

UML

Gregorics Tibor

gt@inf.elte.hu

<http://people.inf.elte.hu/gt/oep>

UML diagramjai

- ❑ Az UML (*Unified Modeling Language*) egy grafikus nyelv, amely az objektumelvű szoftverfejlesztési folyamat fázisainak (elemzésnek, tervezésnek, megvalósításnak, tesztelésnek) a dokumentálására szolgál.
- ❑ A nyelv diagramjaival egy készülő vagy már elkészült szoftver, illetve az azt körülölelő rendszer egyes részeit lehet más-más nézőpontból bemutatni; a diagramok így egy elosztott dokumentációt alkotnak.
- ❑ A diagramokat aszerint szokás csoportosítani, hogy a rendszer szerkezetét vagy viselkedését írják-e le:
 - **szerkezeti nézet diagramjai**: pl. objektum-, osztály-, komponens-, kihelyezési-, csomag diagram
 - **viselkedési nézet diagramjai**: pl. használati eset-, kommunikációs-, szekvencia-, állapotgép-, tevékenység diagram

A kurzuson használt diagrammok

- **használati eset diagram** (elemzés)
 - ez az egyetlen nem objektumelvű nézet, amely a feladat fő funkcióit írja le
- **objektum diagram** (elemzés)
 - egy adott időpillanatban létező objektumokat és azok kapcsolatait jellemzi
- **kommunikációs diagram** (elemzés)
 - objektumok közötti üzenetváltásokat (szignál küldés, metódus-hívás) ábrázolja
- **osztály diagram** (elemzés, tervezés, megvalósítás)
 - az objektumok és a közöttük létrejövő kapcsolatok általános leírására szolgál
- **szekvencia diagram** (elemzés, tesztelés)
 - objektumok közötti üzenetváltások lehetséges forgatókönyvét (szkenárióját) mutatja be egy időtengelyen
- **állapotgép diagram** (tervezés)
 - egy objektumnak az állapot-változásait, azaz a működését szemlélteti a neki küldött üzenetek (pl. egy metódusának hívása, egy szignál küldés) hatására.

UML

1.rész

Használati eset diagram

Gregorics Tibor

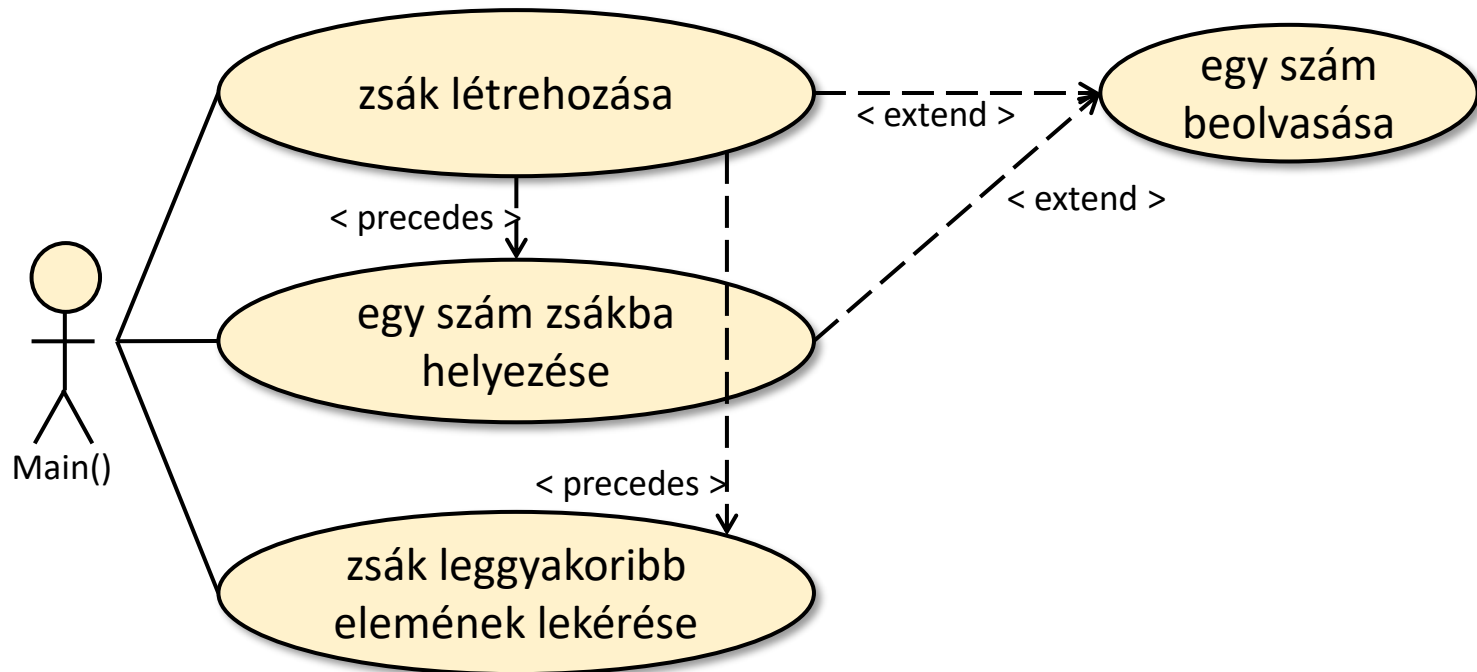
gt@inf.elte.hu

<http://people.inf.elte.hu/gt/oep>

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
- milyen (tartalmazási, rákövetkezési) kapcsolatoknak kell fenn állni az egyes funkciók között



Használati eset diagram elemei

- ❑ **Használati esetek:** azon tevékenységek (funkciók, szolgáltatások), amelyek a rendszert kívülről szemlélő (azaz a felhasználó vagy más külső szereplő) szempontjából azonosíthatóak, és kezdeményezhetőek.
- ❑ **Kapcsolatok** a használati esetek között, amelyek a használati esetek
 - részekre bontását, illetve plusz funkciókkal történő kiegészítését ábrázolják
 - sorrendjét határozzák meg
 - általános használati esetből történő származtatására mutatnak rá
- ❑ **Aktorok:** a funkciókat használó felhasználók vagy más külső szereplők, akik vagy amik a tervezett rendszer működését vezérik. Jelölhető az, hogy egy aktor az aktorok mely típusához tartozik (specializálás), illetve az, hogy egy adott típusú aktorból hány darab lehet (multiplicitás).

Használati esetek kapcsolatai

□ Használati eset **részei**

- **include**: rámutat egy használati esetnek egy olyan jól elkülöníthető, önállóan is kezdeményezhető részére, amely nélkül a tartalmazó tevékenység már nem lenne teljes (hanem absztrakt)
- **extend**: egy önmagában is teljes (tehát nem absztrakt) használati esetet opcionálisan kiegészítő másik esetre mutat

□ Használati esetek **rákövetkezési sorrendje**

- **precedes**: aktor által közvetlenül kezdeményezhető tevékenységek közötti sorrendet jelöli ki.
- **invokes**: rámutat arra, hogy egy tevékenység végrehajtását mely másik tevékenységnek kell követnie az aktor akaratától függetlenül

□ Használati esetek közötti **származtatás**

- amikor egy általánosan megfogalmazott használati esetnek több eltérő konkrét változata is beazonosítható

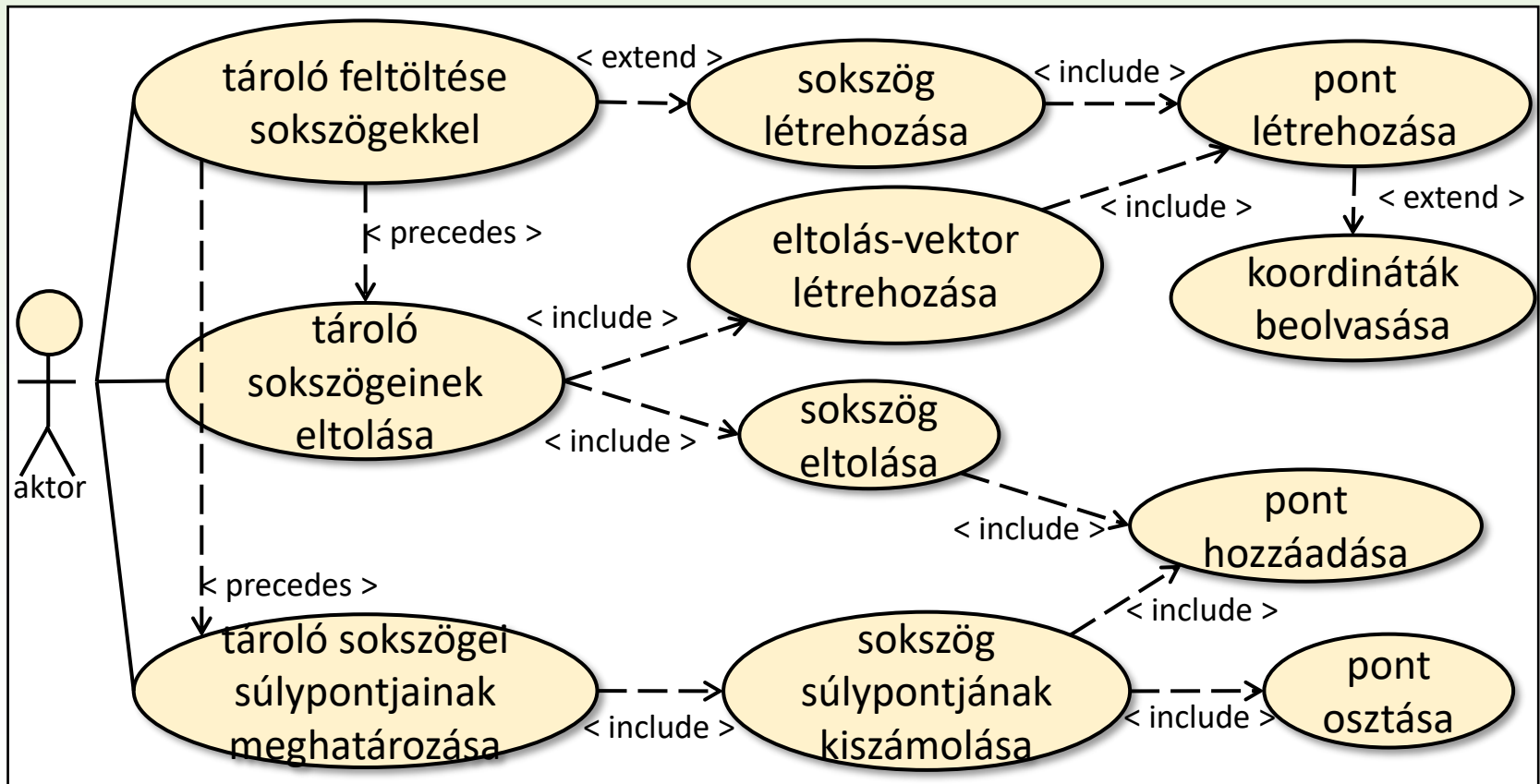
Felhasználói történet (user story)

- ❑ A használati eset diagram önmagában nem ad elégséges képet a megvalósítandó rendszer funkcionalitásáról, mivel az egyes tevékenységeknek lényegében csak a nevét adja meg.
- ❑ A felhasználói esetek tevékenységeit részletesen ki lehet fejteni egy táblázatos leírásában („user story”):
 - mi a tevékenység **neve**,
 - milyen **előfeltétel** meglétét feltételezi (GIVEN)
 - milyen **esemény** hatására következik be (WHEN)
 - mi a **hatása** a végrehajtásának, milyen eredményt ad (THEN)
 - melyik felhasználói csoport számára biztosított

AS a felhasználói csoport		
eset		leírás
tevékenység neve	GIVEN	tevékenység kiváltásakor feltételezett alaphelyzet
	WHEN	tevékenység kiváltása
	THEN	tevékenység hatása

Példa

Töltsünk fel egy tárolót különféle sokszögekkel, és mindegyiket toljuk el ugyanazon irányba és mértékkel, majd számoljuk ki az így nyert sokszögek súlypontjait. A csúcspontok és súlypontok koordinátái, sőt az eltolást leíró helyvektor végpontjának koordinátái is legyenek valós számok.



Példa

eset		Leírás
tároló feltöltése sokszögekkel	GIVEN	sokszögeket tárolni képes tároló (pl. sorozat)
	WHEN	egy sokszögnek a tárolóban történő elhelyezésének kérése
	THEN	a tároló sokszögeket tartalmaz
sokszög létrehozása	GIVEN	alkalmazás egy sokszög létrehozására vár
	WHEN	csúcsok megadása (darabszámuk és a csúcsok síkbeli pontjai)
	THEN	létrejön egy sokszög (objektum)
pont létrehozása	GIVEN	síkbeli pont koordinátái
	WHEN	síkbeli pont létrehozása
	THEN	létrejön egy síkbeli pont (objektum)
tároló sokszögeinek eltolása	GIVEN	tároló a sokszögekkel
	WHEN	sokszögek eltolása az eltolás helyvektora végpontjának megadásával
	THEN	A tároló sokszögeit egymás után eltolja a megadott vektorral
eltolás irányának és mértékének megadása	GIVEN	síkbeli pont koordinátái
	WHEN	az eltolás helyvektora létrehozása
	THEN	egy síkbeli pont létrehozása

Példa

eset		leírás
tároló sokszögei súlypontjainak meghatározása	GIVEN	tároló sokszögekkel
	WHEN	súlypontok meghatározásának kérelme
	THEN	a tároló sokszögeinek egymás után kiszámolja és kiírja a súlypontját
sokszög súlypontjának kiszámolása	GIVEN	sokszög
	WHEN	súlypont számítás igénye
	THEN	sokszög csúcspontjai alapján létrejön egy síkbeli pont (objektum)
sokszög eltolása	GIVEN	sokszög és az eltolás helyvektorának végpontja
	WHEN	eltolás igénye
	THEN	módosulnak a sokszög csúcspontjai
pont eltolása	GIVEN	egy pont és az eltolás vektor végpontja
	WHEN	eltolás igénye
	THEN	az adott pont koordinátáihoz az eltolás végpontjának koordinátáit adjuk
pont osztása	GIVEN	síkbeli pont és egy egész szám
	WHEN	osztás igénye
	THEN	a pont koordinátái az eredeti koordináták adott számmal vett hányadosai

UML

2.rész

Objektum-, kommunikációs diagram

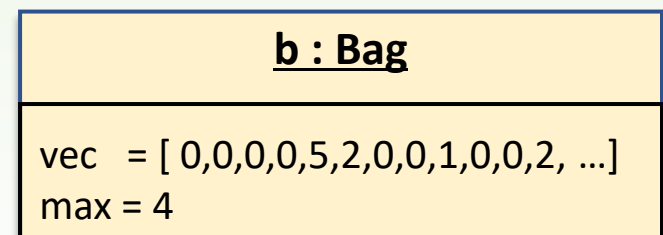
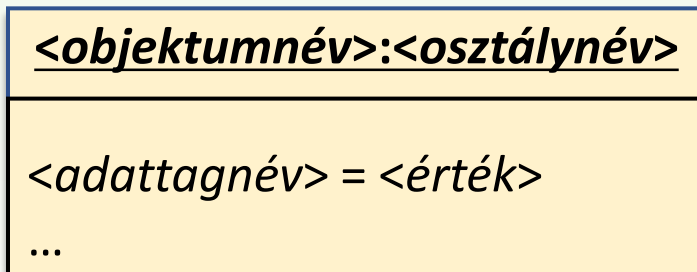
Gregorics Tibor

gt@inf.elte.hu

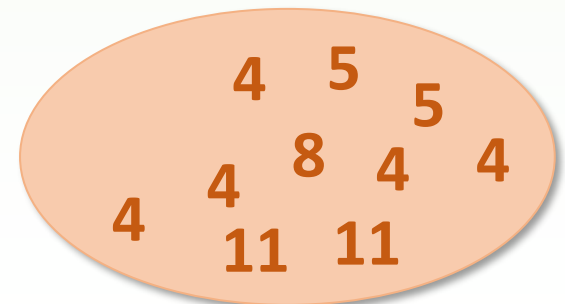
<http://people.inf.elte.hu/gt/oep>

Objektum diagram

- ❑ Az objektum diagram a tervezett rendszer működésének egy adott időpillanatában létező objektumok (a **populáció**) leírására szolgál.
- ❑ Egy objektumot meghatároz
 - az **osztálya**, amely az ugyanolyan adattagokkal és metódusokkal rendelkező objektumokat jellemzi
 - a **neve** (amit nem kötelező megadni)
 - az **állapota** (amit az adattagjainak értékei jelölnek ki)



Az objektum diagram nem tartalmazza az objektumra meghívható metódusokat.

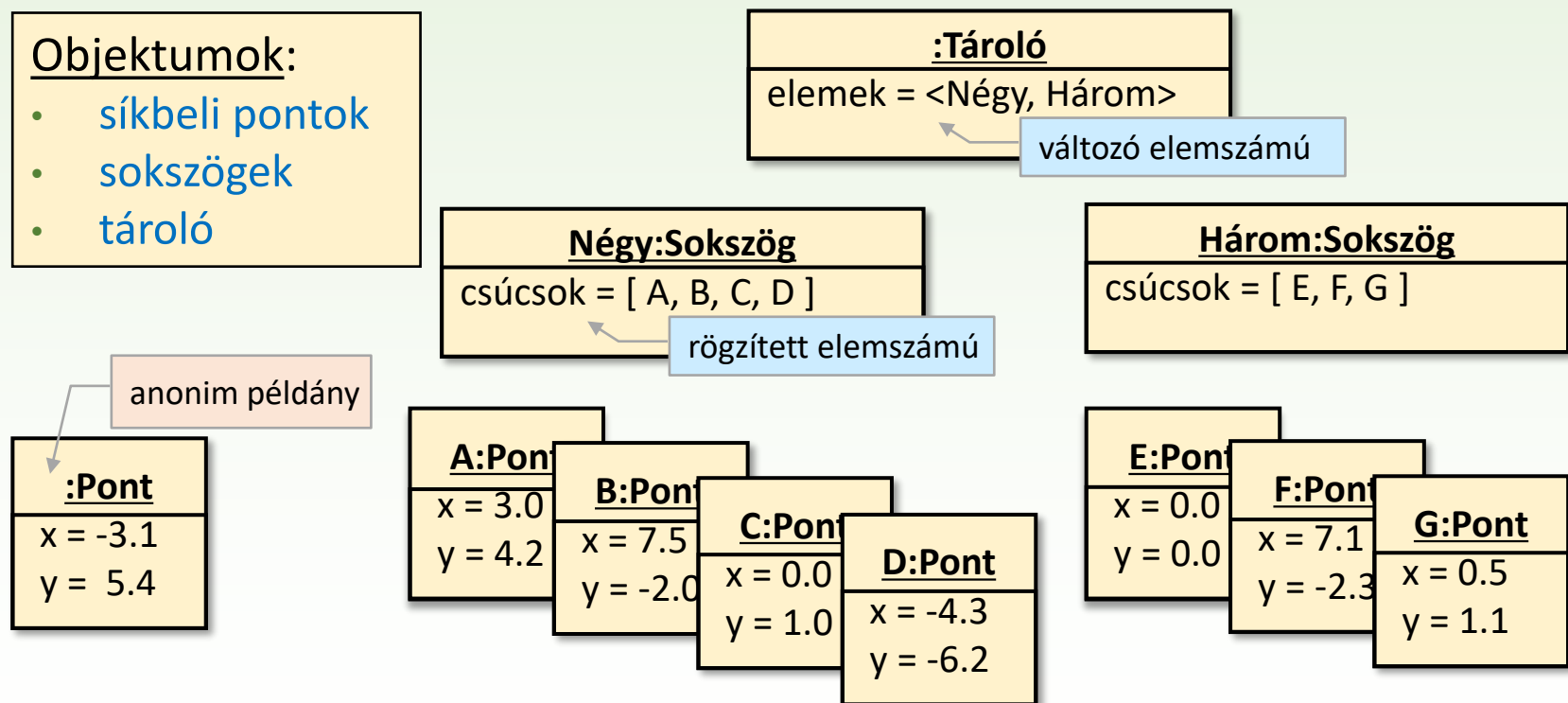


Példa

Töltsünk fel egy tárolót különféle sokszögekkel, és mindegyiket toljuk el ugyanazon irányba és mértékkel, majd számoljuk ki az így nyert sokszögek súlypontjait. A csúcspontok és súlypontok koordinátái, sőt az eltolást leíró helyvektor végpontjának koordinátái is legyenek valós számok.

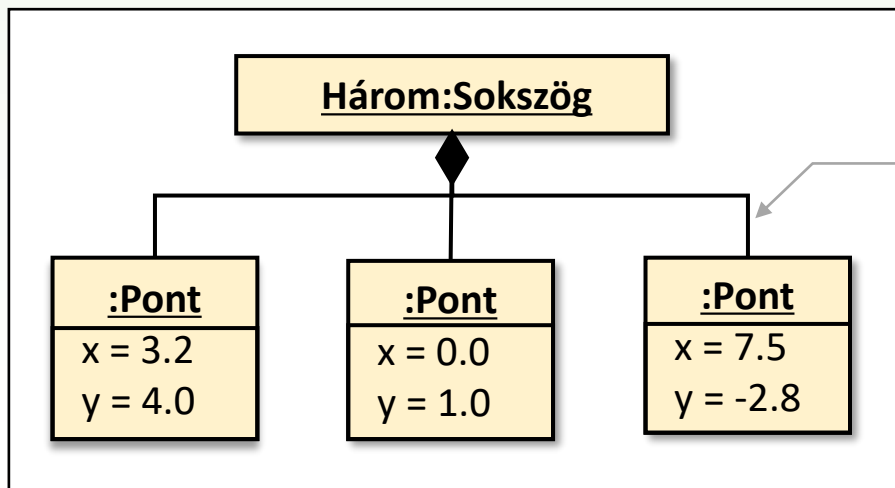
Objektumok:

- síkbeli pontok
- sokszögek
- tároló



Objektumok kapcsolatai

- Az objektumok között különböző célból jöhetnek létre kapcsolatok:
 - objektum adattagja hivatkozhat másik objektumra (például arra, amelyik az egyik részét alkotja),
 - objektum egy metódusa hívhatja egy másik objektum metódusát,
 - objektum küldhet másik objektumnak jelzést (szignált)
- Egy kapcsolatban az egyik objektumnak ismernie kell a másik objektum hivatkozását: ezt tárolhatja egy adattagjában, vagy egy metódusa megkaphatja paraméterként, esetleg maga példányosítja azt.



Folytonos összekötő vonal jelzi két objektum között fenn álló kapcsolatot. Ennek végén lehet nyíl, amely arra az objektumra mutat, amely a másiktól elérhető (látható); vagy rombusz, ha az így megjelölt objektum tartalmazza a vele kapcsolatban levő objektumokat.

Objektumok felelősségi köre

- ❑ Egy objektumnak a metódusa az objektum adattagjaival végez olyan tevékenységet, amely a feladat megoldásának egy részéért felelős.
- ❑ Amikor egy feladat megoldásához szükséges tevékenységeket metódusként az egyes objektumokhoz rendeljük, akkor ezzel kijelöljük az objektumnak a feladat megoldásában játszó felelősségi körét.
- ❑ Ügyelni kell arra, hogy ezeket a felelősségi köröket arányosan osszuk szét az objektumok között.

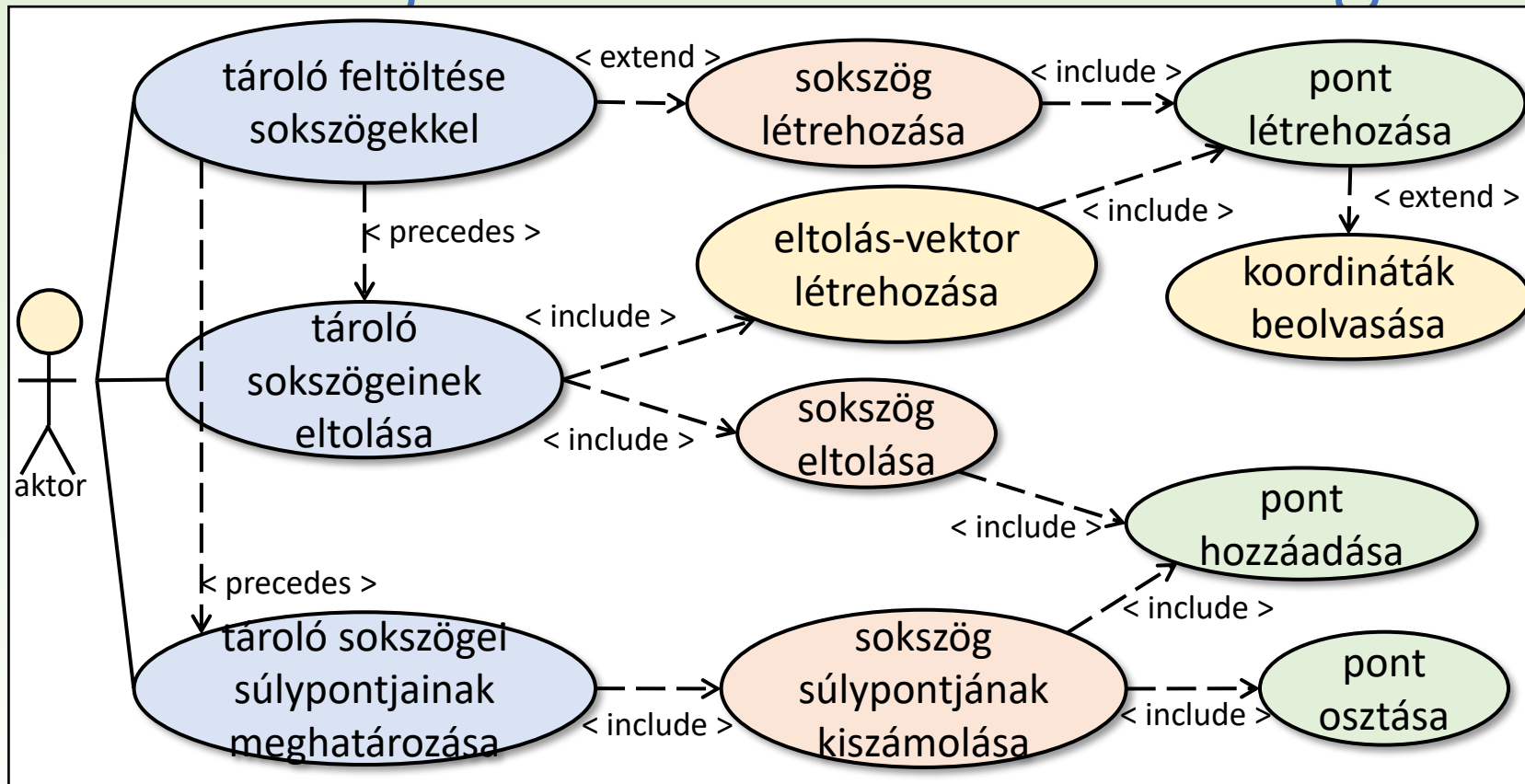
Single responsibility

O
L
I
D

Egy objektum – egy felelősségi kör

- Ne legyenek "mindenható" objektumok, amelyek több felelősségi kört érintő metódusokkal is rendelkeznek.
- Ne osztozzon több objektum ugyanazon a felelősségi körön.

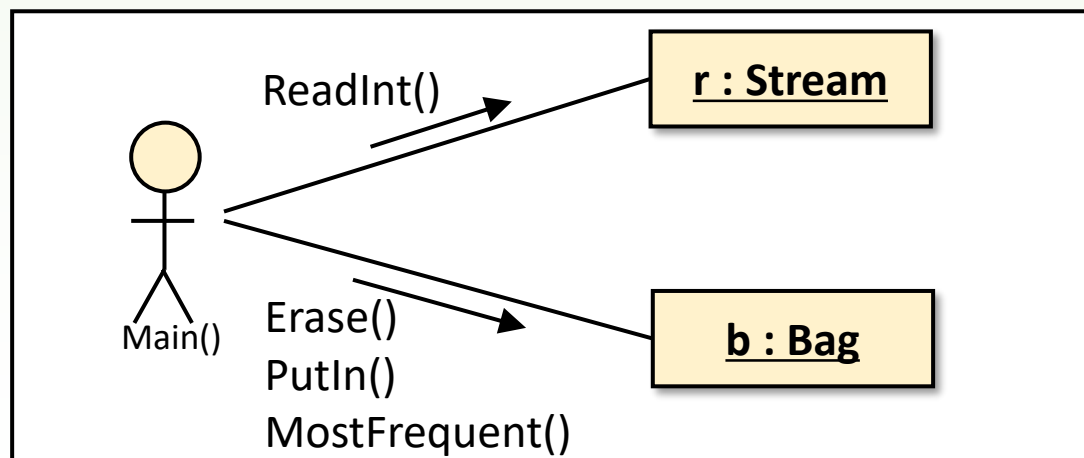
Példa objektumainak felelőssége



- **síkbeli pont:** létrehozás, eltolás, osztás
- **sokszög:** létrehozás, eltolás, súlypont kiszámítás
- **tároló (sorozat):** elem elhelyezése (hozzáfűzése), elemek kinyerése felsorolásukkal)

Kommunikációs diagram

- ❑ Ez a diagram bár hasonlít az objektum diagramhoz, de nem tünteti fel az objektumok adattagjainak értékét, viszont megmutatja, hogy a kapcsolatban álló **objektumok** milyen **üzeneteket** válthatnak egymással.
- ❑ Egy üzenet lehet egy **metódus meghívása** vagy egy **szignál elküldése**.
- ❑ Egy kapcsolat mellé rajzolt **nyíl** a **küldő** objektum felől a **fogadó** objektum felé mutat (egy kapcsolat lehet két irányú is), és azokat a metódusokat/szignálokat írjuk a nyílra, amelyeket a fogadó objektum birtokol/kezel. Így a diagram lényegében kijelöli egy-egy objektum felelősségi körét, ezen belül a metódusait, és ezzel pótolja az objektum diagram ebbéli hiányosságát.

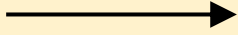
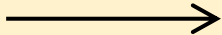


Kommunikációs diagram üzenetei

□ Üzenet az, amikor

- a küldő objektum meghívja a fogadó objektum egyik **metódusát**,
- a küldő objektum **szignált** küld a fogadó objektumnak.

□ Az üzenetküldés módja lehet

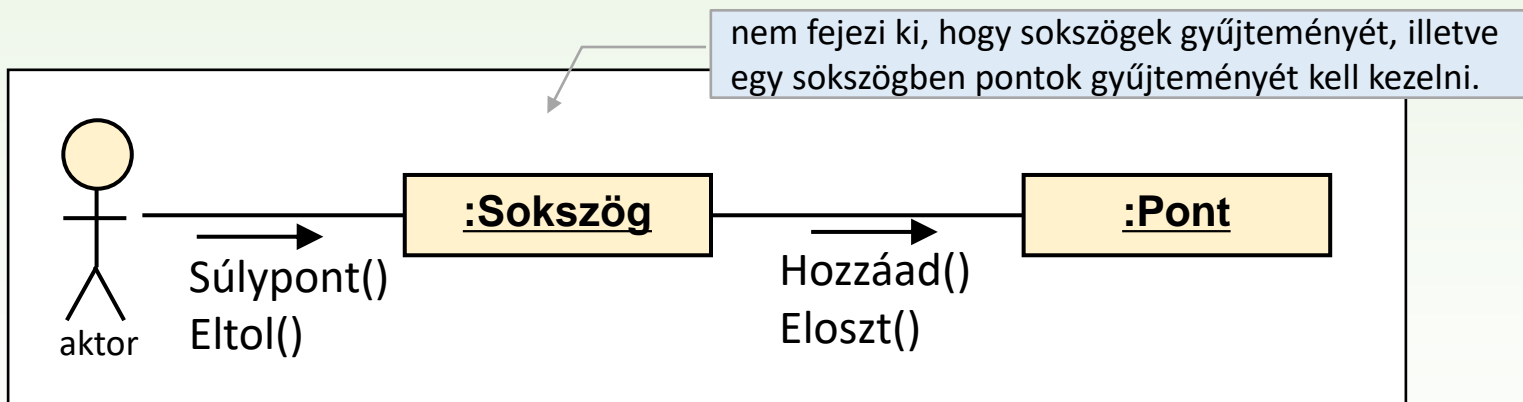
- **szinkron**: a küldő objektum megvárja, amíg a fogadó feldolgozza az üzenetét, és csak ezen feldolgozás esetleges eredményét megkapva folytatja a saját tevékenységét, 
- **aszinkron**: az üzenet elküldése után a küldő és a fogadó objektumok tevékenységei párhuzamosan folynak. 

□ Egyéb jelölések

- Kijelölhető az üzenetek **sorrendje** (azok sorszámozásával).
- Megadható az üzenetek **előfeltétele** (szögletes zárójelpár között).
- Jelölhető (csillaggal), ha egy üzenet ciklikusan ismétlődik.

Példa

Töltsünk fel egy tárolót különféle sokszögekkel, és mindegyiket toljuk el ugyanazon irányba és mértékkel, majd számoljuk ki az így nyert sokszögek súlypontjait. A csúcspontok és súlypontok koordinátái, sőt az eltolást leíró helyvektor végpontjának koordinátái is legyenek egész számok.



UML

3.rész

Osztály diagram

Gregorics Tibor

gt@inf.elte.hu

<http://people.inf.elte.hu/gt/oep>

Osztály diagram

- ❑ Egyetlen osztály az azonos típusú objektumok adattagjait és metódusait képes általánosan definiálni.
- ❑ Különböző típusú objektumok leírásához több osztályt tartalmazó diagramot kell készíteni, amely nemcsak a tervezett rendszer működése során létrejövő objektum-populációk általános leírására szolgál, hanem az objektumok között létrejövő kapcsolatokat is jellemezni tudja.
- ❑ Egy osztály diagramnak számos objektum diagramot lehet megfeleltetni, más szavakkal, az osztály diagram az objektum diagramok absztrakciója.
- ❑ Az osztály diagram elsősorban egy szoftver tervezését támogató eszköz, de már a szoftver elemzésénél is használható, valamint a megvalósítás leírására is alkalmas.

Osztály leírás elemei

- Egy osztálynak a leírása a belőle példányosítható objektumokat jellemzi. Tartalmazza az
 - **osztálynevet**, ami az objektumok típusának neve is egyben
 - **adattagok** (tulajdonság, attribútum, mező) felsorolását a nevükkel, típusukkal, és egyéb jellemzőikkel
 - **metódusok** (művelet, tagfüggvény) felsorolását a nevükkel, paraméterlistájukkal, visszatérési típusukkal, és egyéb jellemzőikkel
- Az adattagok és metódusok egyenként beállítható láthatósága azt jelzi, hogy egy tagra hivatkozhatunk-e az osztály-leíráson kívül vagy sem.
 - kívülről is látható, azaz publikus (*public +*)
 - külvilág elől rejtett: privát (*private -*) vagy védett (*protected #*)

<osztálynév>
<+ - #> <adattagnév> : <típus>
...
<+ - #> <metódusnév>(<paraméterek>) : <típus>
...

Osztály diagram részletezettsége

- Az osztály diagram **a modellezés során alakul ki**. Kezdetben hiányos, és csak fokozatosan, több lépésben bővül, módosul úgy, hogy tartalmazza a megvalósításhoz szükséges információt.
 - először csak a főbb osztályok neve születik meg, amit a felelősségi köröket lefedő metódusok neve követ
 - majd felkerülnek az adattagok (névvel és típussal)
 - később döntünk a tagok láthatóságáról
 - utána pontosítjuk a metódusok szignatúráját (paraméterek neve, típusa, visszatérési típus), és megadjuk a törzsüket
 - végül megjelennek a rejtett láthatóságú segédmetódusok

<osztálynév>

<osztálynév>

<metódusnév>()

<osztálynév>

- <adattagnév> : <típusnév>

+ <metódusnév>(<paraméterek>) : <típusnév>

Példa

Töltsünk fel egy tárolót különféle sokszögekkel, és mindegyiket toljuk el ugyanazon irányba és mértékkel, majd számoljuk ki az így nyert sokszögek súlypontjait. A csúcspontok és súlypontok koordinátái, sőt az eltolást leíró helyvektor végpontjának koordinátái is legyenek egész számok.

Négy:Sokszög
csúcscok = [A, B, C, D]

Három:Sokszög
csúcscok = [E, F, G]

Sokszög
csúcscok : Pont[]
Súlypont() Eltol()

A:Pont x = 3.0 y = 4.2	B:Pont x = 7.5 y = -2.0	C:Pont x = 0.0 y = 1.0	D:Pont x = -4.3 y = -6.2
-------------------------------------	--------------------------------------	-------------------------------------	---------------------------------------

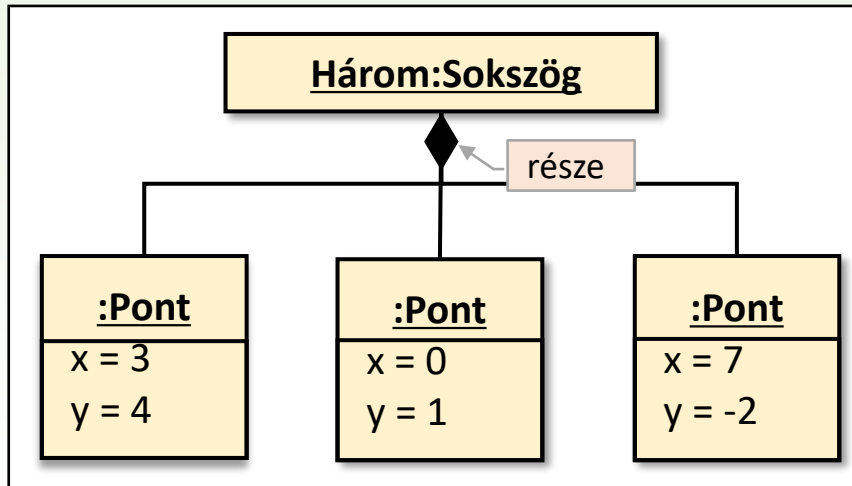
E:Pont x = 0.0 y = 0.0	F:Pont x = 7.1 y = -2.3	G:Pont x = 0.5 y = 1.1
-------------------------------------	--------------------------------------	-------------------------------------

Pont
x : real y : real
Hozzáad() Eloszt()

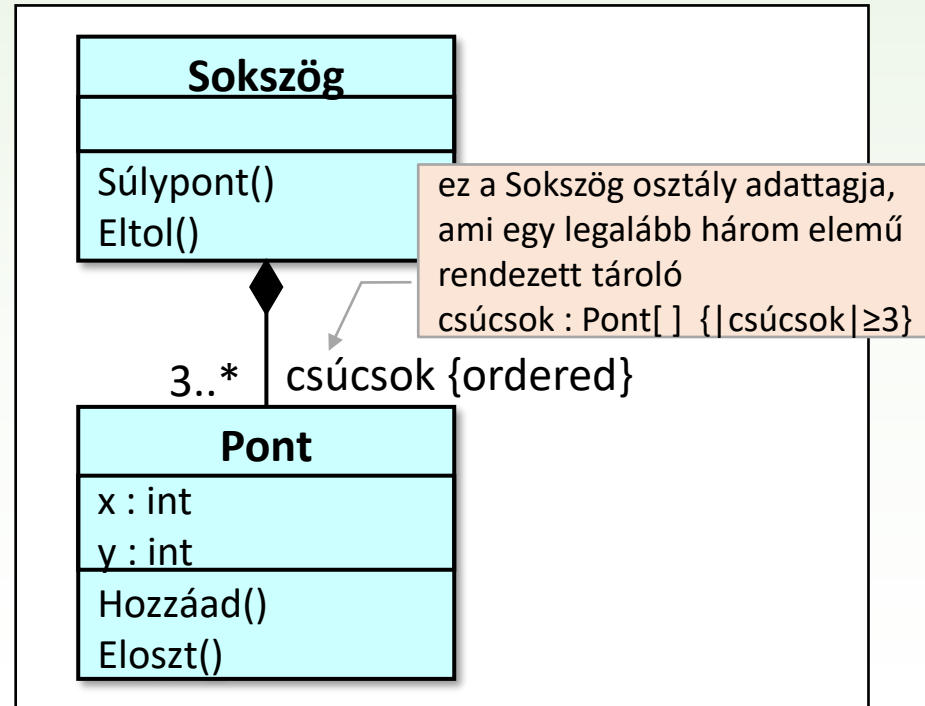
Osztály diagramban ábrázolt objektum-kapcsolatok

- Az objektumok közötti kapcsolatok osztályszintű leírását a következő előadásban fogjuk részletesen tárgyalni.

Objektumok közötti kapcsolatok ábrázolása objektum diagramban:

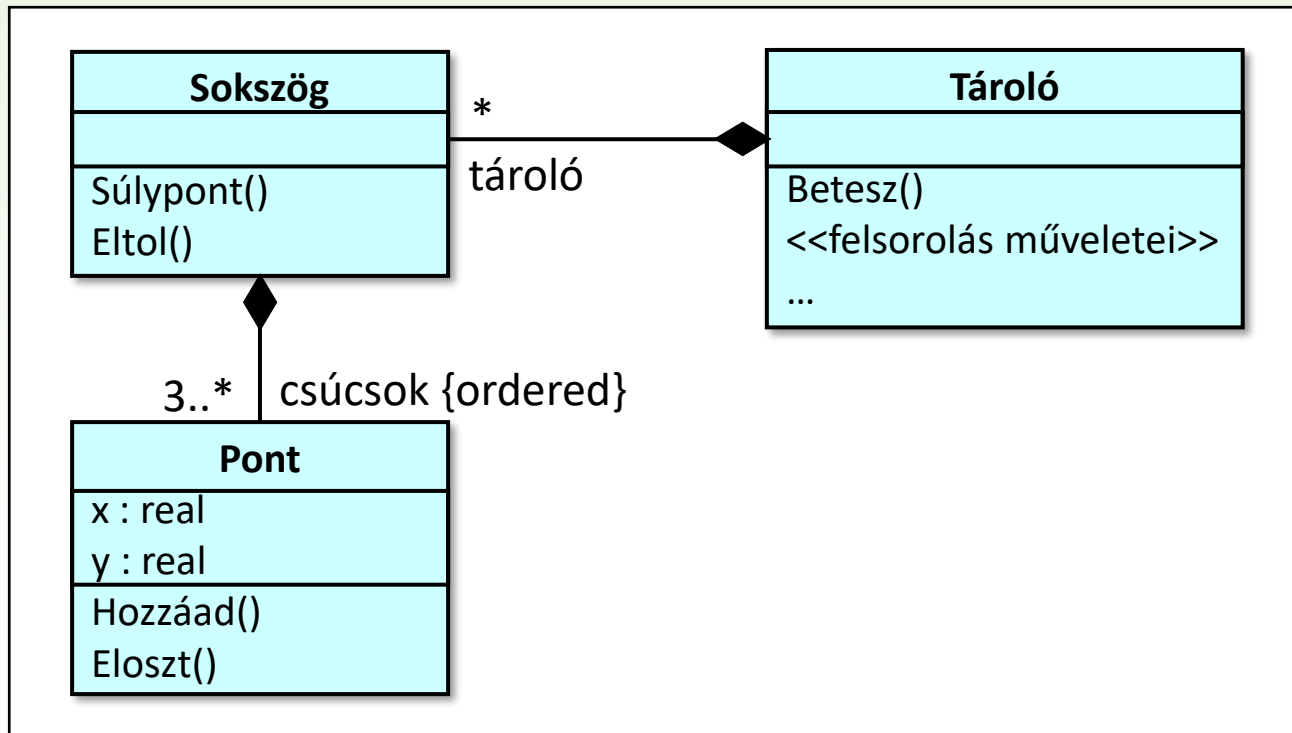


Objektumok közötti kapcsolatok ábrázolása osztály diagramban:



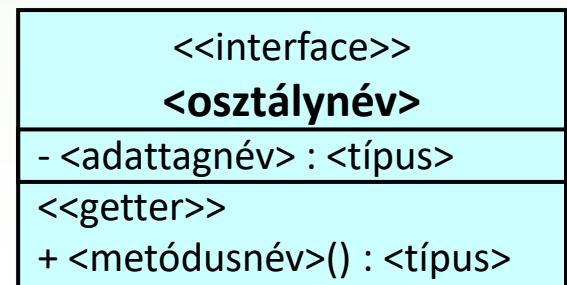
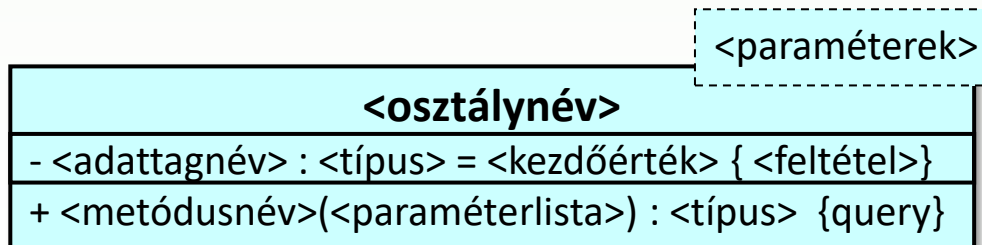
Példa

Töltsünk fel egy tárolót különféle sokszögekkel, és mindegyiket toljuk el ugyanazon irányba és mértékkel, majd számoljuk ki az így nyert sokszögek súlypontjait. A csúcspontok és súlypontok koordinátái, sőt az eltolást leíró helyvektor végpontjának koordinátái is legyenek egész számok.



Osztály diagram kiegészítései

- ❑ Az osztályok a példányaik jellegzetes viselkedése alapján különféle **kategóriákba** (sztereotípiákba) sorolhatók, amelyet <<...>> jelzés között írhatunk le (pl. <<interface>> , <<enumeration>>, <<singleton>>).
- ❑ Egy osztályt általánosítva is felírhatunk (generikus-, sablon-osztály), ha (pl. típust helyettesítő) **paramétere**ket vezetünk be.
- ❑ Az adattagokhoz (a '=' jel után) **kezdőértéket** rendelhetünk, amiket a konstruktor állít majd be, illetve {...} jelzésben **megszorításokat** (típus invariáns) is adhatunk rájuk.
- ❑ A metódusok sajátos tulajdonságai (pl. query, virtual, override) is megadható {...} jelzés között. A <<getter>> az adattagok értékét **lekérdező** metódusok csoportját fejlécezi, a <<setter>> az adattagok értékét **felülíró műveletek** csoportját vezeti be.



UML

4.rész

Szekvencia diagram

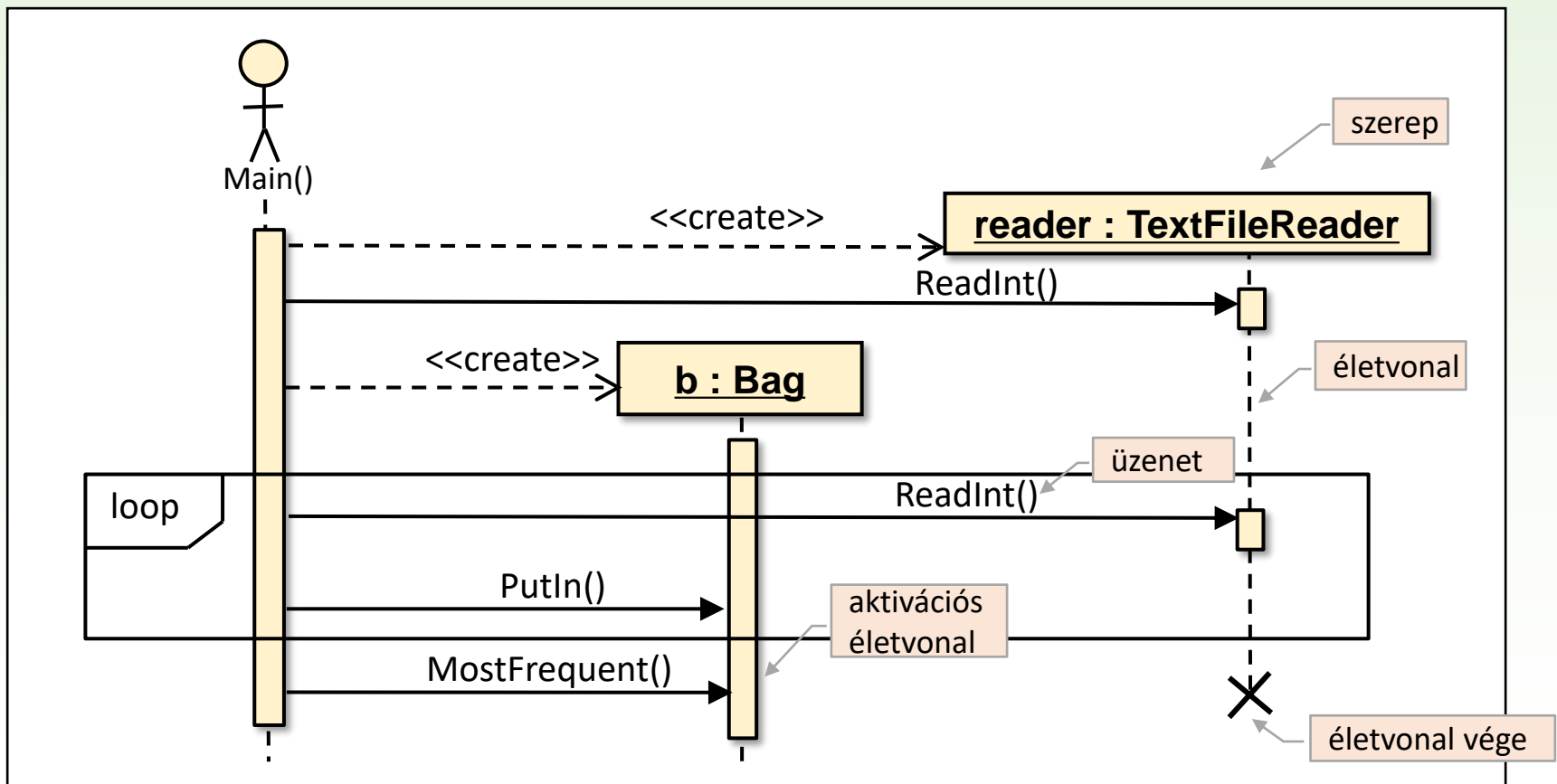
Gregorics Tibor

gt@inf.elte.hu

<http://people.inf.elte.hu/gt/oep>

Szekvencia diagram

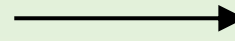
- Az objektumok közötti **üzenetváltások időbeli sorrendjét** mutatja meg.
- Ez a diagram a rendszer vagy a rendszer egy részének egy lehetséges működését írja le, ezért több is szokott készülni belőle. Nemcsak az elemzés, hanem a tesztelés dokumentálásához is használják.



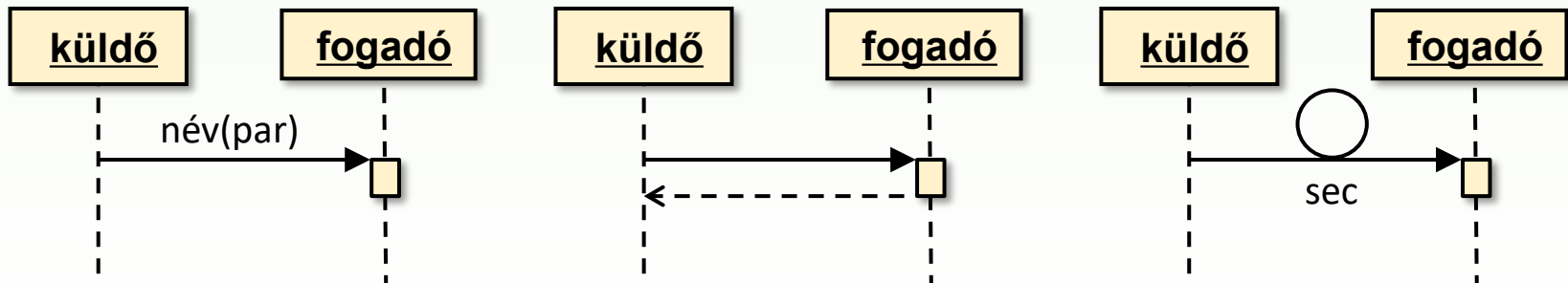
Szekvencia diagram elemei

- ❑ **Osztályszerep** : egy vagy akár több osztály egy vagy több objektumát testesíti meg, sokszor csak egyetlen objektumot.
- ❑ **Osztályszerep életvonal** (szerepből kiinduló függőleges szaggatott vonal) : egy osztályszerep létezését ábrázoló idővonal.
- ❑ **Osztályszerep aktivációs életvonal** (egy függőlegesen elnyújtott téglalap az életvonalon) : egy osztályszerepnek azon időszaka, amelyben az osztályszerepet megtestesítő objektumok más objektumok vezérlése alatt állnak, azaz egy vagy több metódusuk végrehajtás alatt áll.
- ❑ **Üzenet** (általában vízszintes nyíl az életvonalak között) : az objektumok közötti információ átadás formája. Megmutatja, hogy egy objektum egy üzenet elküldésével mikor hoz működésbe (akció) egy másik objektumot. (pl. meghívja annak egy metódusát.)
 - Az egymás alatt jelzett üzenetek azok időbeli sorrendjét mutatja.
 - Feltüntethető két üzenet között eltelt idő is.

Szinkron üzenetek



- ❑ Szinkron üzenetről akkor beszélünk, amikor a küldő szerep átadja a vezérlést a fogadó szerepnek, és a saját tevékenységét mindaddig blokkolja, amíg a fogadó ezt nem oldja fel. Ez lehet
 - a fogadó egy **metódusának szinkron hívása** vagy részére egy **szignál szinkron elküldése** (ez ritka). Az üzenet címkéje a meghívott metódus/szignál neve az esetleges paraméterekkel.
 - egy **szinkronizációs üzenet**, amely során a küldő szerep várakozik a fogadó szerep visszajelzésére.
 - **időhöz kötött várakozó üzenet**, amikor a küldő szerep megjelölt ideig várakozik a fogadó szerep visszajelzésére.



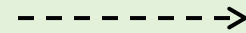
Aszinkron üzenetek

—————> régebben: —————>

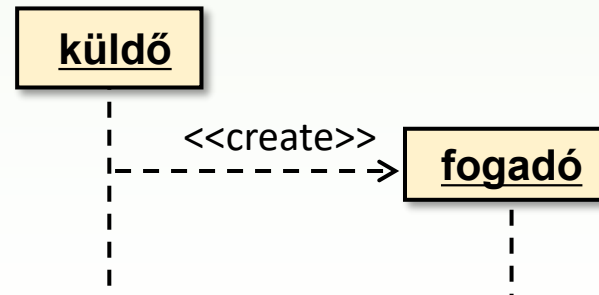
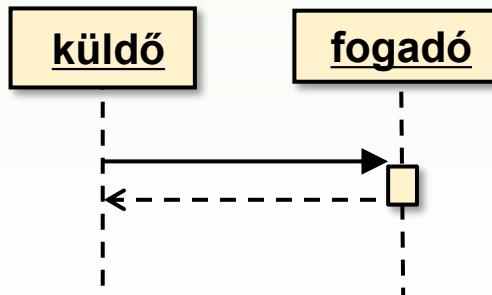
- ❑ Aszinkron üzenetről akkor beszélünk, amikor a küldő szerep az üzenet elküldése után nem szakítja meg a saját tevékenységét, nem várakozik a fogadó szerep visszajelzésére. Ez lehet
 - **aszinkron metódus-hívás**, vagy **aszinkron küldött szignál**, amely után a küldő és a fogadó szerep párhuzamosan tevékenykedik. Az üzenet címkéje a meghívott metódus/szignál neve a paraméterekkel.
 - **randevú üzenet**, amely során a küldő szerep megjelölt ideig várakozik arra, hogy a fogadó szerep fogadja az üzenetét.



Speciális üzenetek



- Ezek azok az üzenetek, amelyeket gyakran fel sem tüntetünk a könnyebb áttekinthetőség érdekében:
 - **visszatérési üzenet**, amelyet a fogadó szerep küld el az őt aktiváló szerepnek egy korábban attól kapott szinkron üzenetre válaszként, hogy megszüntesse annak blokkolását. Egy szinkron metódushívás esetében ez magától értetődő (ezért sokszor elhagyjuk), de a szinkronizációs üzenetre adott választ mindenképpen jelölni kell.
 - **példányosító (create) üzenet**: létrehoz egy új osztályszerepet

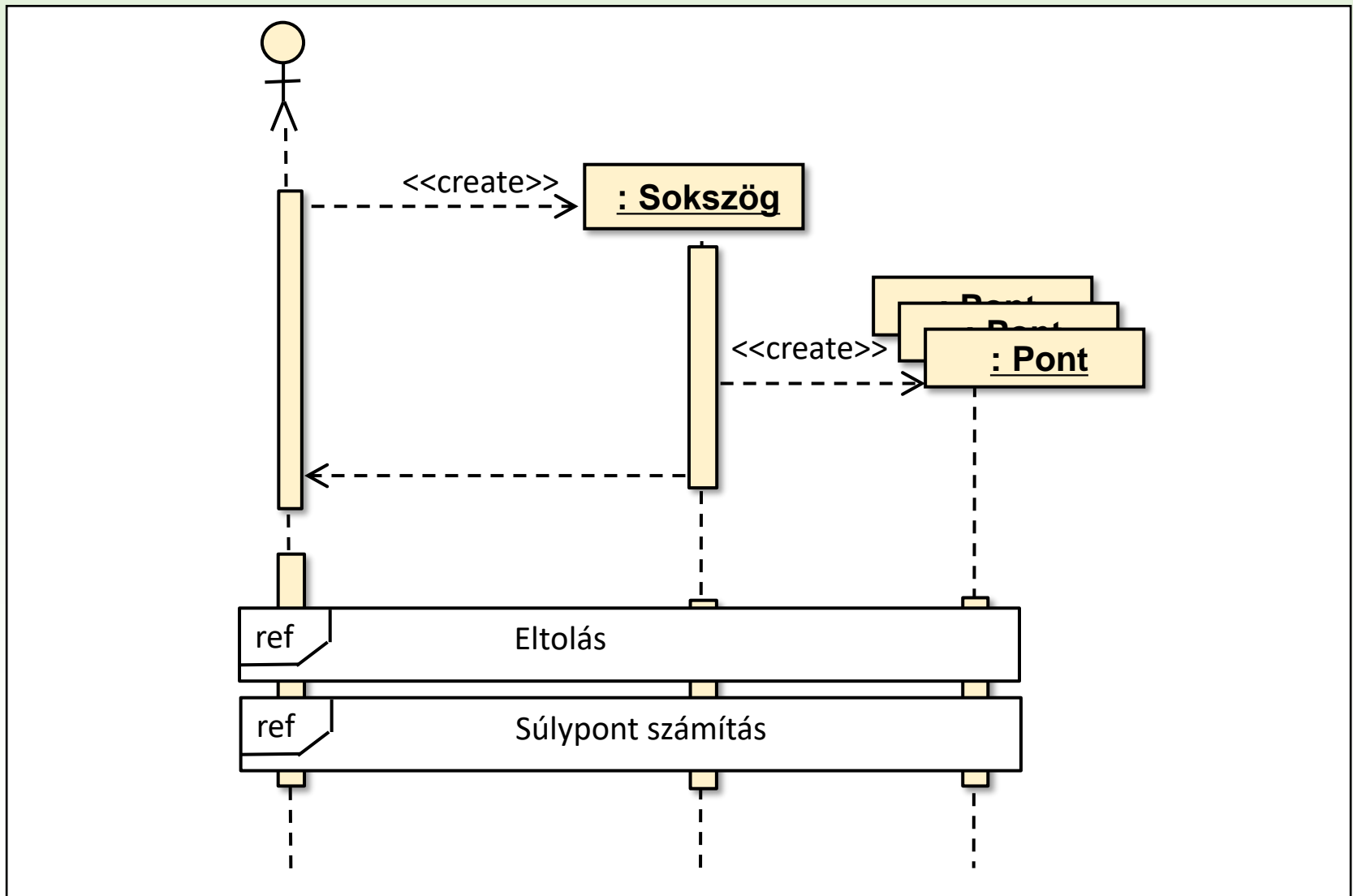


Szekvencia diagramok hierarchiája

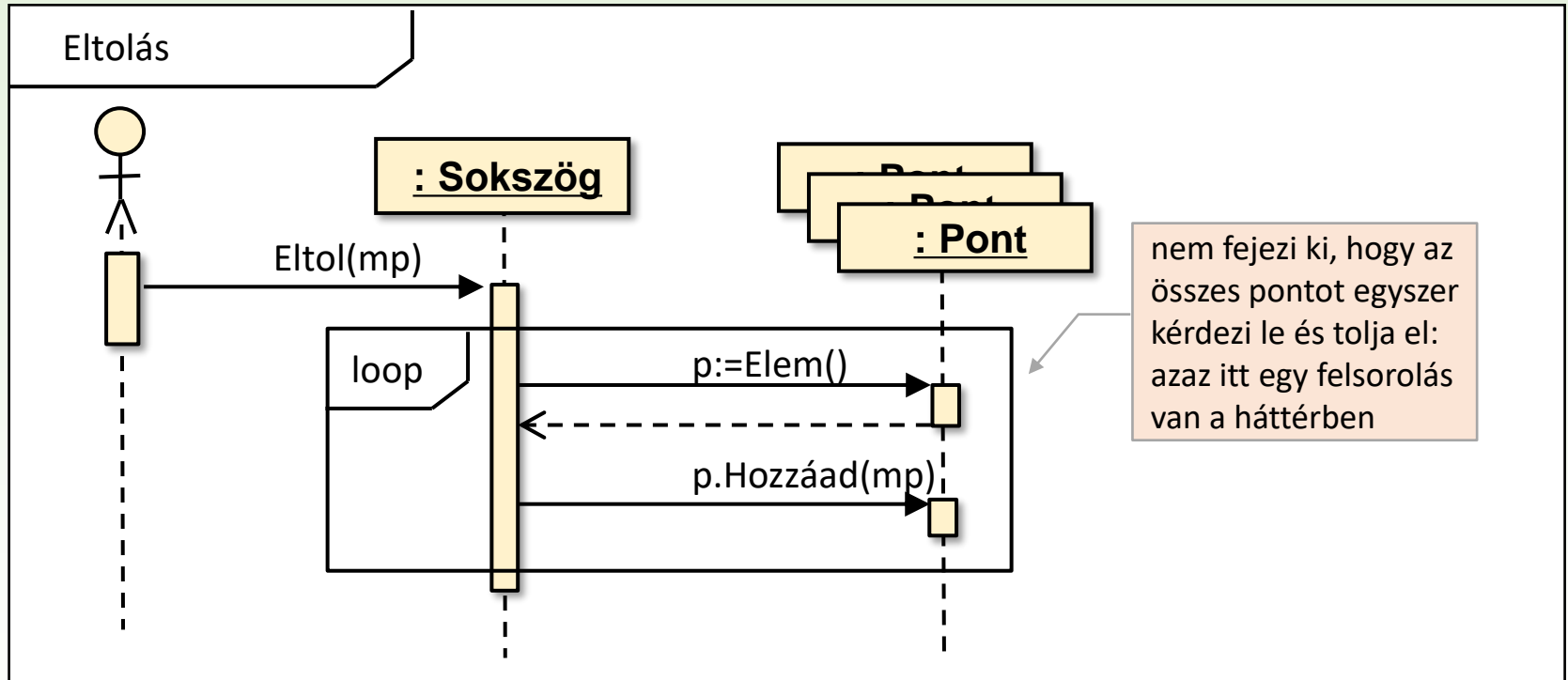
Lehetőség van arra, hogy egy összetett szekvencia diagram bizonyos részeit az olvashatóság érdekében ideiglenesen elrejtjük, és ezeket majd külön részletezzük, finomítsuk.

- ❑ Egy szekvencia diagram egyetlen osztályszerepe elfedheti például a szerepet alkotó objektumok, objektum-csoportok közötti üzenetváltásokat.
- ❑ Kiemelhető a diagram egy adott időintervallumának egy része is (ref).
- ❑ Kijelölhetünk olyan részleteket (fragmentek), amelyek
 - ciklikusan ismétlődnek (loop)
 - adott feltétel esetén következnek csak be (alt, opt)
 - párhuzamosan futnak (par)
 - ...

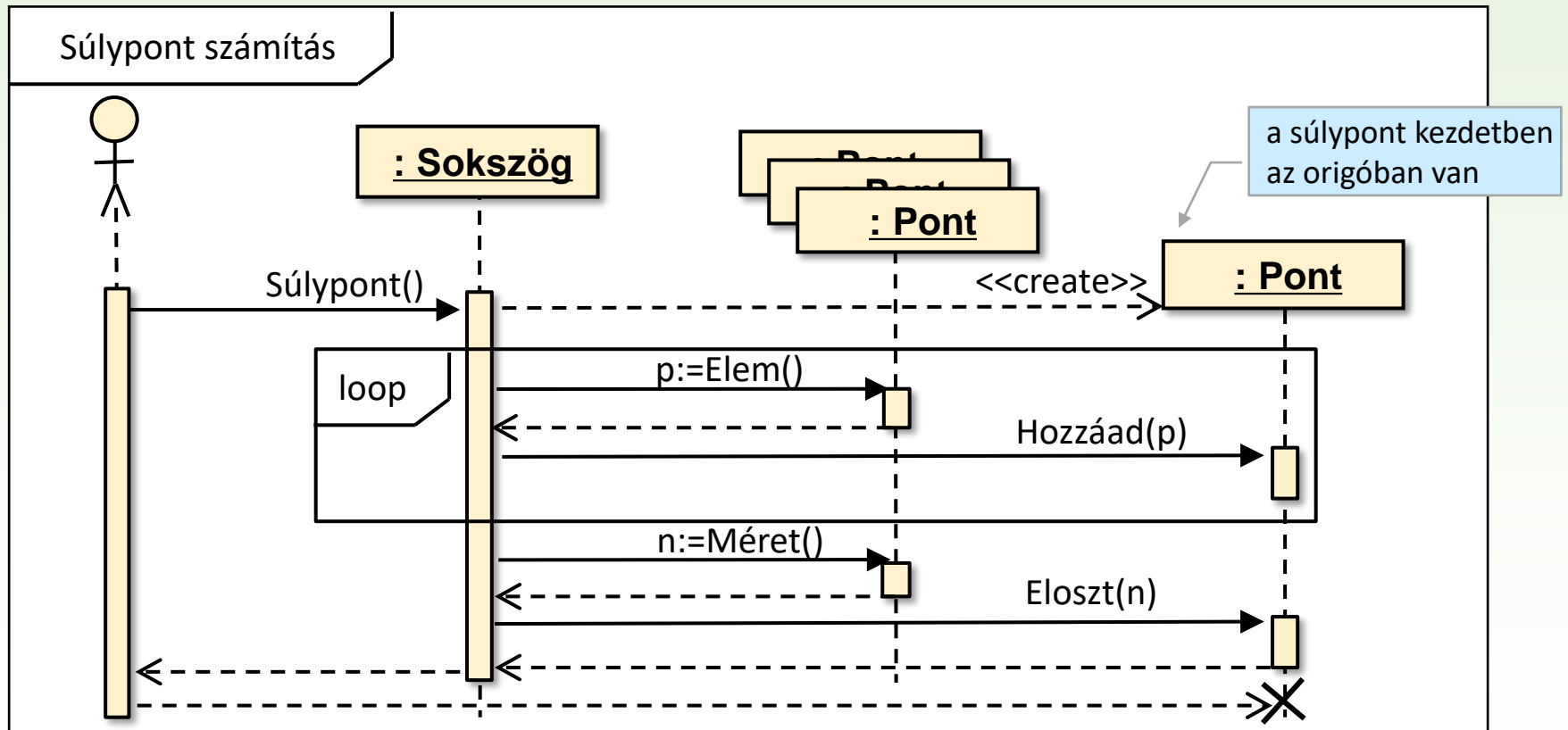
Példa



Példa



Példa



UML

5.rész

Állapotgép diagram

Gregorics Tibor

gt@inf.elte.hu

<http://people.inf.elte.hu/gt/oep>

Állapotgép diagram

- ❑ Egy objektumnak különböző állapotai lehetnek. Állapotnak tekinthető
 - az objektum adatai által felvett értékek együttese (**fizikai állapot**)
 - közös tulajdonságú fizikai állapotok halmaza (**logikai állapot**)
- ❑ Az objektum állapotai közül egyszerre egy lehet **aktív**, amely egy **esemény** hatására változhat meg: **állapot-átmenet**.
- ❑ Esemény lehet az objektumnak küldött **üzenet** (az objektum metódusának hívása vagy az objektumnak küldött szignál), vagy egy feltétel bekövetkezése.
- ❑ Az objektum állapotának változásait egy **irányított gráffal** ábrázolhatjuk, amelyben
 - a csúcsok jelölik az objektum állapotait,
 - az irányított élek mutatják az állapotok közötti átmeneteket, az élek címkéi pedig az átmenetet kiváltó eseményt.
- ❑ **Ezt a diagramot később ennél sokkal részletesen tárgyaljuk majd.**

Felsoroló állapotgépe

- ❑ Egy felsoroló objektumnak három állapotát különböztetjük meg: *indulásra kész, folyamatban van, befejeződött.*
- ❑ A felsorolás műveletei csak bizonyos állapotokban értelmesek, máskor a hatásuk nem definiált.

