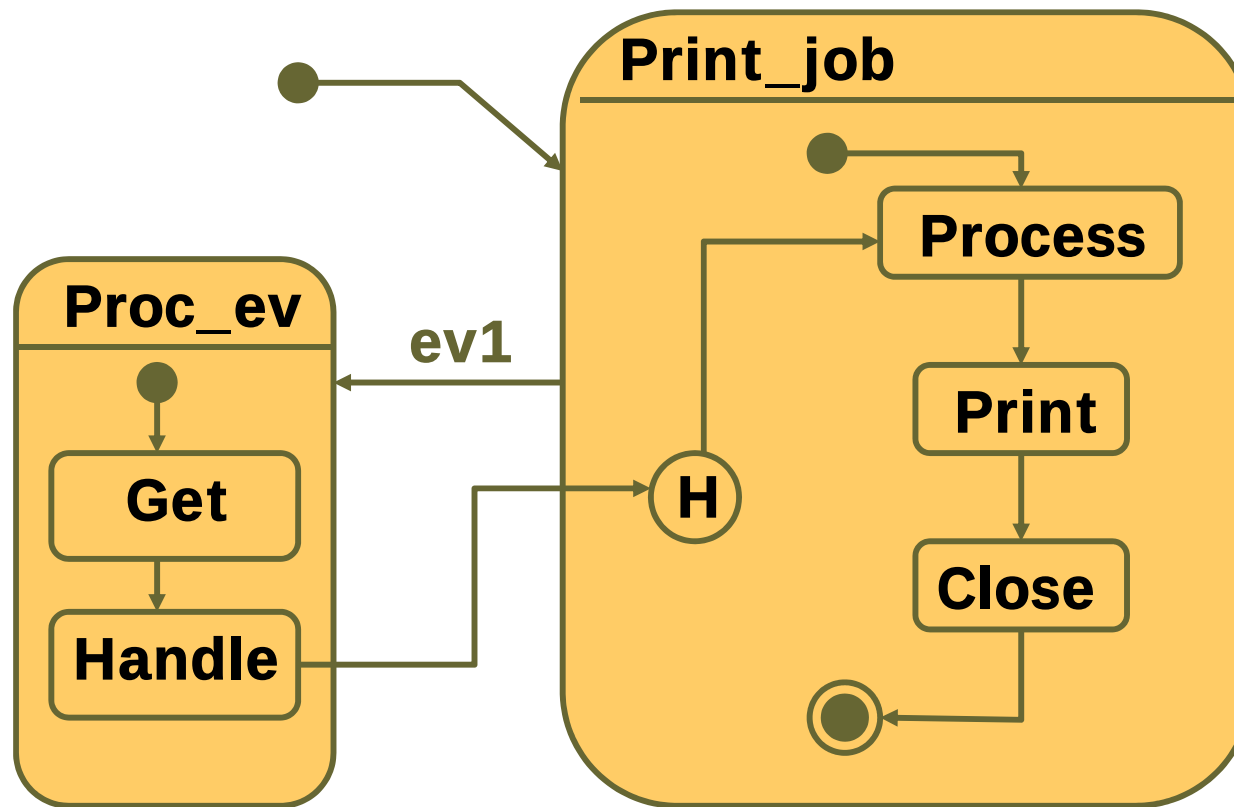


Pszedo-állapotok

- **Kezdőállapot** jelzés: régióba való belépéskor lesz aktív
 - Minden OR finomításban egy
 - Minden AND régióban egy
- **Végállapot** jelzés: állapotgép terminálás
- **Emlékező** állapotok (history state)
 - A legutolsó aktív állapotkonfigurációt „tárolja”
 - **Egyszerű** emlékező állapot: csak az adott finomítási szinten
 - **Mélyen emlékező** állapot: a mélyebb finomítási szinteket is
 - Mit jelent az emlékező állapothoz húzott **bemenő** átmenet?
 - Tüzeléskor a „tárolt” állapotkonfigurációba kerül az objektum
 - Az emlékező állapot egy „hivatkozás” a tárolt állapotkonfigurációra
 - Mit jelent az emlékező állapotból húzott **kimenő** átmenet?
 - Alapértelmezett „tárolt” állapotot jelöl ki arra az esetre, ha előzőleg még nem volt aktív állapot

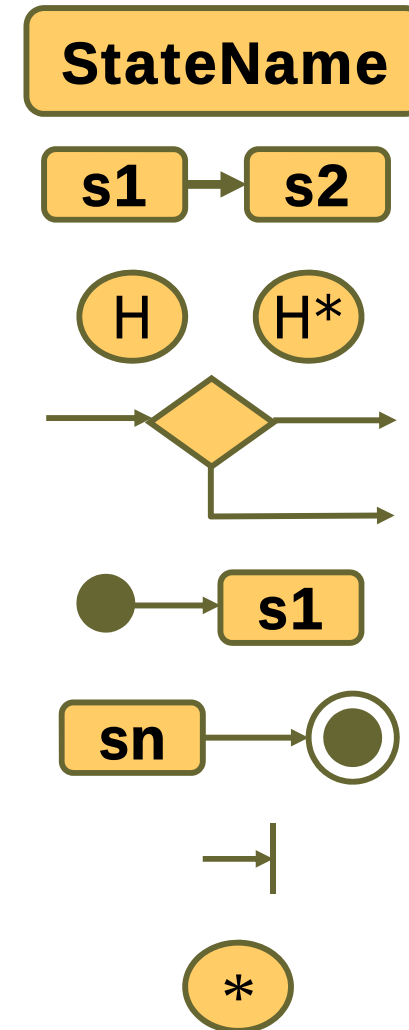


Példa: Emlékező állapot



Állapottérképek elemei

- Állapot
- Állapotátmenet
- Emlékező állapot
- Feltétel
- Kezdőállapot
- Végállapot
- Állapotcsonk
- (Szinkronizáció)



(Állapot)átmenetek

- Állapotátmenetek megadása
- Szintaxis:

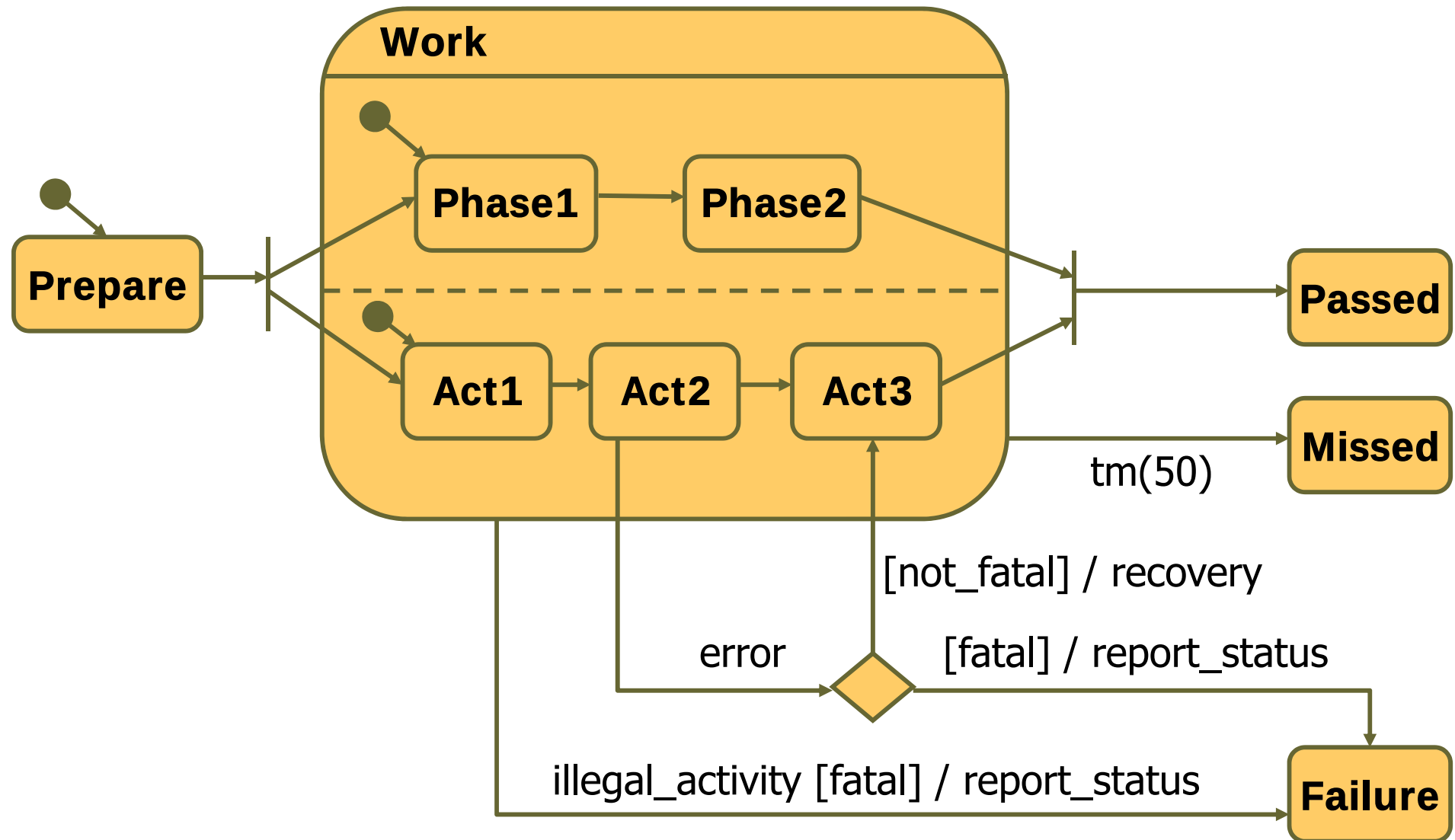
trigger [guard] / **action**

- **trigger**: kiváltó esemény
- **guard**: az átmenet őrfeltétele
 - Predikátum az állapotváltozók és esemény paraméterek felhasználásával
 - Állapotra való hivatkozás is lehet: `is_in(state)`
- **action**: akció (művelet)
 - akció szemantika: művelet részletezése

Átmenetek specialitásai

- Time-out mint trigger:
 - Fennáll, ha az objektum az átmenethez tartozó kiindulási állapotban tartózkodik végig az adott időintervallumban
- Összetett átmenetek:
 - Szétváló (fork): konkurens régiókban lévő állapotokba való együttes belépés
 - Egyesülő (join): konkurens régiókban lévő állapotokból való együttes kilépés
 - Elágazó (condition): több, őrfeltételtől függő átmenet egyszerűsített jelölése (szegmensek)
- Állapothierarchián átívelő átmenetek
 - Megengedett (nem elegáns)

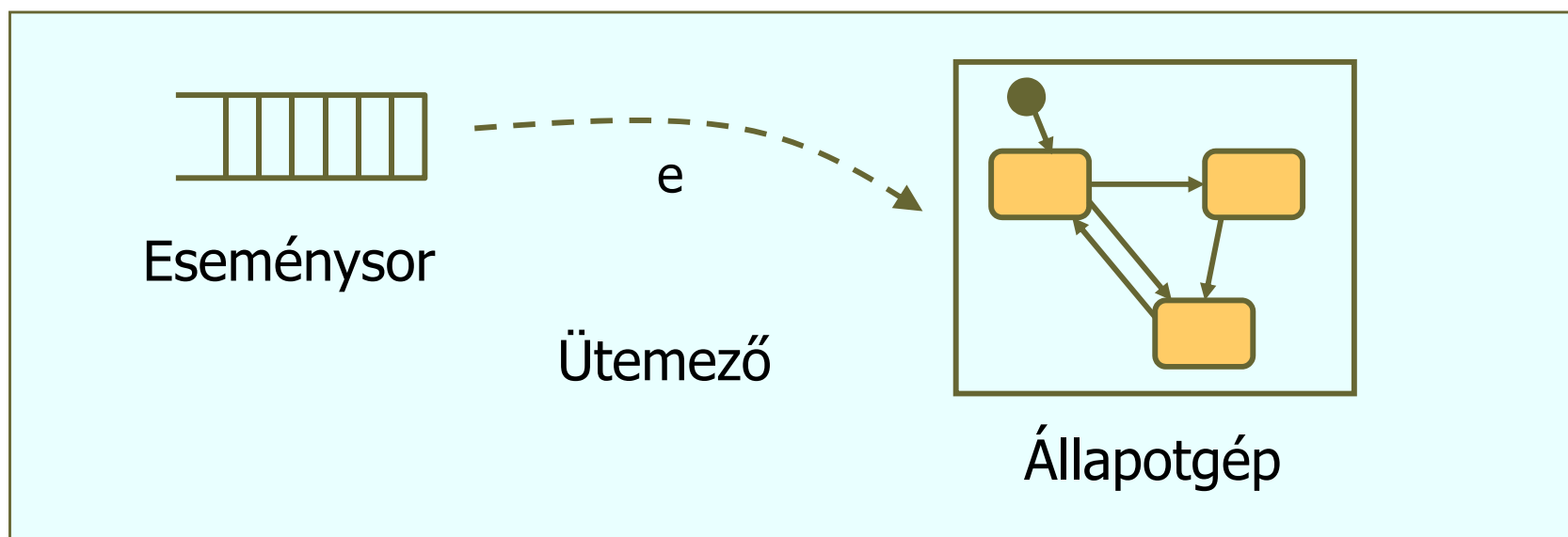
Példa: Állapotátmenetek



Az állapottérképek informális szemantikája (UML szerinti szemantika)

Szemantika: Hogyan működik?

- Alapelemek:
 - **Állapotgép:** Az állapottérkép írja le a viselkedését
 - **Eseménysor + Ütemező:** „Futtató rendszer”
(az állapotgép szempontjából külső elemek)



Mit ad meg a szemantika?

Mit tesz az állapotgép egy esemény hatására

→ állapotgép egy **lépése**

- **Állapotátmenetek tüzelnek**
 - Újdonság: egy esemény hatására több konkurens állapotátmenet tüzelhet
- **Állapotkonfiguráció változik**
 - **Több aktív állapot lehet**
 - Aktív állapot minden régiójában kell legyen egy aktív állapot
 - Aktív állapot OR finomításában kell legyen egy aktív állapot
 - Egy OR finomításban illetve egy régióban csak egy aktív állapot lehet
 - Rekurzívan érvényes

A szemantika alaptulajdonságai

- Egyenként feldolgozott események
 - Az ütemező akkor küld újabb eseményt, ha az előző feldolgozása teljes egészében megtörtént
 - Stabil állapotkonfiguráció kialakult: nincs trigger nélküli átmenet
- Események teljes feldolgozása (run to completion)
 - Átmenetek maximális halmaza tüzel
 - Minden engedélyezett átmenet tüzel, kivéve ha ezt konfliktus megakadályozza
 - Ezek tüzelése után dolgozza fel a következő eseményt
- Az eseményfeldolgozás a szemantika lényege
 - Ez alapján lehet programkód alakjában megvalósítani az állapotgépet (forráskód generálás)

Az eseményfeldolgozás lépései 1/4

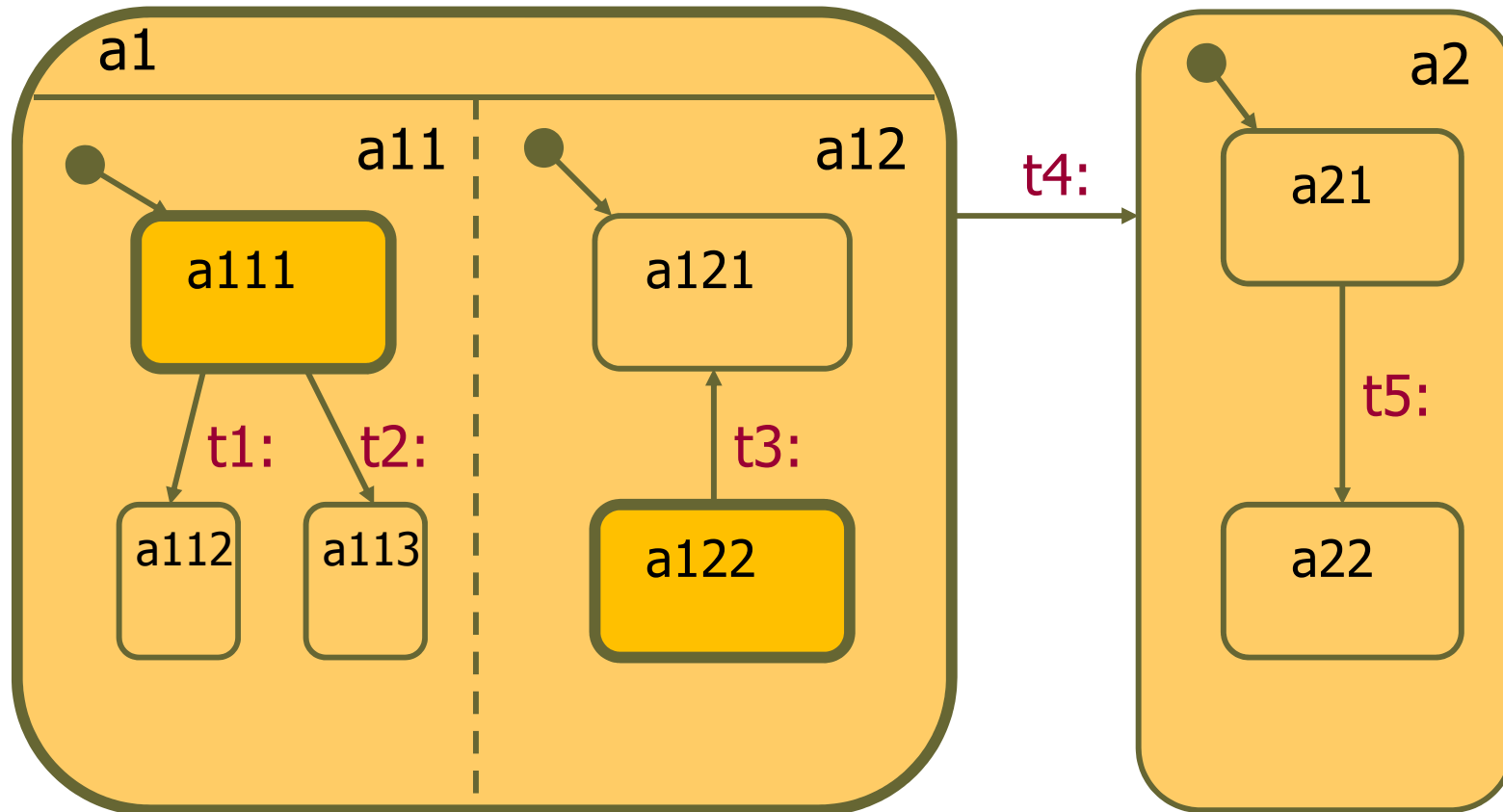
- Külső feltétel: Ütemező a stabil konfigurációban lévő állapotgépnek egy eseményt juttat
- **Engedélyezett** átmenetek:
 - Kiindulási (forrás-) állapot aktív
 - A kiválasztott esemény az átmenet triggere
 - Az őrfeltételek teljesülnek

Esetek az engedélyezett átmenetek száma alapján:

- Ha csak egy van: Tüzelhet!
- Ha nincs: Halasztott-e az esemény?
 - Igen: tárolás, új esemény kérése az ütemezőtől
 - Nem: esemény eldobható (hatás nélküli)...
- Ha több van: Tüzelő átmenetek **kiválasztása** szükséges
 - Befolyásol: A konfliktus

Példa: Konfliktus

A $t1, \dots, t5$ átmenetek ugyanazon e eseményre triggereltek.
Melyek nem tüzelhetnek együtt?



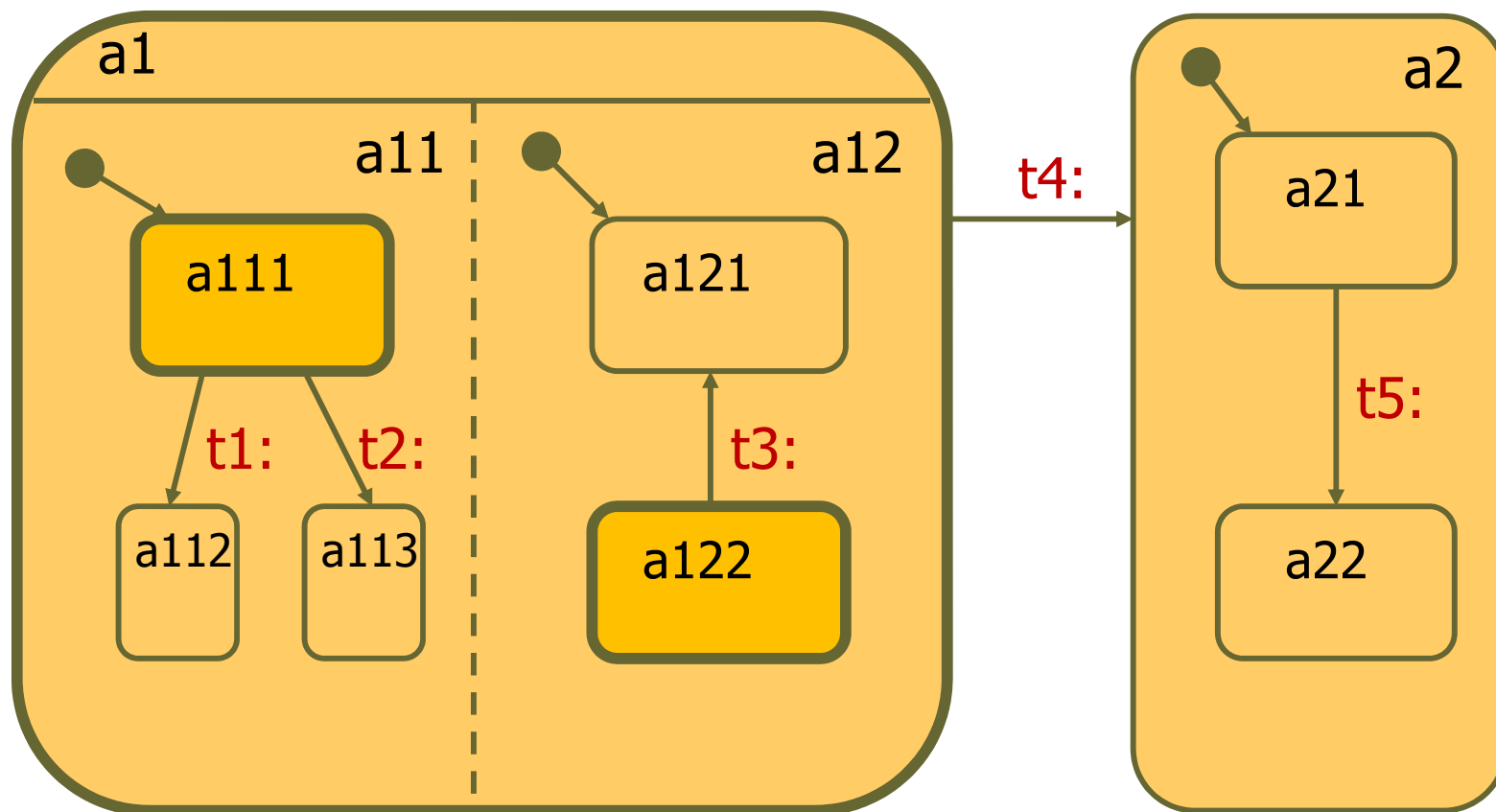
- Nem engedélyezett: $t5$ (forrásállapota nem aktív)
- Egyszerre nem tüzelhetnek: $(t1, t2)$; $(t1, t4)$; $(t2, t4)$; $(t3, t4)$
- Egyszerre is tüzelhetnek: $(t1, t3)$; $(t2, t3)$;

Eseményfeldolgozás lépései 2/4

- **Tüzelő átmenetek kiválasztása:**
 - Maximális számú átmenet, nem lehet közöttük konfliktus
 - Konkurens átmenetek szimultán tüzelése
- **Konfliktusban lévő átmenetek:**
 - Ugyanazt az állapotot hagyják el, pontosabban az elhagyott állapothalmazok metszete nem üres
- **Konfliktus feloldása:**
 - **Prioritás alapján:** egy átmenet prioritása nagyobb, ha az átmenet kiindulási állapota az állapothierarchiában (finomításban) **alacsonyabb szintű**
 - OO koncepció: a finomítás „felüldefiniál”
 - **Véletlenszerű választás,** ha azonos prioritásúak

Példa: Konfliktusfeloldás

A **t1**, ..., **t5** átmenetek ugyanazon eseményre triggereltek.
Melyek tüzelhetnek együtt?

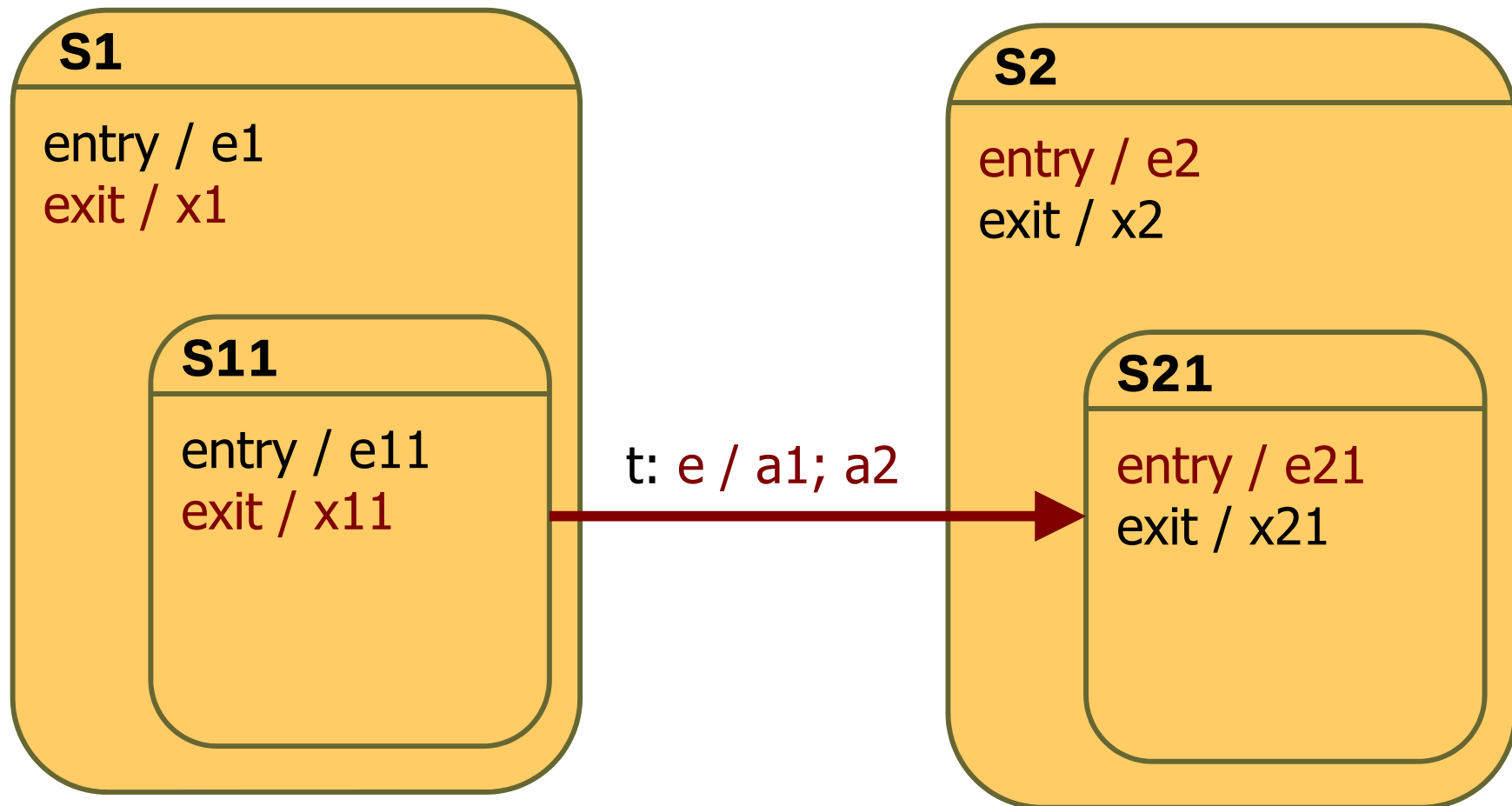


- Nagyobb prioritású t4-nél: t1 és t2
- Tüzelhet: (t1,t3) vagy (t2,t3)

Eseményfeldolgozás lépései 3/4

- Kiválasztott átmenetek **tüzelnek**:
 - Sorrend véletlenszerű (nincs köztük konfliktus)
 - Akció végrehajtási sorrend nemdeterminisztikus
- Egy átmenet **tüzelése**:
 1. Kiindulási állapotok **elhagyása**
 - Alacsonyabb hierarchiaszinten először
 - Kilépési akciók végrehajtása (**exit** akciók)
 2. Átmenetek akcióinak **végrehajtása**
 3. Céllállapotokba való **belépés** → új konfiguráció
 - Magasabb hierarchiaszinten először
 - Belépési akciók végrehajtása (**entry** akciók)

Példa: Akciók sorrendezése

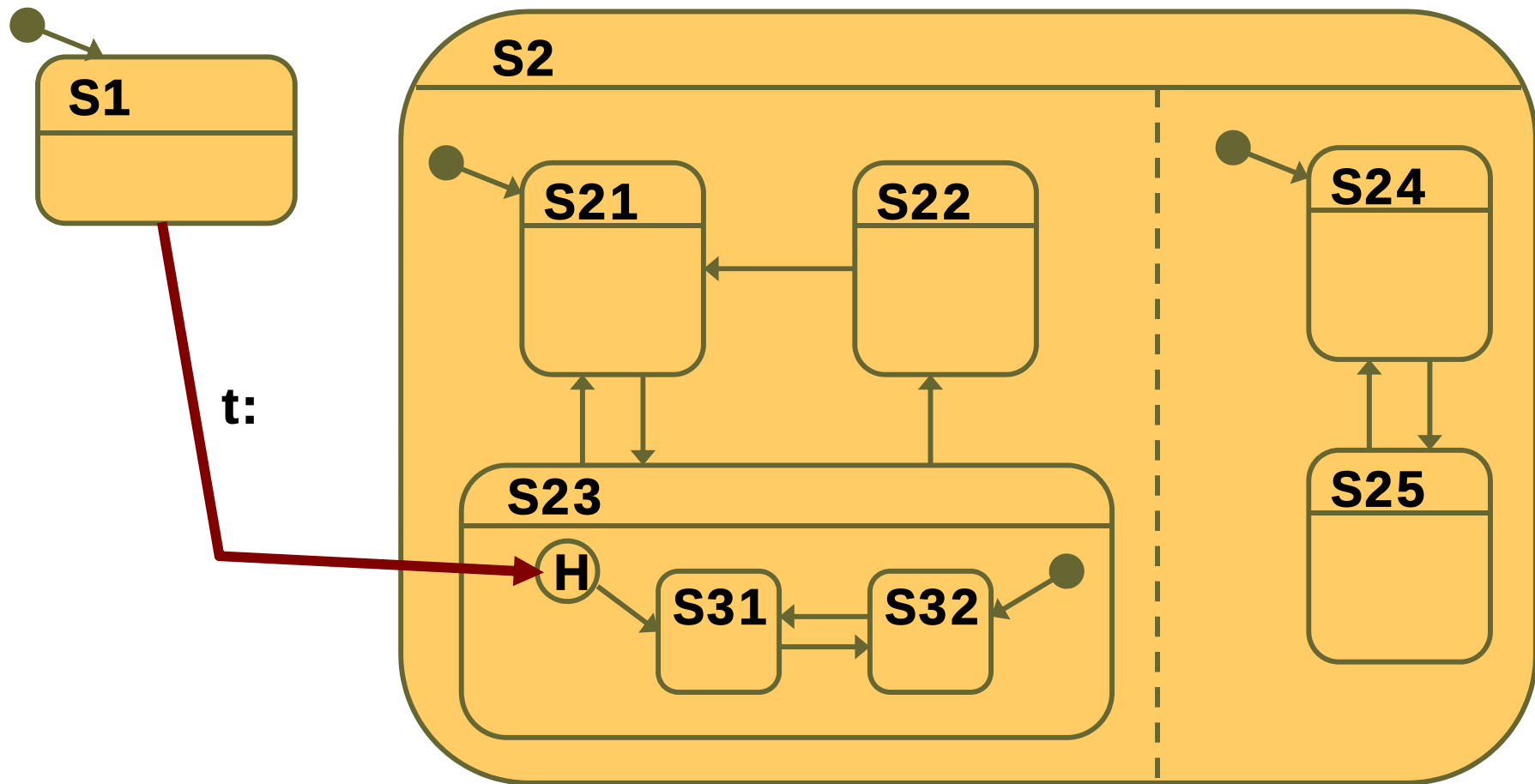


Akciók sorrendje: x11; x1; a1; a2; e2; e21

Eseményfeldolgozás lépései 4/4

- **Belépés** új konfigurációba különféle jellegű célállapotok esetén:
 - Ha **egyszerű** (nem finomított) a célállapot:
 - Új konfiguráció része lesz
 - Ős állapotai (amelyek finomításában szerepel) is aktívak lesznek
 - Aktívvá váló ős állapotok minden régiójában lesz aktív állapot (kezdőállapot jelöli ki)
 - Ha **OR finomítása van** a célállapotnak:
 - A finomításban a kezdőállapot jelöli ki az aktív állapotot
 - Ha **AND finomítása van** a célállapotnak:
 - Minden régiójában kell legyen aktív állapot (kezdőállapot jelöli ki)
 - Ha **emlékező** állapot:
 - Az utoljára elhagyott állapotkonfiguráció lesz újra aktív
 - Ha nem volt még ilyen: a kimenő él határozza meg
 - Ha **nem stabil** az állapot: azonnali továbblépés

Példa: Belépés konkurens állapotba

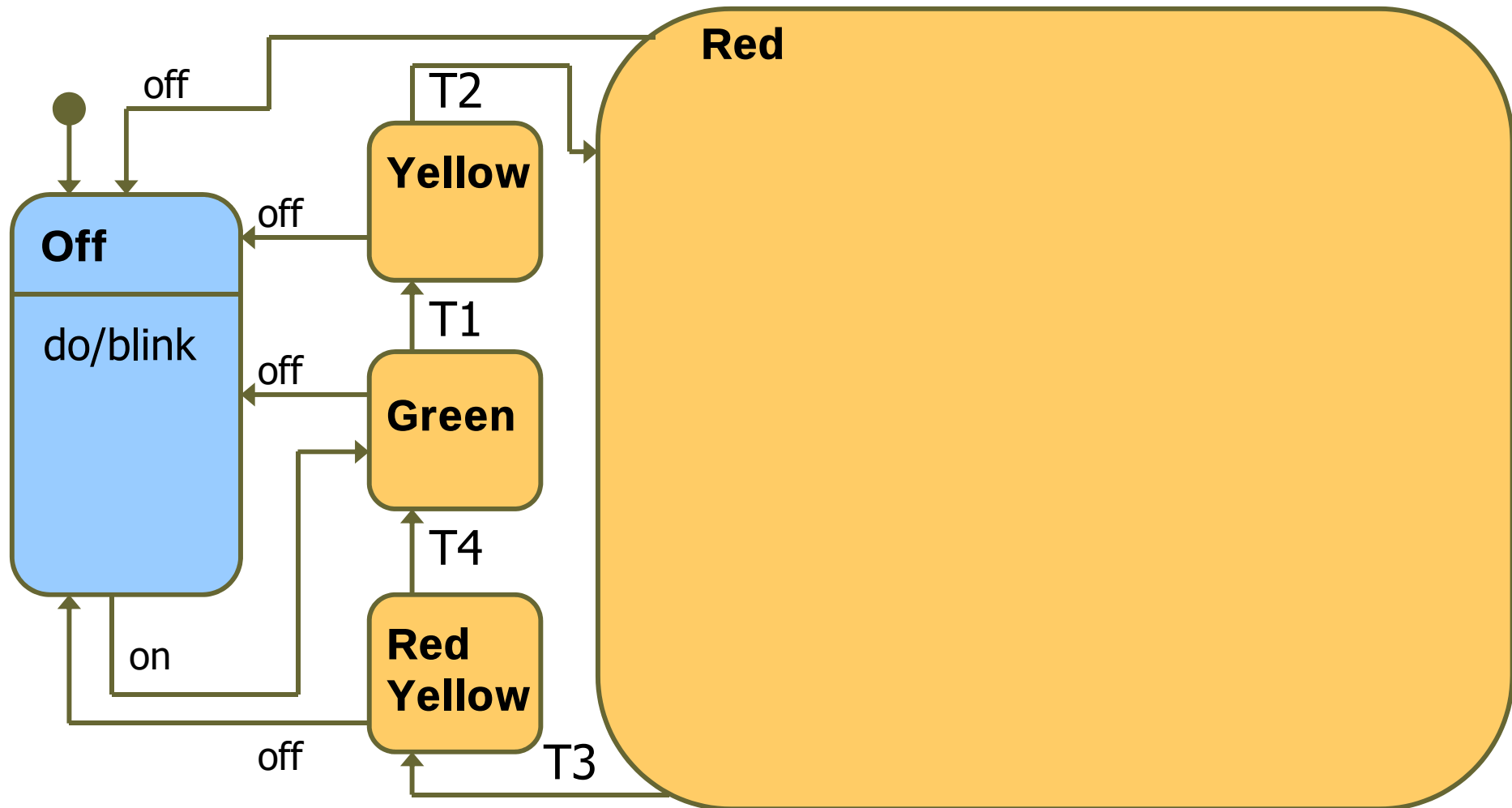


A **t** átmenet tüzelése után mi lesz az új állapotkonfiguráció?

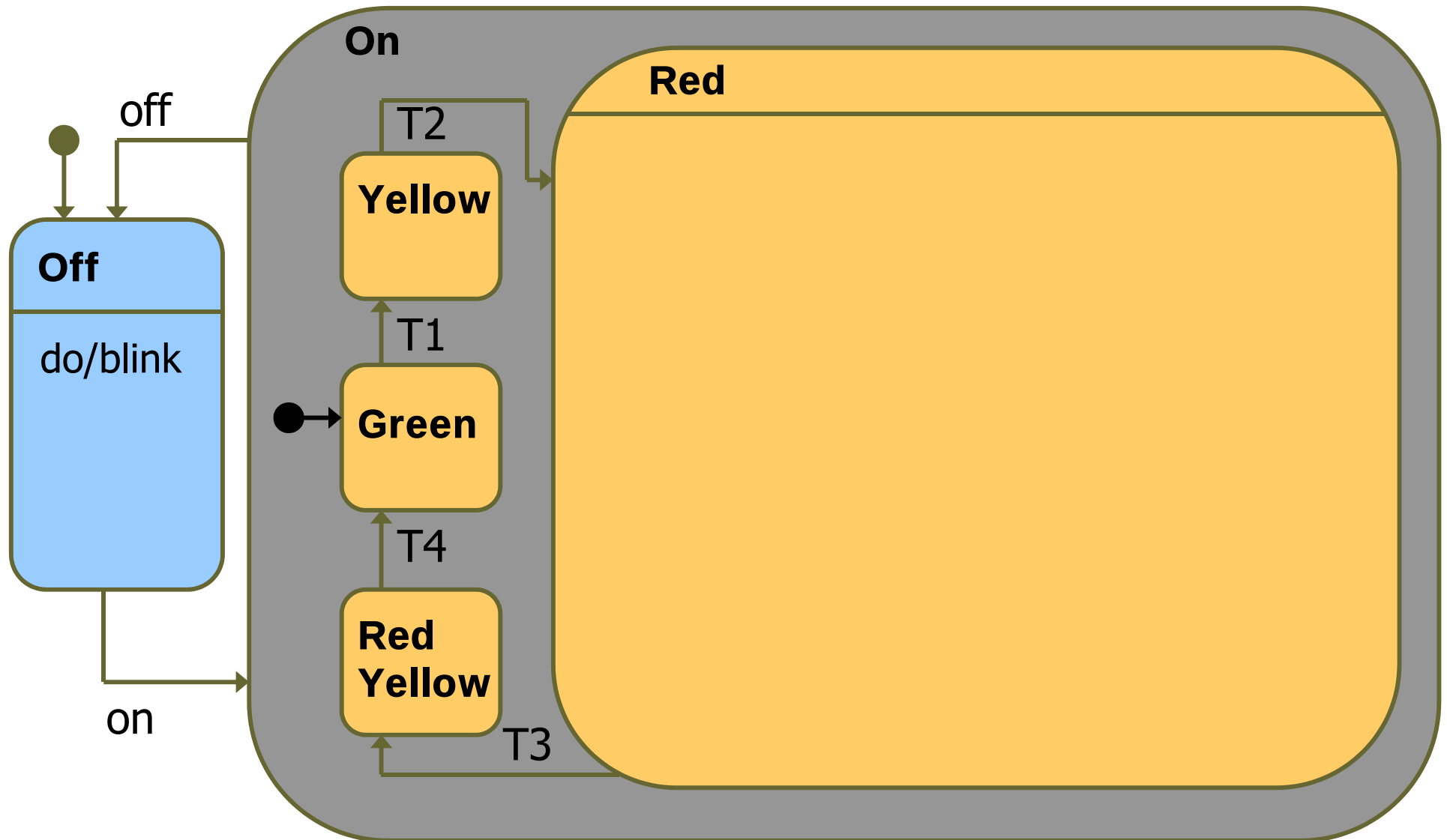
Mintapélda

- Közlekedési lámpa vezérlője egy főútvonal és egy mellékútvonal kereszteződésében
 - Bekapcsoláskor: kezdetben főútvonal számára zöld
 - Kikapcsoláskor: villogó sárga
 - Zöld, sárga, piros váltás: időzítő eseményre
 - Főútvonalon 3 várakozó: időzítőtől függetlenül zöld jelzés szükséges
 - Főútvonalon tilosban áthajtók: fényképezés
 - Ez a funkció kézzel ki-be kapcsolható

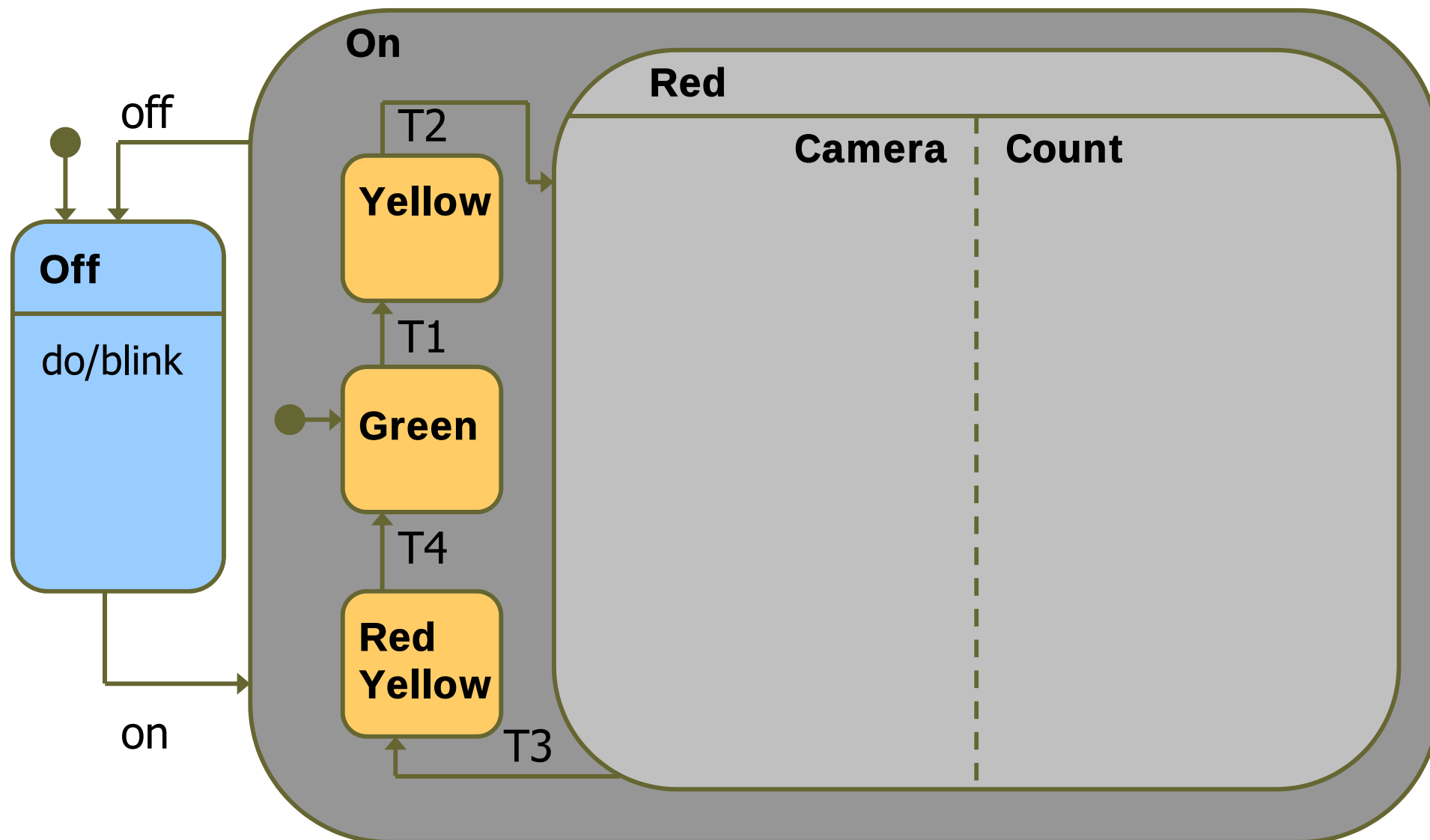
1. Lámpaváltás



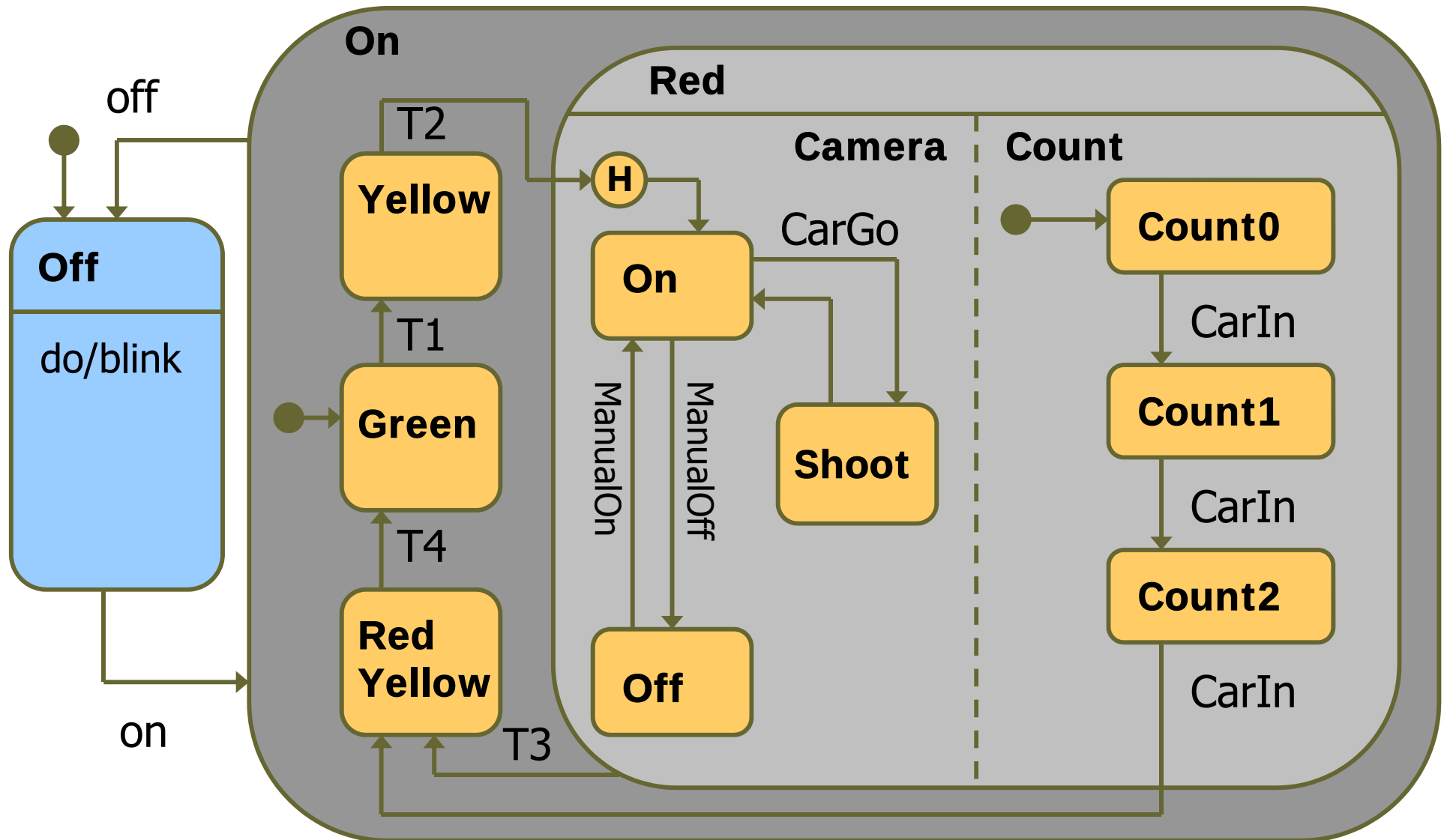
2. Állapothierarchia



3. Konkurens állapotok



4. Teljes vezérlő



Állapottérképek szerepe az UML 2 esetén

- Állapot alapú, eseményvezérelt működés leírása
 - Aktív objektumok viselkedésének leírása
- Felhasználás:
 - Viselkedés szintű kódgenerálás
 - Viselkedés verifikációja
- Akciók formalizálása: UML 2 Action Semantics
 - Metódushívás
 - Attribútum olvasás, írás
 - ... (sokféle művelet)

Állapottérképek alapjai (összefoglalás)

- Kiterjesztések
- Állapottérkép szintaxis
 - Állapothierarchia
 - Konkurens régiók
 - Összetett átmenetek
 - Emlékező állapotok
- Állapottérkép informális szemantika
 - Átmenetek engedélyezettsége
 - Tüzelő átmenetek kiválasztása
 - Átmenetek tüzelése
 - Új állapotkonfiguráció kialakulása