4. Analízis modell II.

70 – Edgerunners

Konzulens:
Szabó Bence Sándor

Csapattagok

Csabuda Nóra	GHDCTP	csabudan@gmail.com
Gulybán Dániel	HUCHEX	gulyban.daniel@gmail.com
Nagy Gergely	HC1QJP	nagyg2002@gmail.com
Papp Levente	HV0CRH	papp.levente2003@gmail.com
Sánta Dániel	OGLSI2	daniel.santa0615@gmail.com

2024-03-11

4. Analízis modell kidolgozása

4.1 Objektum katalógus

4.1.1 Hallgató

A játékosok által irányított entitás. Képes átlépni más szobába, ha vezet oda ajtó. Képes tárgyakat magánál tartani és azokat felhasználni védekezésre, teleportálásra vagy éppen egy szoba tulajdonságának megváltoztatására. Célja a logarléc megszerzése.

4.1.2 Oktató

Nem a játékosok által irányított entitás. Célja, hogy elvegye a hallgató lelkét és kibuktassa őt az egyetemről. Ezért, ha az oktató egy szobába kerül egy, de akár több hallgatóval is, akkor megtámadja őket. Képes átlépni más szobába, ha vezet oda ajtó. Képes felvenni tárgyakat, bár azokat nem tudja használni. Az oktató megbénulhat, ha olyan szobába kerül, ahol letörölték a táblát.

4.1.3 Szoba

Ezek azok a felfedezhető entitások, amelyekből felépül a pálya. Elszórva tárgyak találhatóak meg bennük. Olykor található közöttük mérgező, elátkozott vagy olyan is, amelynek le van törölve a táblája. Ezek a tulajdonságok kombinálhatóak is. Megeshet, hogy egyesülnek vagy éppen osztódnak. Ha egy mérgező tulajdonságú szoba egyesül bármilyen más tulajdonságú szobával, akkor az újonnan létrejövő szoba is mérgező lesz. A mérgező szobába lépő entitások eldobják a tárhelyükben található tárgyakat. Az elátkozott tulajdonságú szobák ajtajai eltűnhetnek, viszont mindig lesz legalább egy kivezető út a szobából.

4.1.4 FFP2 maszk

A játékban található tárgyak egyike. Mind a hallgató, mind az oktató fel tudja venni a tárhelyébe, azonban csak a hallgatót védi meg a mérgező szobáktól. Használata a következő: a hallgató belép egy mérgező szobába, és a tárhelyében lévő maszkkal megvédheti magát a gázoktól.

4.1.5 Denevérbőrre nyomtatott TVSZ

Szintén a játékban fellelhető tárgyak egyike. Ha a hallgató tárhelyében megtalálható, akkor védelmet nyújt az oktatók támadásai ellen három alkalommal. Ennek használata passzív jellegű, ha egy hallgató találkozik egy oktatóval a TVSZ mindenképp elhasználódik védelmének érdekében.

4.1.6 Dobozolt káposztás camembert

A játékban megtalálható tárgy, amely felhasználásával egy szoba mérgező tulajdonságot kap. Az elhasználása után a tárgyat tartalmazó szoba mérgező tulajdonságúvá válik.

4.1.7 Nedves táblatörlő rongy

A hallgató a nedves táblatörlő rongy felhasználásával letörli egy szoba tábláját. Az ilyen szobákba való belépéskor az oktatók megbénulnak, ami annyit jelent, hogy a következő lépésükből kimaradnak.

4.1.8 Szent söröspohár

A hallgató el tudja fogyasztani, azaz fel tudja használni. Felhasználást követő három körben a hallgató védett az oktatók támadásai ellen.

4.1.9 Tranzisztor

Ha egy hallgatónál van két ilyen tárgy, akkor azokat össze tudja kapcsolni. Összekapcsolást követően, ha felhasználja az egyiket, akkor az azt jelenti, hogy aktiválja és leteszi a szobában, ahol van. Ekkor az imént letett tranzisztort már nem tudja felvenni a másik tranzisztor felhasználásáig. Ezt követően fel tudja használni a másik tranzisztort is, ekkor azonban az a földre kerül, és a hallgató átkerül az először felhasznált tranzisztor szobájába. Ekkor a két tranzisztor szétkapcsol és mindkettő újra felvehető.

4.1.10 Logarléc

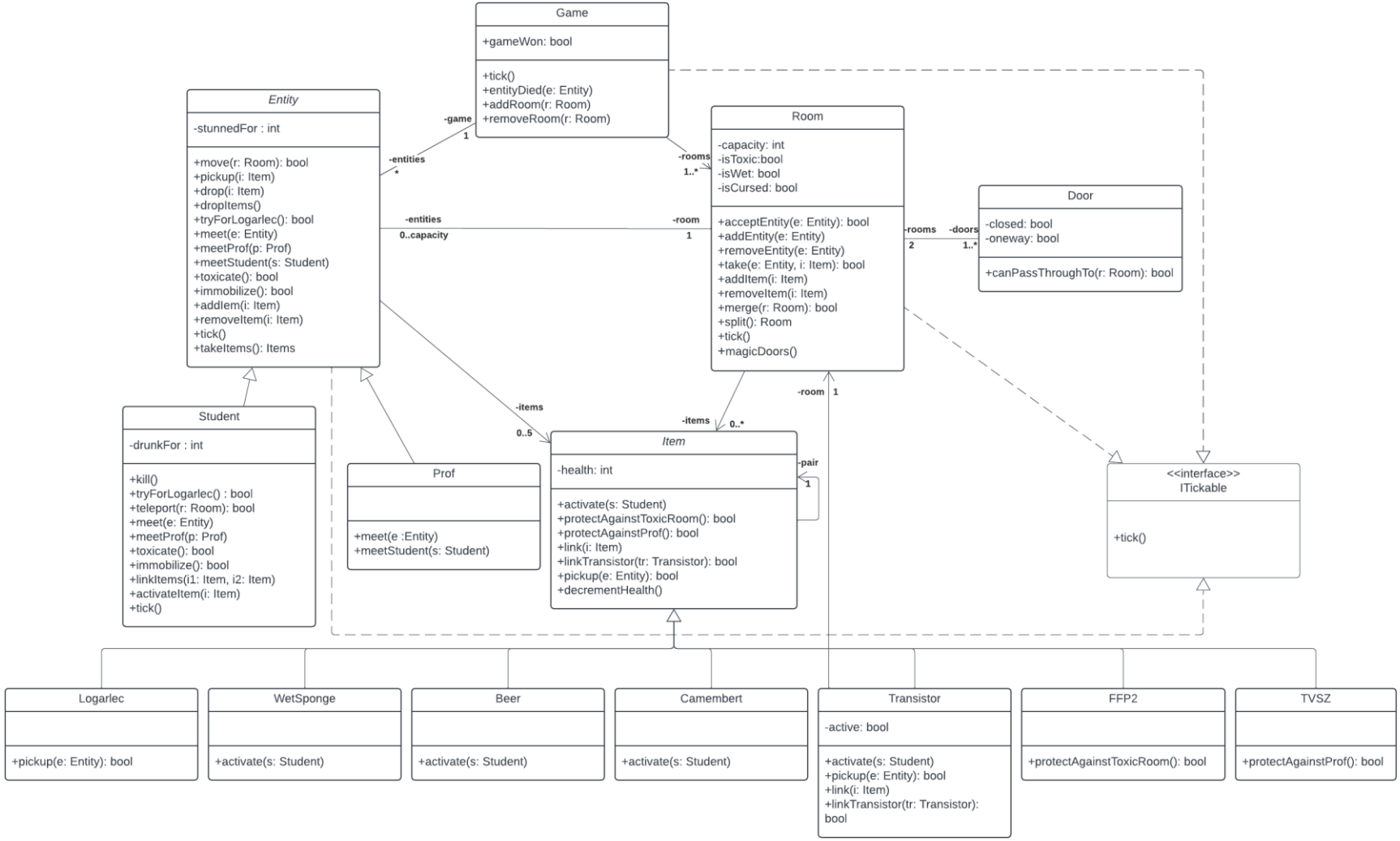
Ez a tárgy az oktatókat hidegen hagyja, ezért nem is veszik fel. A hallgató célja azonban pont az, hogy megszerezze ezen mágikus képességű tárgyat. Ha ez megtörtént, akkor a hallgatók megnyerik a játékot.

4.1.11 Game

Ez az objektum gyakorlatilag a játéktábla. Tartalmazza a szobákat és entitásokat. Időegységenként véletlenszerűen egybeolvaszt és szétoszt szobákat, valamint jelzi az entitásoknak és szobáknak, hogy eltelt egy időegység. Lekezeli a hallgatók, entitások kiesését a játékból és neki tudjuk jelezni azt is, ha megszereztük a mágikus képességű logarlécet.

4.2 Statikus struktúra diagram

Az osztályok tartalmazzák a privát adattagok getter-setter metódusaikat is, amelyeket ezen dokumentum nem jelöl.



4.3 Osztályok leírása

4.3.1 Beer

- **Felelősség**

A sör egy aktiválással hasznosítható tárgy. Aktiválásával a hallgató védettségre tehet szert, (részeggé válással) az oktatók támadásával szemben.

- **Ősosztályok**

Item

- **Interfészek**

Nincs.

- **Asszociációk**

Ősosztályban található.

- **Attribútumok**

Ősosztályban található.

- **Metódusok**
  - **void activate(Student s)**: **[override]** Aktiválja a tárgyat, és a tárgy health-ét megfelelő értékre állítja. Meghívja a hallgató védetté („részeggé”) válásához szükséges függvényeket.

4.3.2 Camembert

- **Felelősség**

Ez a tárgy az őt felhasználó hallgató szobáját mérgezővé teszi, aktiválás hatására.

- **Ősosztályok**

Item

- **Interfészek**

Nincs.

- **Asszociációk**

Ősosztályban található.

- **Attribútumok**

Ősosztályban található.

- **Metódusok**
  - **void activate(Student s)**: **[override]** Aktiválja a tárgyat, és a tárgy health-ét megfelelő értékre állítja. Meghívja a mérgező szoba létrejövéséhez szükséges függvényeket.

4.3.3 Door

- **Felelősség**

Az ajtón át elérhető szobák és az ajtó tulajdonságainak tárolása.

- **Ősosztályok**

Nincs.

- **Interfészek**

Nincs.

- **Asszociációk**
  - **rooms**: Room osztállyal való asszociáció. Egy ajtó mindig két szobát tárol, mégpedig azon szobákat, amelyhez tartozik.

- **Attribútumok**
  - **bool closed**: Ezen attribútum bool értékkel tárolja, hogy az adott ajtó zárt vagy nyitva van.
  - **bool oneway**: Ezen attribútum bool értékkel tárolja, az esetleges egyirányúságot, mégpedig olyan módon, hogy ha igaz az értéke, akkor csak az osztályban a elsőként tárolt szobából lehet a másodiként tárolt szobába átjutni az ajtón, ha hamis, akkor pedig mindkét irányban használható az ajtó.

- **Metódusok**
  - **bool canPassThroughTo(Room r)**: Megvizsgálja, hogy a paraméterben kapott r szoba felé járható-e, az adott ajtó. Amennyiben igen a visszatérési értéke igaz, ha nem hamis.

4.3.4 Entity

- **Felelősség**

A játékban szereplő karakterek, funkcióit, tulajdonságait megvalósító osztály. Ez az osztály felel az entitások mozgatásáért, tárgyaik felvételeiért, lehelyezéséért és entitások tulajdonságainak beállításáért.

- **Ősosztályok**

Nincs.

- **Interfészek**

ITickable

- **Asszociációk**
  - **game**: Game osztállyal való asszociáció. Célja a játék alapvető állapotának változtatása. (Pl. nyerés regisztrálása)
  - **room**: Room osztállyal való asszociáció. Célja, hogy a karakter, folyton tárolhassa mely szobába tartózkodik. Csak azon egy szobát tárolja a karakter, amelyben tartózkodik.
  - **items**: Item osztállyal való asszociáció. Célja, hogy a karakter tárolni tudja az általa birtokolt tárgyakat. Egyszerre maximum öt tárgyat birtokolhat egy karakter.

- **Attribútumok**
  - **int stunnedFor:** Számértékkel tárolja, hány egységig bénult még a karakter.

- **Metódusok**
  - **void move(Room r)**: A paraméterben kapott szobába megpróbálja átléptetni, az adott karaktert.
  - **void pickup(Item i):** A paraméterben kapott tárgyat megpróbálja felszedni az adott karakter, azaz a birtokolt tárgyak közé helyezni.
  - **void drop(Item i):** Ezen metódus hatására a paraméterben kapott tárgyat eldobja/lerakja az adott karakter, azaz kikerül a tárolt tárgyai közül.
  - **void dropItems():** Az adott karakter összes birtokolt tárgyát eldobatja ezen metódus.
  - **bool tryForLogarlec():** Ezen metódus abban az esetben hívódik, amennyiben az adott karakter logarléc típusú tárgyat próbál felvenni. Ezen metódus ezen alapimplementációjában a felvétel nem sikeres, így visszatérési értéke hamis (false).
  - **void meet(Entity e):** Ezen metódus a hívott entitást felszólítja, hogy mutatkozzon be a paraméterben kapott e entitásnak.
  - **void meetProf(Prof p):** Üres implementáció, a leszármazott osztályok ezen metódust fogják felüldefiniálni a bemutatkozásra, vagy a bemutatkozásra való reagálásra.
  - **void meetStudent(Student s):** Üres implementáció, a leszármazott osztályok ezen metódust fogják felüldefiniálni a bemutatkozásra, vagy a bemutatkozásra való reagálásra.
  - **bool toxicate():** Ezen metódus, az adott karaktert megpróbálja megmérgezni. Ezen alapimplementációban ez mindig sikeres, ezért a visszatérési értéke igaz lesz.
  - **bool immobilize():** Ezen metódus, az adott karaktert megpróbálja lebénítani. Ezen alapimplementációban ez mindig sikeres, ezért a visszatérési értéke igaz lesz.
  - **void addIem(Item i):** Ezen metódus a paraméterben kapott tárgyat az adott karakter birtokolt tárgyaihoz adja.
  - **void removeItem(Item i):** Ezen metódus a paraméterben kapott tárgyat az adott karakter birtokolt tárgyai közül kiveszi.

- **void tick():** Ezen metódus adott időegységenként szükséges feladatok elvégzését látja el az adott karakteren. Ezen implementációja esetén, ha szükséges az attribútum értékét csökkenti.
- **Items takeItems():** Ezen metódus visszatérési értékben visszaadja, az adott karakter által tárolt tárgyakat és eltávolítja az adott karakter összes birtokolt tárgyát.

4.3.5 FFP2

- **Felelősség**

Azon tárgy, amely a hallgatókat képes megvédeni mérges gázzal teli szoba hatásától.

- **Ősosztályok**

Item

- **Interfészek**

Nincs.

- **Asszociációk**

Ősosztályban található.

- **Attribútumok**

Ősosztályban található.

- **Metódusok**
  - **bool protectAgainstToxicRoom(): [override]** Megvizsgálja ezen metódus, hogy az adott FFP2-es maszk, az adott állapotában képes-e megvédeni, egy hallgatót mérges gázzal teli szoba hatásától, és amennyiben igen igaz, ha nem hamis értékkel tér vissza a metódus.

4.3.6 Game

- **Felelősség**

A játék alapvető állapotát, építő elemeit tároló és azokon alapvető globális feladatokat hívó osztály.

- **Ősosztályok**

Nincs.

- **Interfészek**

ITickable

- **Asszociációk**
  - **entities**: Entity osztállyal való asszociáció. Célja tárolni a játékban szereplő összes entitást és időközönként frissítést hívni azokon.
  - **rooms**: Room osztállyal való asszociáció. Célja tárolni a játékban szereplő összes szobát és időközönként frissítést hívni azokon.

- **Attribútumok**
  - **bool gameWon:** Ezen attribútum bool értékben tárolja, hogy megnyerték-e már a hallgatók a játékot.

- **Metódusok**
  - **void tick()**: Ezen metódus időegységi frissítést hív, az osztályban tárolt entitásokon és szobákban.
  - **void entityDied(Entity e)**: Ezen metódus a paraméterben kapott e entitást törli az osztályban tárolt entitások közül.

4.3.7 Item

- **Felelősség**

A játékban megtalálható tárgyakat reprezentáló osztály. A tárgyak felhasználásáért, összekapcsolásáért, valamint felvételéért felel.

- **Ősosztályok**

Nincsenek.

- **Interfészek**

Nincsenek.

- **Asszociációk**
  - **pair**: Tárol egy tárgyat, ezzel a tárggyal van összekapcsolva, ha az érvényes objektum.

- **Attribútumok**
  - **int health**: Azt tárolja, hogy hány életpontja van még az adott tárgynak. Ez annyit jelent, hogy ennyiszer tudjuk még aktiválni, vagy ennyiszer fog még megvédeni minket támadástól.

- **Metódusok**
  - **activate(s: Student):** Paraméterként kapja azt a hallgatót, aki meghívta. Ez egy üres implementáció, arra szolgál, hogy a leszármazott tárgyak felüldefiniálják, ha parancsra fel lehet őket használni.
  - **bool protectAgainstToxicRoom():** Arra szolgál, hogy a tárgy megvédje a hallgatót a mérgező szoba támadásától. Ebben az implementációban hamissal tér vissza, a leszármazottak, amiknek van ilyen képességük felüldefiniálják.
  - **bool protectAgainstProf():** Arra szolgál, hogy a tárgy megvédje a hallgatót az oktatók támadásaitól. Ebben az implementációban hamissal tér vissza, a leszármazottak, amiknek van ilyen képességük felüldefiniálják.
  - **link(i: Item):** A hívott tárgyat összekapcsolja a kapott tárggyal.
  - **bool linkTransistor(tr: Transistor):** Ebben az implementációban hamissal tér vissza, arra szolgál, hogy a tranzisztor felüldefiniálja. Használata azt biztosítja, hogy tranzisztort csak tranzisztorral lehessen összekapcsolni.
  - **bool pickup(e: Entity):** Ebben az implementációban igazzal tér vissza. Arra szolgál, hogy például a logarléc vagy a tranzisztor felüldefiniálja és új viselkedést vezessenek be.
  - **decrementHealth():** Csökkenti a tárgy életpontjait eggyel, ha az nem nulla.

4.3.8 ITickable

- **Felelősség**

A játékban megtalálható osztályokból mindnek meg kell valósítani, amely egy egység eltelte után szeretne csinálni valamit.

- **Ősosztályok**

Nincsenek.

- **Asszociációk**

Nincsenek.

- **Metódusok**
  - **tick():** Az adott időegységenként szükséges feladatok elvégzését hívja elő.

4.3.9 Logarlec

- **Felelősség**

A játékban megtalálható logarléc objektumot reprezentálja. Különleges felelőssége, hogy oktatóknak ne engedje a felvételét, hallgatóknak pedig igen.

- **Ősosztályok**

Item

- **Interfészek**

Nincsenek.

- **Asszociációk**

Ősosztályban megtalálható.

- **Attribútumok**

Ősosztályban megtalálható.

- **Metódusok**
  - **bool pickup(e: Entity): [override]** A paraméterében megkapja, hogy melyik entitás szeretné felvenni. Ezen entitáson meghívja a tryForLogarLec metódust, hogy eldöntse, hogy az entitás felveheti-e az objektumot vagy sem.

4.3.10 Prof

- **Felelősség**

Az oktató alapvető funkcióit, azaz mozgását tárgy kezelését, és más karakterekkel való találkozását megvalósító osztály.

- **Ősosztályok**

Entity

- **Interfészek**

ITIckable

- **Asszociációk**

Ősosztályban megtalálható.

- **Attribútumok**

Ősosztályban megtalálható.

- **Metódusok**
  - **void meet(Entity e): [override]** Ezen metódus a hívott entitást felszólítja, hogy mutatkozzon be a paraméterben kapott e entitásnak, mégpedig a meetProf() metódus segítségével.
  - **void meetStudent(Student s): [override]** A hívott oktató ezen metódus hatására megpróbálja kibuktatni(, játék szempontjából megölni) a paraméterben kapott s hallgatót.

4.3.11 Room

- **Felelősség**

A szoba objektumokat reprezentáló osztály. Entitások befogadásáért, eltávolításáért, tárgyak szobában való elhelyezéséért, szobából való felvételéért, egyesülésért és osztódásért felel. Tulajdonságától függően máshogy reagálhat egy befogadási kérelemre. Az entitások rajta keresztül tudnak elkérni, tőle tudnak felvenni tárgyakat.

- **Ősosztályok**

Nincsenek.

- **Interfészek**

ITickable

- **Asszociációk**
  - **doors**: A szoba ajtókkal rendelkezik. Fontos a szobák közötti mozgáskor.
  - **entities**: A szobában található entitások. A szoba a tulajdonságától függően lép kapcsolatba velük, valamint értesíti őket, ha más entitás a szobába lép.
  - **items**: A szobában található tárgyak. Az entitások tudnak a szobában letenni és felvenni tárgyakat.

- **Attribútumok**
  - **int capacity:** Egy szobában ennyi entitás tartózkodhat egyidejűleg.
  - **bool isToxic:** Tulajdonság. Azt jelzi, hogy a szoba mérgező gázzal teli szoba-e.
  - **bool isWet:** Tulajdonság. Azt jelzi, hogy a szobának a táblája nedves-e.
  - **bool isCursed:** Tulajdonság. Azt jelzi, hogy a szoba elátkozott-e.

- **Metódusok**
  - **bool acceptEntity(e: Entity):** Megpróbálja befogadni a paraméterként kapott entitást. Először ellenőrzi, hogy a szoba kihasználja-e a kapacitását, ha igen, akkor nem engedi be az entitást. Tulajdonságaitól függően megtámadhatja a belépő entitást, tehát megmérgezheti, ha mérgező a szoba és megbéníthatja, ha nedves a szoba táblája. Ha az entitás menet közben elbénult, akkor hamis értékkel tér vissza. Ha kiállta a próbákat az entitás és nem bénult el, akkor felszólítja a már szobában lévő entitásokat, hogy mutatkozzanak be az újonnan érkező entitásnak. Ha mindez megtörtént és igazzal tér vissza.
  - **addEntity(e: Entity):** A szoba entitásaihoz ad egy újabbat.
  - **removeEntity(e: Entity):** Egy entitást eltávolít a sajátjai közül.
  - **bool take(e: Entity, i: Item):** Jelentése: az e entitás az i tárgyat szeretné elvenni tőle, igazzal tér vissza, ha el tudja venni és hamissal, ha nem.
  - **addItem(i: Item):** Hozzáad a szobában tárolt tárgyakhoz egy újat.
  - **removeItem(i: Item):** Eltávolít a szobákban található tárgyak közül egyet (ha tartalmazza azt).
  - **bool merge(r: Room):** A hívott szobába olvad a paraméterként kapott szoba. Ennek jelentése az, hogy a hívott szoba magára aggasztja a paraméterként kapott szoba tulajdonságait, átveszi (és átírja) az ajtajait, valamint átveszi a tárgyait is. Nem megy végbe az egyesülés, ha tartózkodnak a szobákban entitások vagy nem szomszédosok. Ilyenkor hamissal tér vissza, egyébként igazzal.
  - **Room split():** A hívott szobát bontja, osztja, ezáltal létrehoz egy új szobát. A szobák osztoznak a hívott szoba tulajdonságain, ajtajain (át kell írni), valamint tárgyain. Nem megy végbe az osztódás, ha tartózkodnak a szobában, ilyenkor érvénytelen értéket ad, egyébként a szobát kapjuk vissza.
  - **tick():** Tulajdonságaitól függően megtámadhatja a benne megtalálható entitásokat, valamint eltűntetheti/megjelenítheti a szoba ajtajait.
  - **magicDoors():** Eltűnteti, megjeleníti a szoba ajtajait.

4.3.12 Student

- **Felelősség**

A hallgatók alapvető funkcióit, azaz mozgását tárgy kezelését, és más karakterekkel való találkozását megvalósító osztály

- **Ősosztályok**

Entity

- **Interfészek**

ITickable

- **Asszociációk**

Ősosztályban megtalálható.

- **Attribútumok**
  - **int drunkFor:** Ezen attribútumba számértékkel tárolódik, hány egységig részeg még a karakter, azaz védett a professzor támadástól.

- **Metódusok**
  - **void kill():** A hívott karaktert megpróbálja megölni ezen metódus.
  - **bool tryForLogarlec(): [override]** Ezen metódus abban az estben hívódik, amennyiben az adott karakter logarléc típusú tárgyat próbál felvenni. A student osztály által ezen felüldefiniált verziója a metódusnak, a játék megnyeréséhez vezet.
  - **bool teleport(Room r):** Ezen metódus az adott hallgatót a paraméterben kapott r szobába próbálja teleportálni. Visszatérése a teleportálás sikerességétől függ, amennyiben sikeres igaz, ha nem sikeres hamis a visszatérési érték.
  - **void meet(Entity e): [override]** Ezen metódus a hívott hallgatót felszólítja, hogy mutatkozzon be a paraméterben kapott e entitásnak a meetStudent() metódus segítségével.
  - **void meetProf(Prof p): [override]** Ezen metódus a hívott hallgatót felszólítja, hogy mutatkozzon be a paraméterben kapott p oktatónak a meetStudent() metódussal.
  - **bool toxicate(): [override]** Ezen metódus, az adott karaktert megpróbálja megmérgezni. Amennyiben sikeres a visszatérési értéke igaz, ha nem sikeres hamis.
  - **bool immobilize(): [override]** Ezen metódus, az adott karaktert megpróbálja lebénítani. Ezen implementációjában a metódus visszatérési értéke hamis, mivel a hallgatók ilyen formában nem bénithatók.
  - **void linkItems(Item i1, Item i2):** Ezen metódus a paraméterben kapott két tárgyat próbálja összepárosítani.
  - **void activateItem(Item i):** Ezen metódus a paraméterben kapott i tárolt tárgyat aktiválja.
  - **void tick(): [override]** Ezen metódus adott időegységenként szükséges feladatok elvégzését látja el az adott karakteren. Ezen implementációja esetén, ha szükséges az attribútumok értékét csökkenti.

4.3.13 Transistor

- **Felelősség**

A játékban megtalálható tranzisztorokat reprezentáló osztály. A tranzisztorok összekapcsolásáért, aktiválásért és felvételéért felel.

- **Ősosztályok**

Item

- **Interfészek**

Nincsenek.

- **Asszociációk**
  - **room:** Tárol egy szobát. Ha aktív és van párja, akkor a párjának aktiválása esetén az aktiválást kezdeményező hallgató ebbe a szobába fog kerülni.

- **Attribútumok**
  - **bool active**: Azt mondja meg, hogy az adott tranzisztor aktív, tehát a párjának aktiválásával lehet hozzá teleportálni.

- **Metódusok**
  - **activate(s: Student): [override]** A tranzisztor (és párjának) állapotától függően eltérően viselkedik. Ha a tranzisztornak nincs párja, akkor azonnal visszatér. Ha van párja, de az nem aktív, akkor beregisztrál egy szobát és aktívvá állítja a tranzisztort, majd lehelyezi azt a szobában. Ha van párja és az aktív, akkor ez a tranzisztor kikerül a hallgató tárgyai közül a hallgató szobájába és a párjában beregisztrált szobához teleportálja felhasználóját, a hallgatót.
  - **bool pickup(e: Entity): [override]** Ha a tranzisztor aktív, akkor hamissal tér vissza, ezzel megtagadva a felvételt. Ellenkező esetben igazzal tér vissza.
  - **link(i: Item): [override]** Meghívja a paraméterként kapott tárgyon a linkTransistor metódust, ha az tranzisztor, akkor a linkTransistor igazzal tér vissza, ilyenkor a link hívója is beregisztrálja magában a másik tárgyat, hiszen az egy tranzisztor.
  - **bool linkTransistor(tr: Transistor): [override]** Beregisztrálja magában a kapott tranzisztort egy párként, majd igazzal tér vissza.

4.3.14 TVSZ

- **Felelősség**

A játékban szereplő TVSZ objektumok reprezentálásért felel. A felhasználóját megvédi az oktatók támadásaitól.

- **Ősosztályok**

Item

- **Interfészek**

Nincsenek.

- **Asszociációk**

Ősosztályban megtalálható.

- **Attribútumok**

Ősosztályban megtalálható.

- **Metódusok**
  - **protectAgainstProf(): [override]** Arra szolgál, hogy a tárgy megvédje a hallgatót az oktatók támadásaitól. Ebben az implementációban igazzal tér vissza.

4.3.15 WetSponge

- **Felelősség**

A szobát, amelyben a hallgató aktiválja ezt a tárgyat nedvessé teszi (letörlődik a tábla).

- **Ősosztályok**

Item

- **Interfészek**

Nincs.

- **Asszociációk**

Ősosztályban megtalálható.

- **Attribútumok**

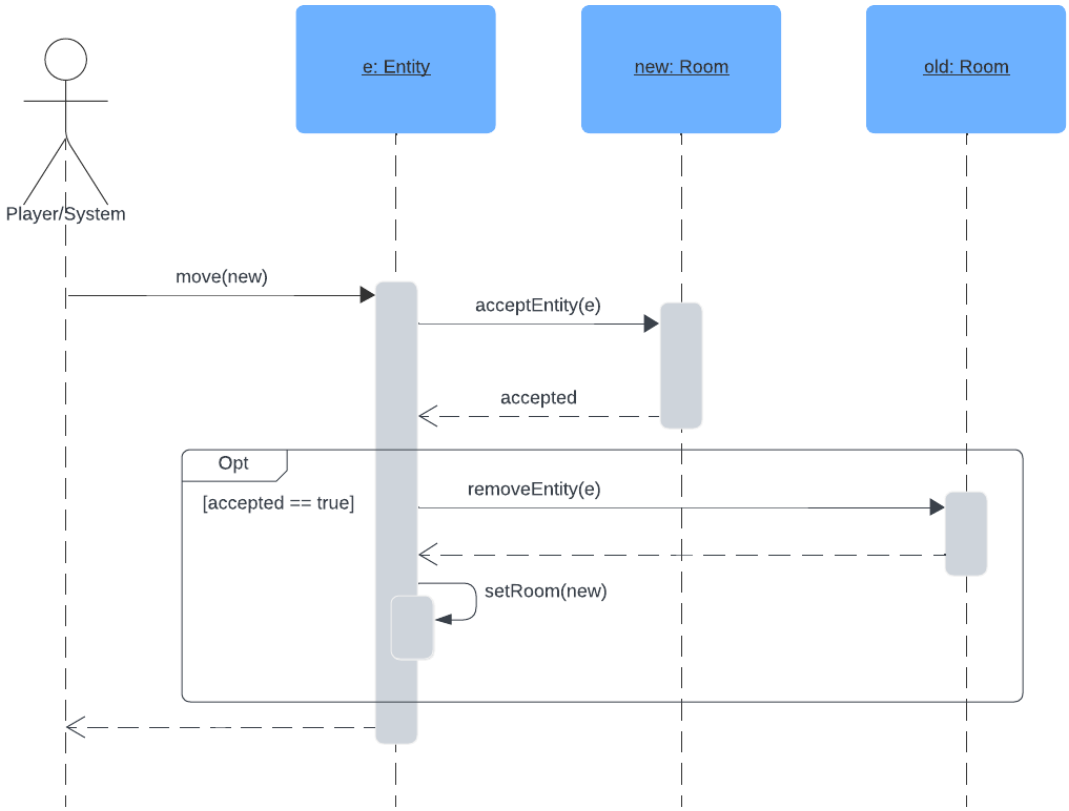
Ősosztályban megtalálható.

- **Metódusok**

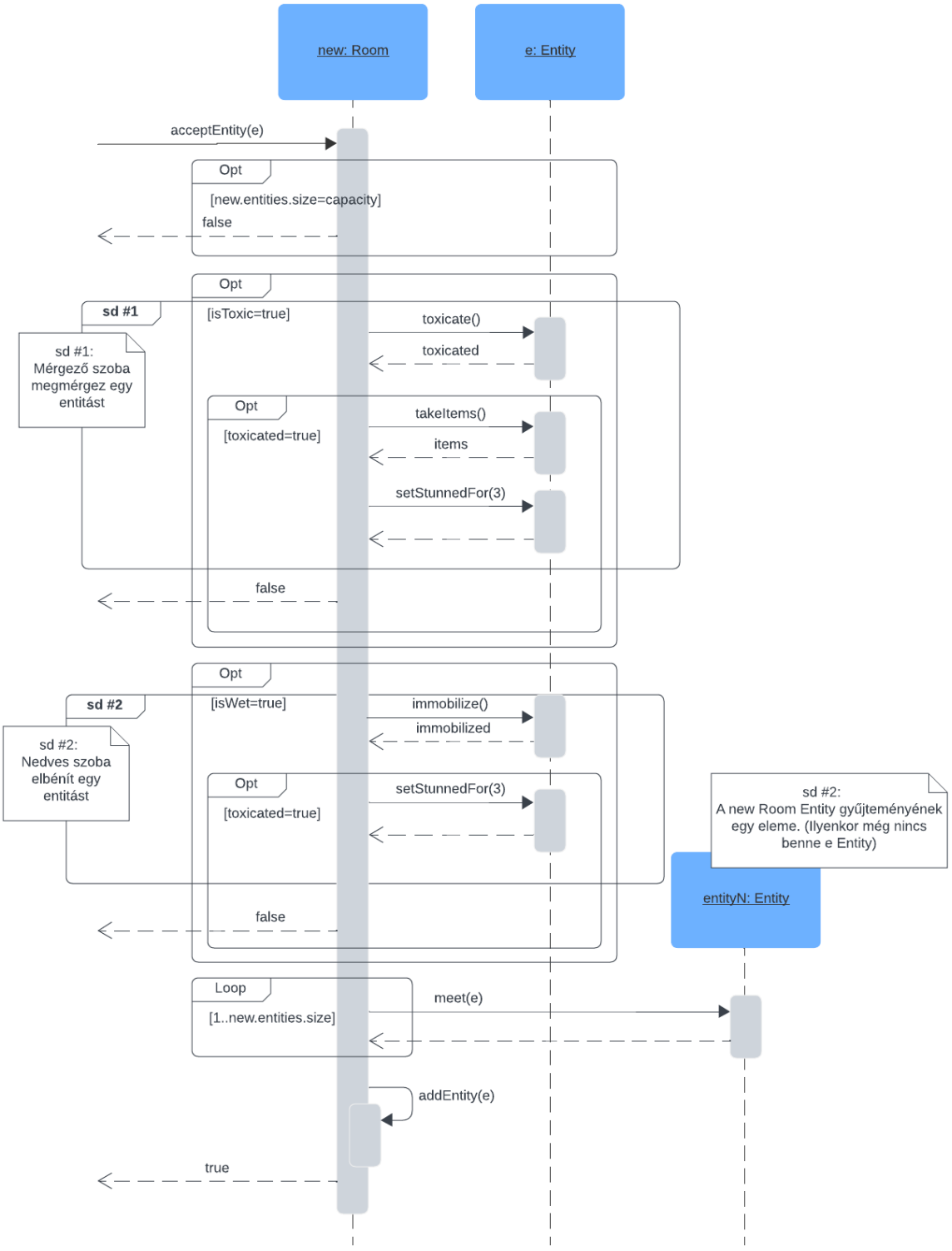
void activate(Student s): [override] Aktiválja az adott szivacsot ezzel, nedvesre állítva szobát, amelyben a paraméterben kapott s hallgató tartózkodik.

4.4 Szekvencia diagramok

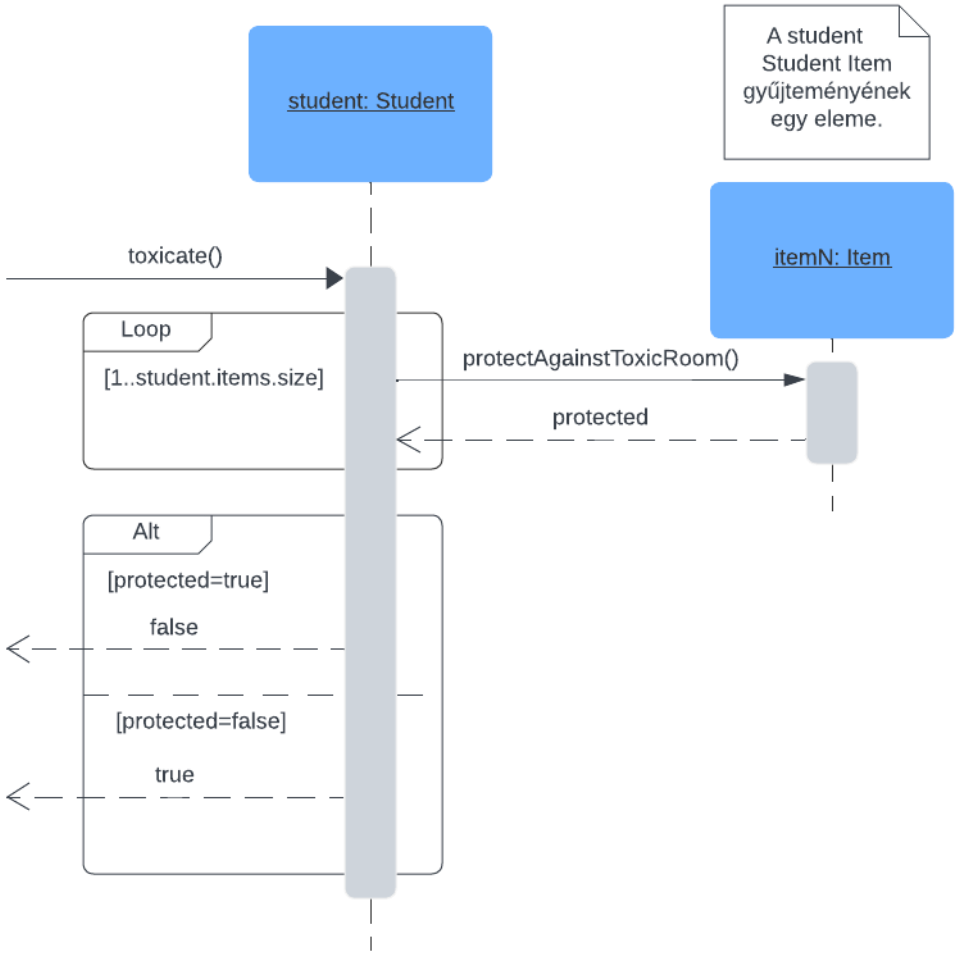
4.4.1 Entitás új szobába lép



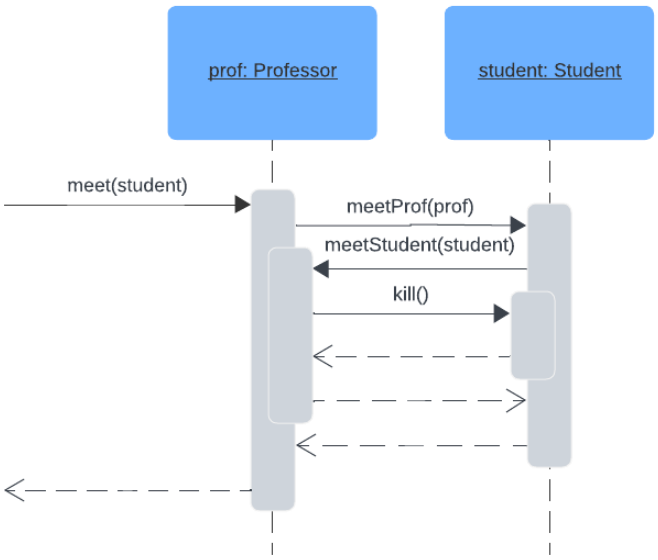
4.4.2 Szoba befogad egy entitást



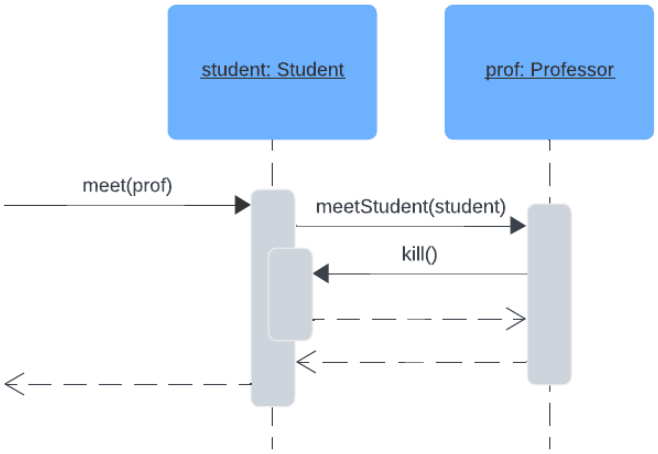
4.4.3 Megmérgeznek egy tanulót



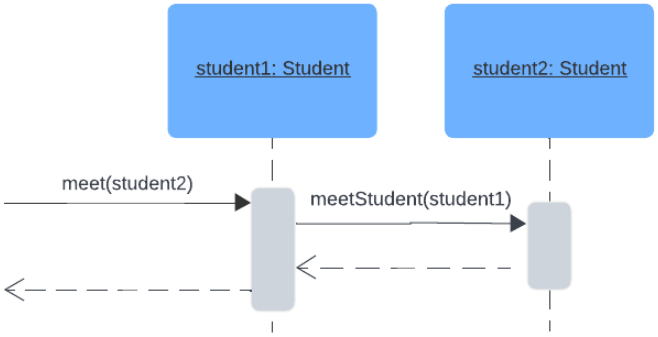
4.4.4 Oktató felszólítást kap, hogy mutatkozzon be egy hallgatónak



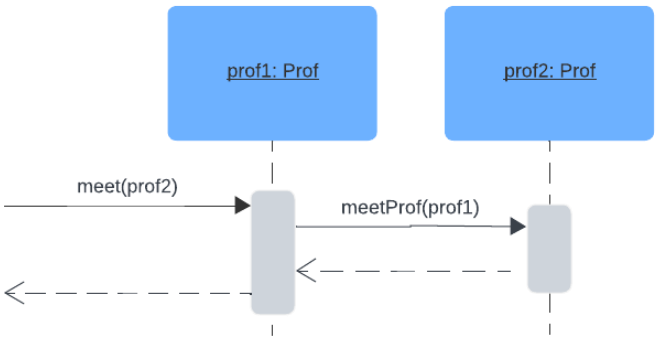
4.4.5 Hallgató felszólítást kap, hogy mutatkozzon be egy oktatónak



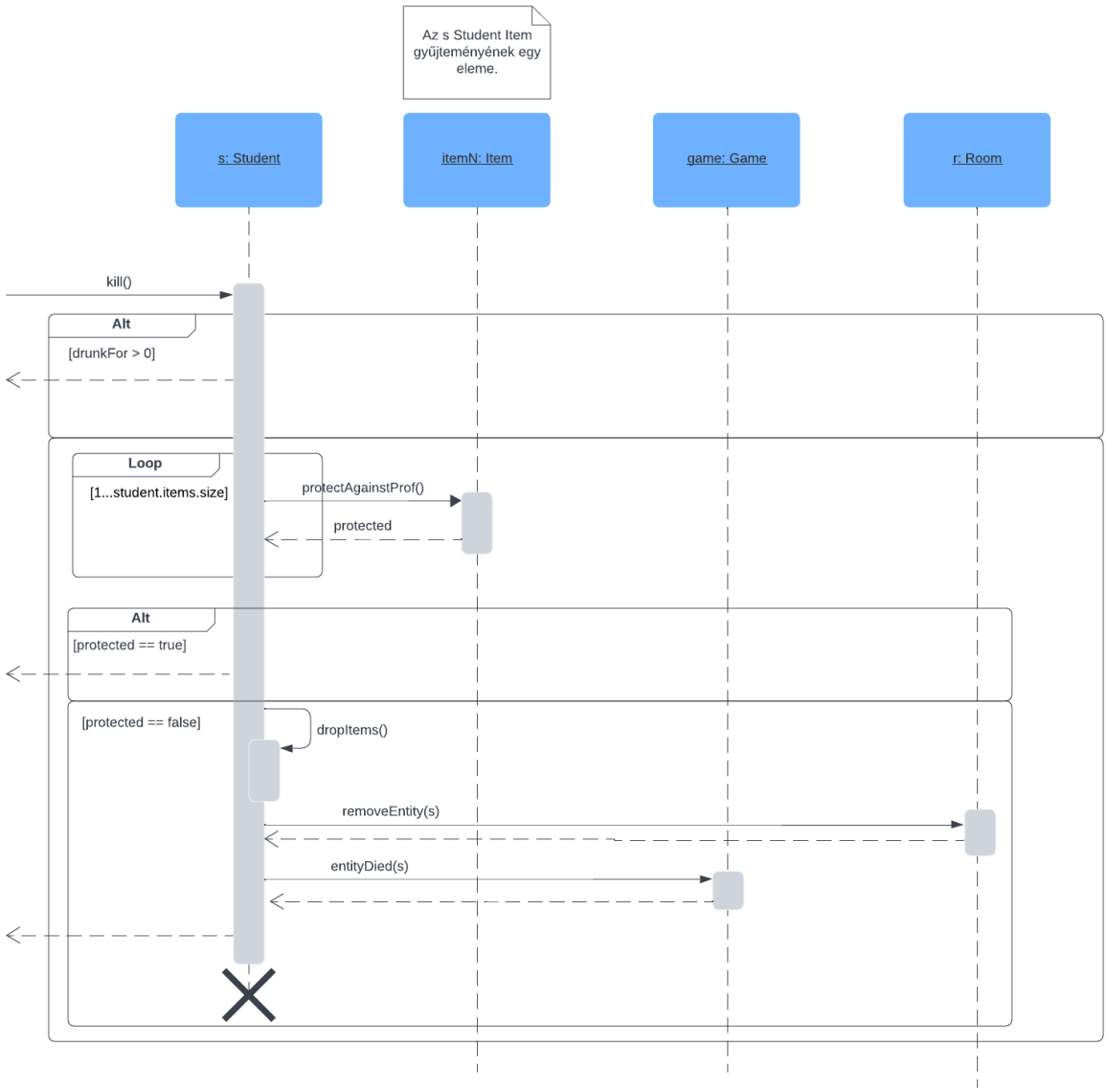
4.4.6 Hallgató felszólítást kap, hogy mutatkozzon be egy hallgatónak



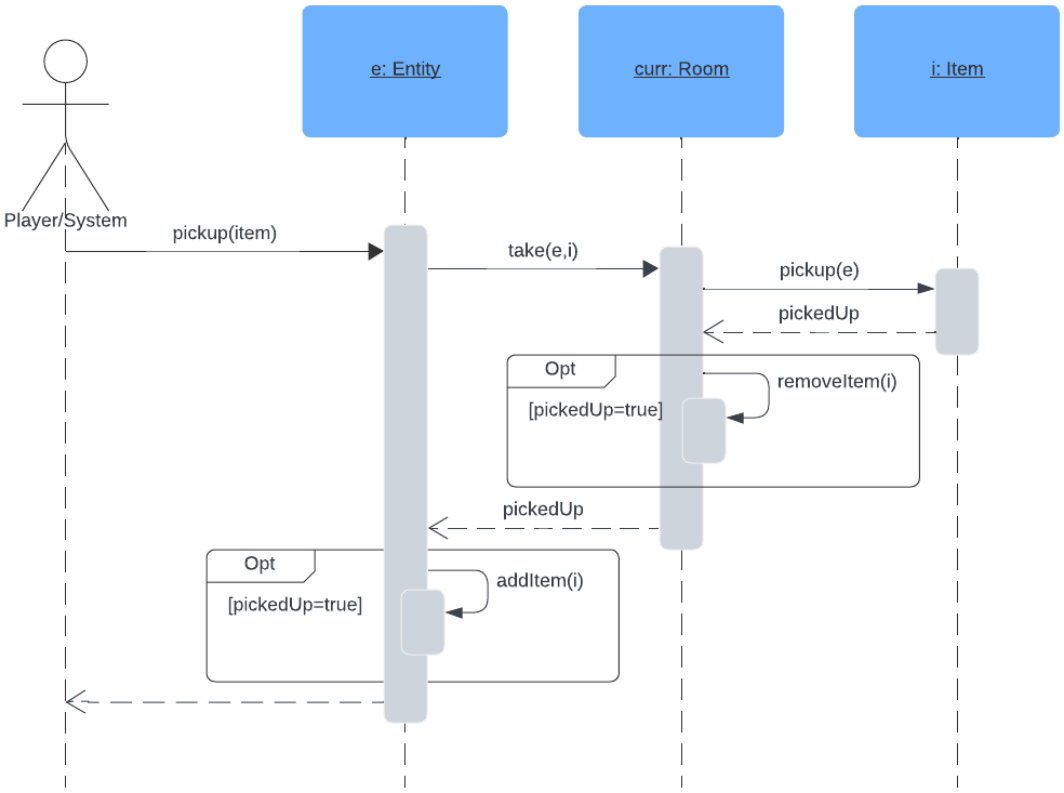
4.4.7 Oktató felszólítást kap, hogy mutatkozzon be egy oktatónak



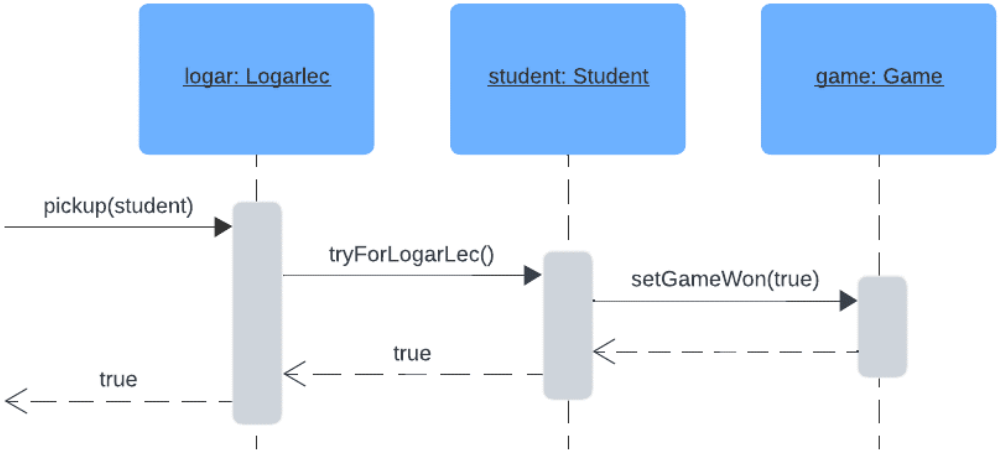
4.4.8 Megtámadnak egy hallgatót



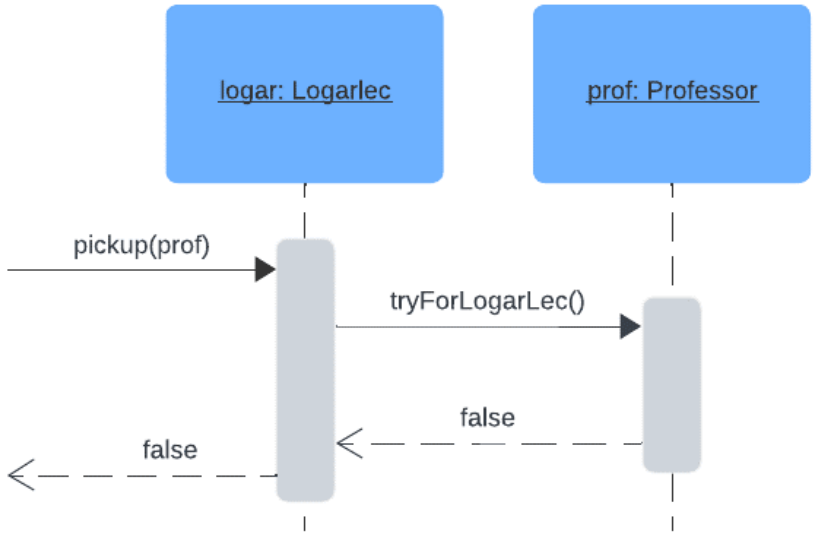
4.4.9 Tárgy felvétele



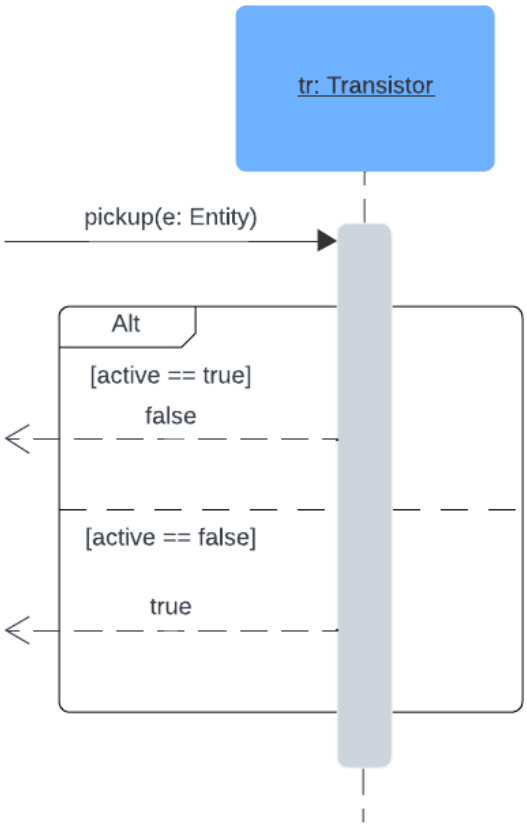
4.4.10 Hallgató felveszi a logarlécet



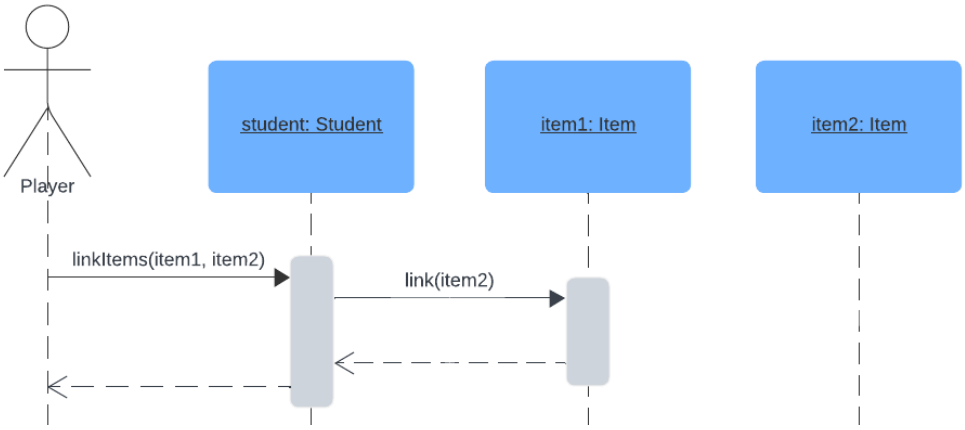
4.4.11 Oktató megpróbálja felvenni a logarlécet



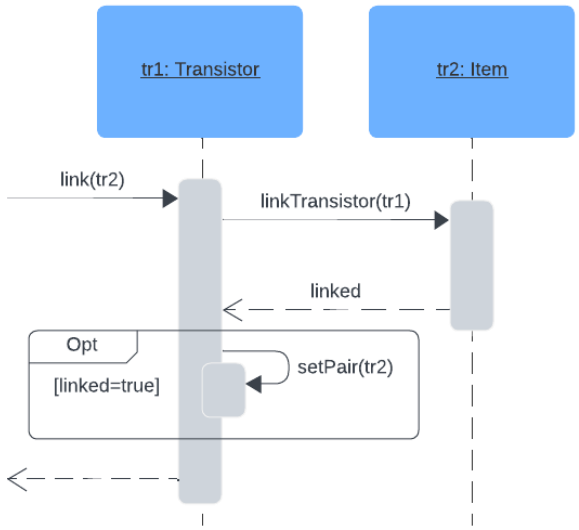
4.4.12 Tranzisztor felvétele



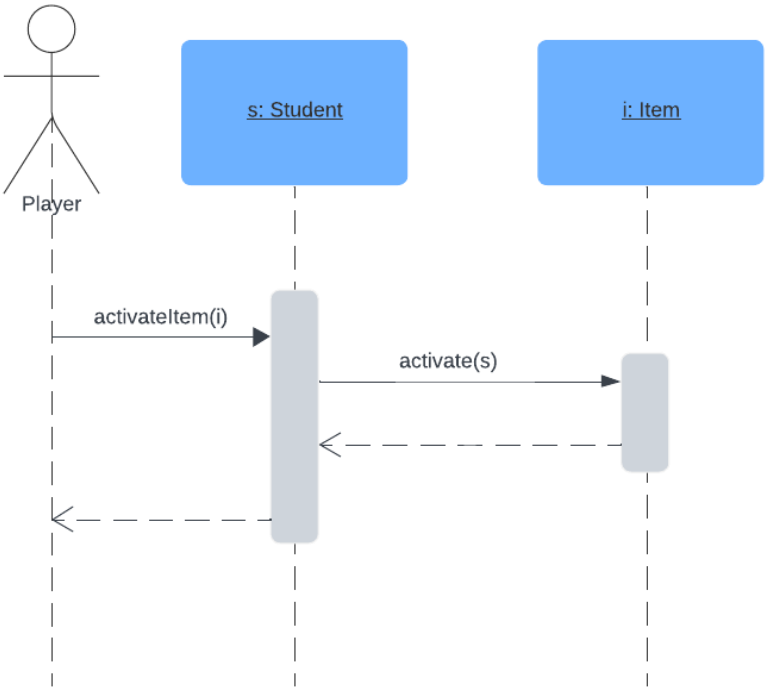
4.4.13 Két tárgy összekapcsolása



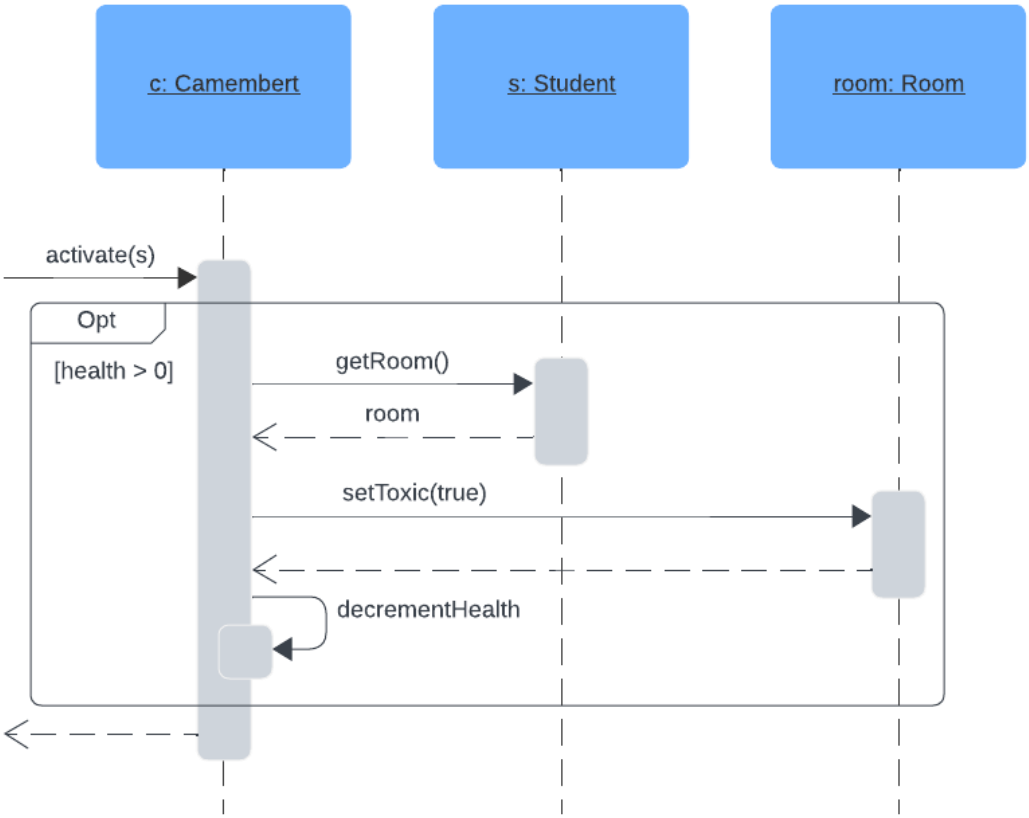
4.4.14 Tranzisztor összekapcsolása tárggyal



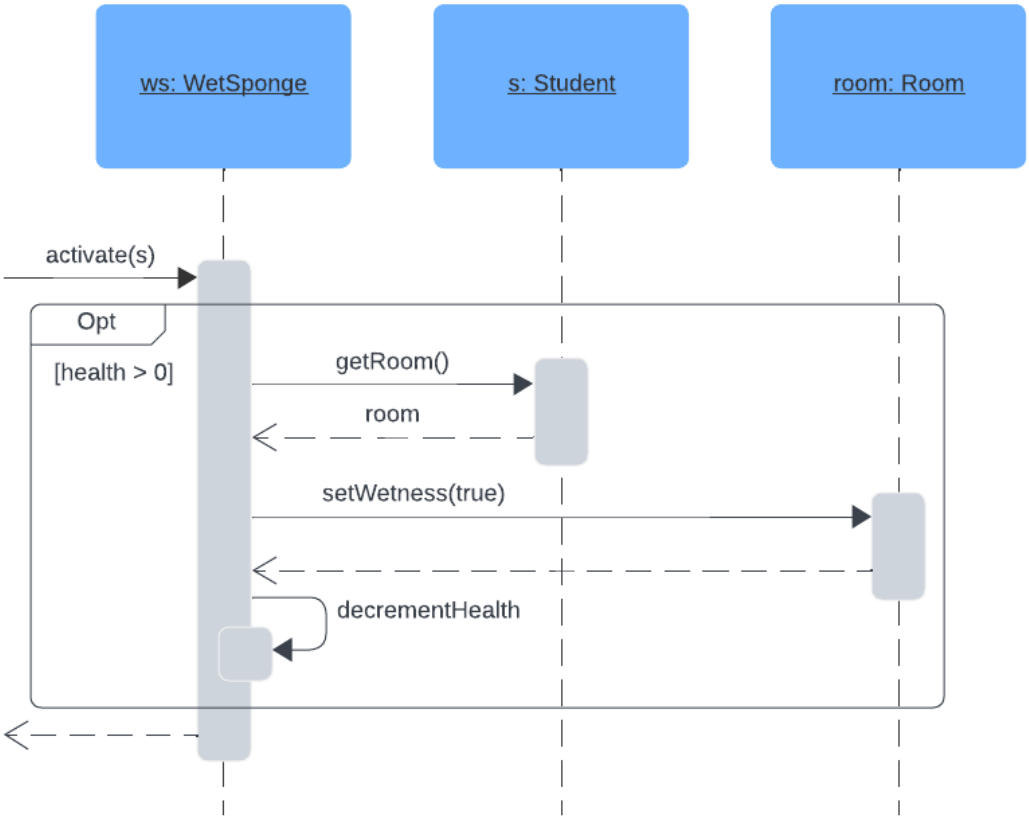
4.4.15 Tárgy aktiválása (felhasználása)



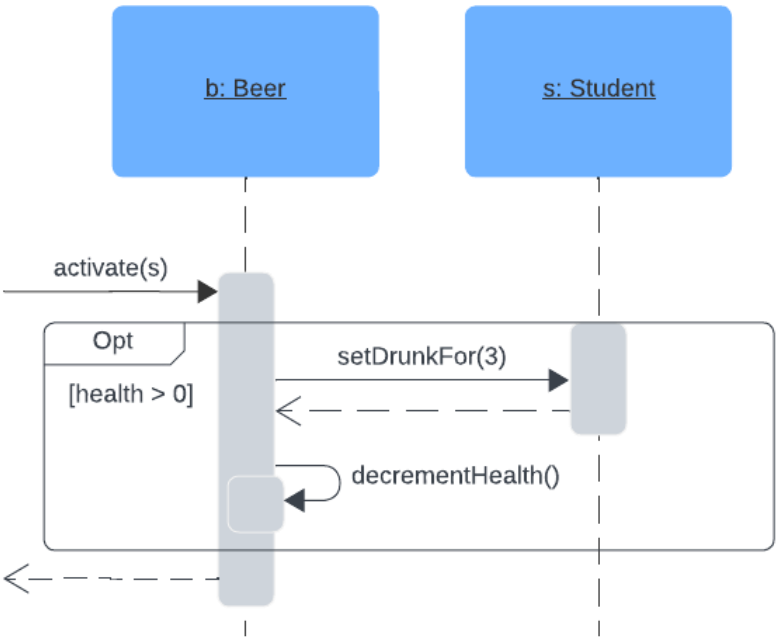
4.4.16 Dobozolt káposztás camembert aktiválása (felbontása)



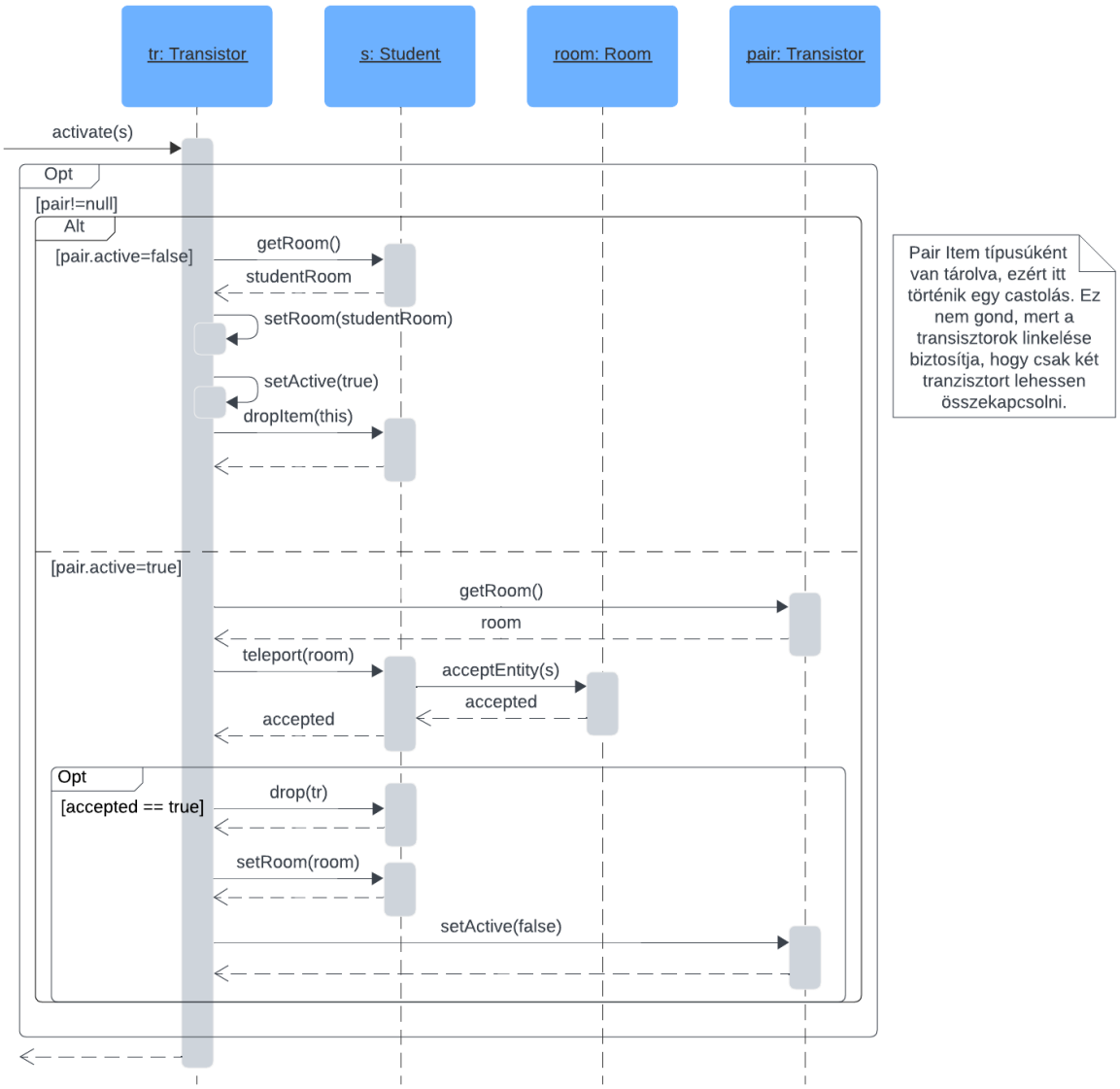
4.4.17 Nedves táblatörlőrongy aktiválása (tábla letörlése)



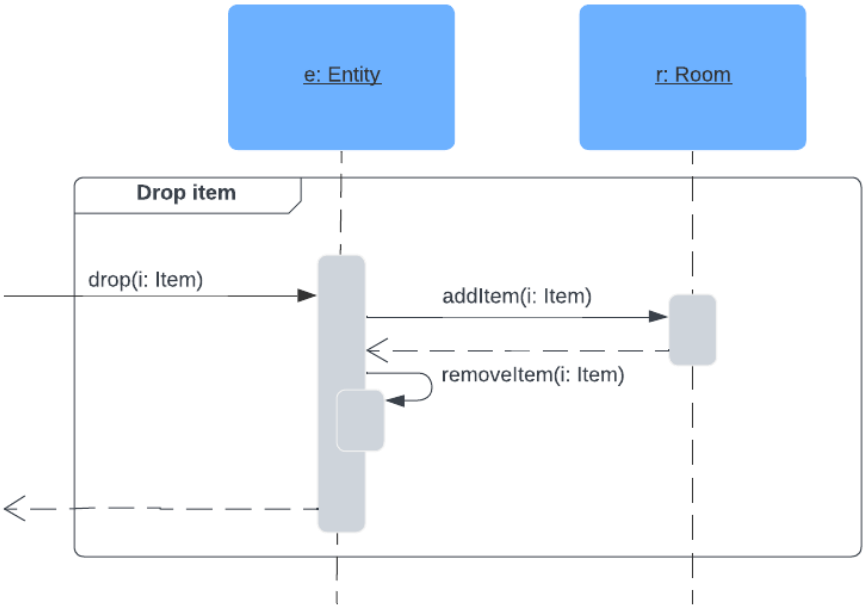
4.4.18 Sör aktiválása (elfogyasztása)



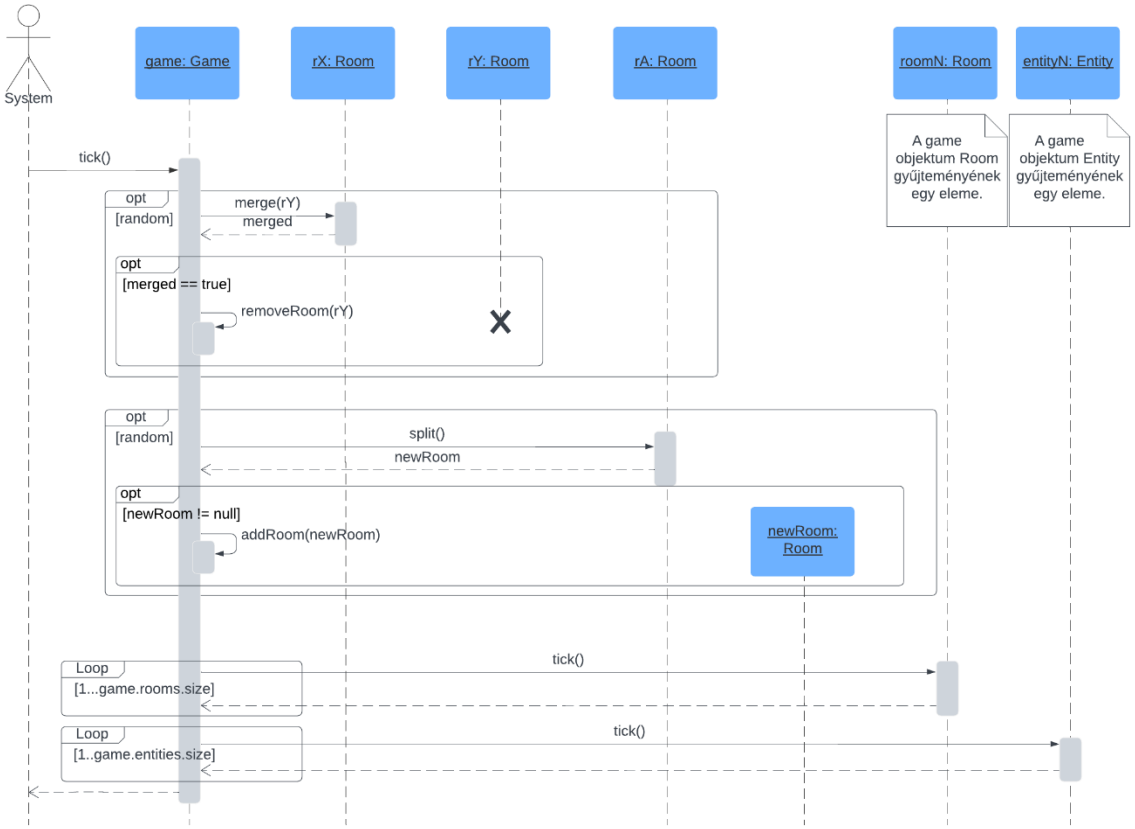
4.4.19 Tranzisztor aktiválása



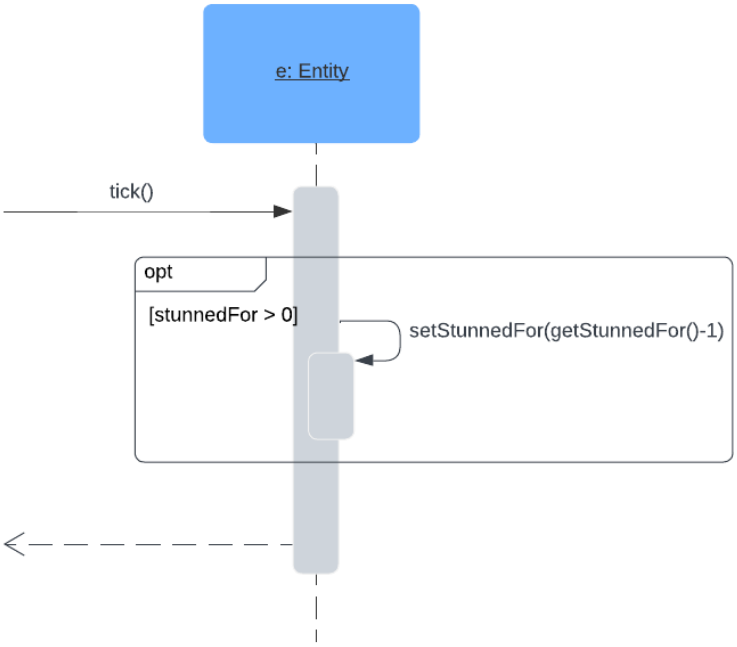
4.4.20 Tárgy letétele



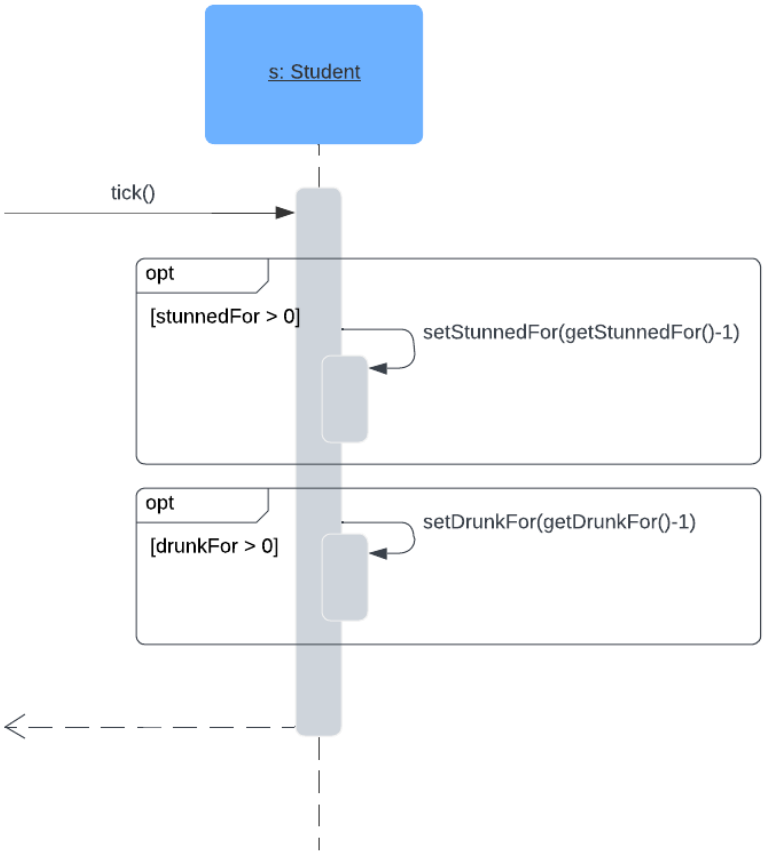
4.4.21 Játéktábla tickelése



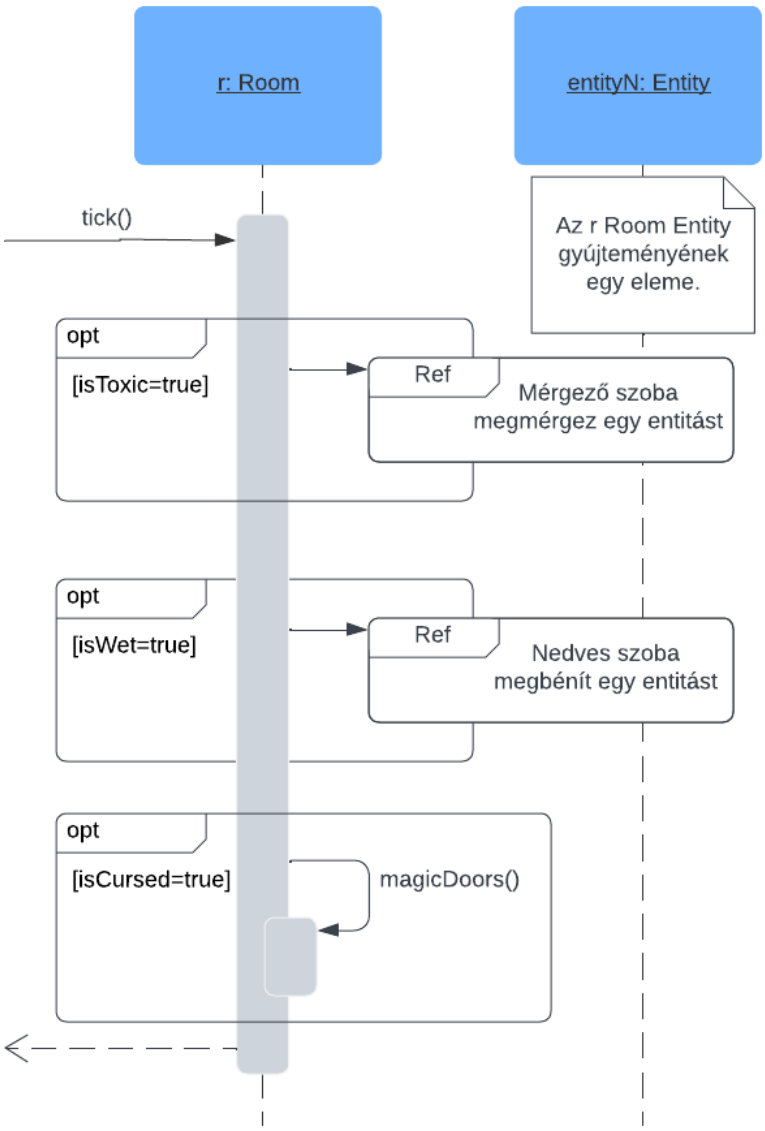
4.4.22 Entitás tickelése



4.4.23 Hallgató tickelése

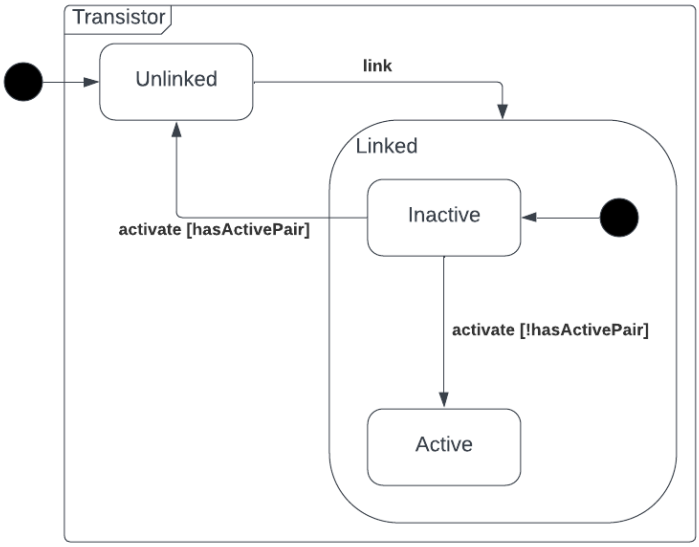


4.4.24 Szoba tickelése

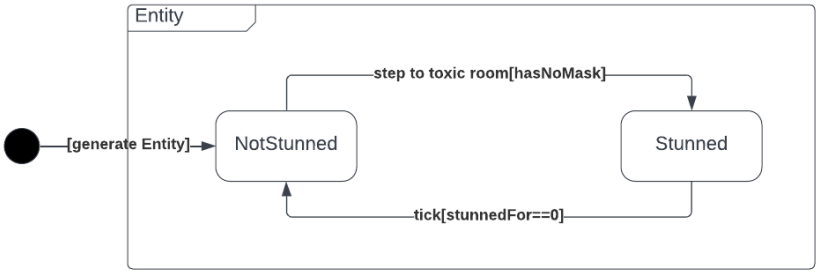


4.5 State-chartok

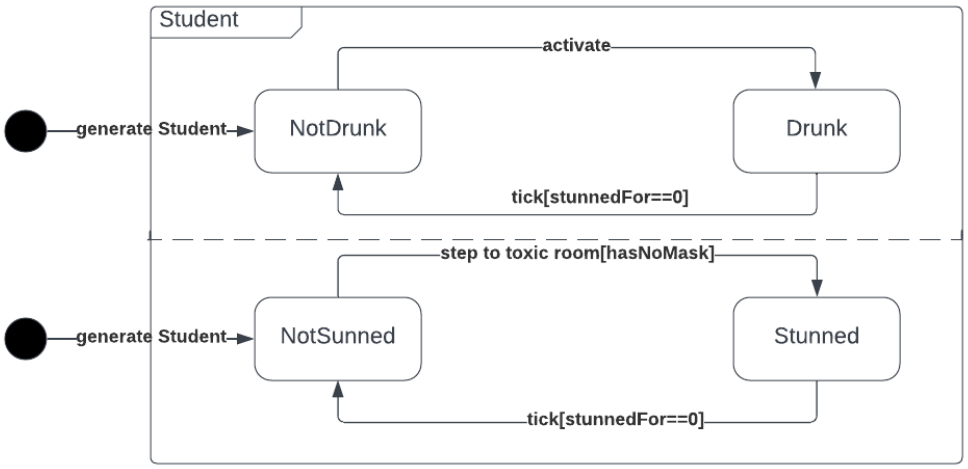
4.5.1 Tranzisztor állapotdiagram



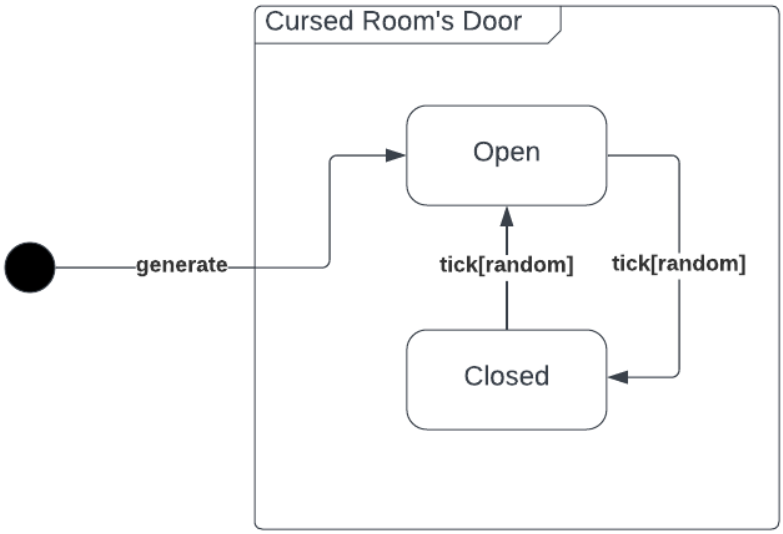
4.5.2 Entitás állapotdiagram



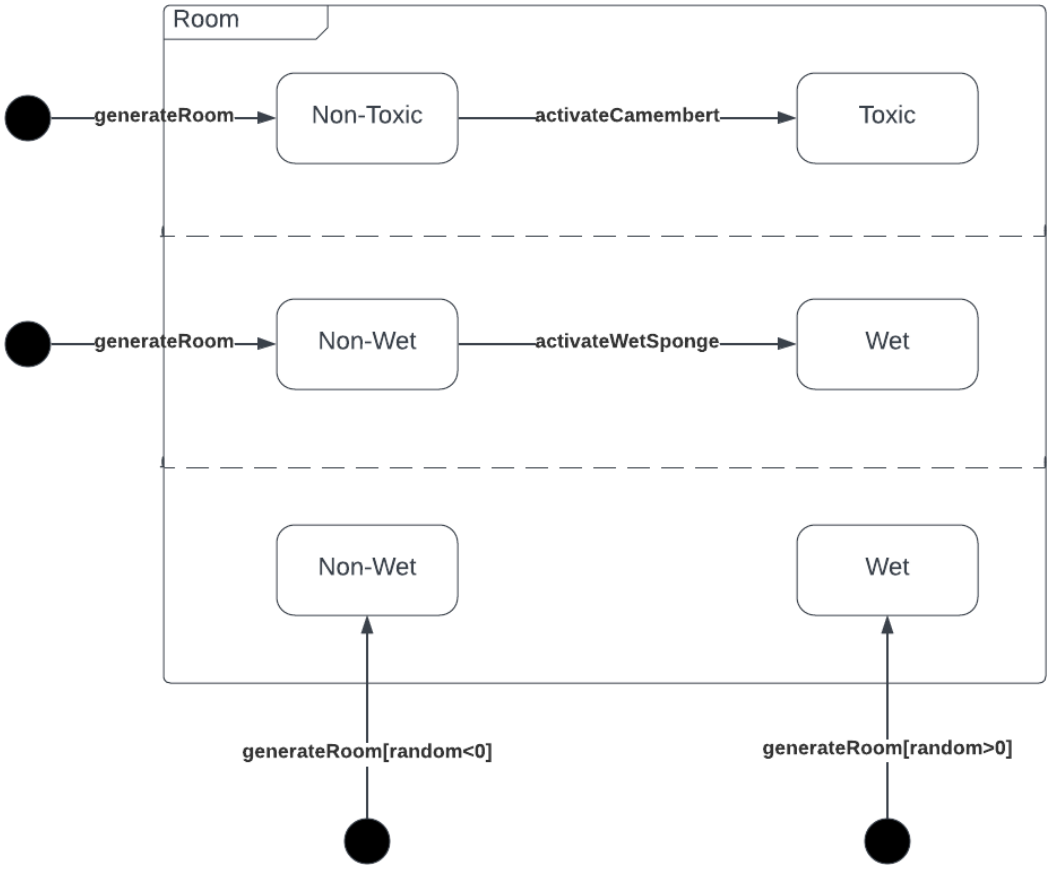
4.5.3 Hallgató állapotdiagram



4.5.4 Elátkozott szoba egy ajtajának állapotdiagramja



4.5.5 Szoba állapotdiagram



Napló

Kezdet	Időtartam	Résztvevők	Leírás
2024.03.07. 14:00	2 óra	Sánta	Állapot diagramm készítése
2024.03.08. 14:00	3 óra	Papp Sánta	Osztálydiagramm tervezés és kialakítás
2024.03.09. 10:00	3 óra	Papp Sánta	Osztálydiagramm, Szekvencia diagrammok rajzolása, tervezése
2024.03.09. 13:15	1 óra	Nagy Sánta	Állapot diagramm átnézés és szerkesztés
2024.03.09. 17:00	4,5 óra	Papp	Szekvencia diagramm tervezés és készítés
2024.03.09. 19:00	2 óra	Nagy	Szekvencia diagramm tervezés és készítés
2024.03.10. 10:00	2 óra	Nagy Papp	Szekvencia diagramm tervezés és osztálydiagramm pontosítás
2024.03.10. 12:00	2 óra	Nagy	Szekvencia diagramm és állapot diagram készítés
2024.03.10. 14:00	1 óra	Nagy Papp	Szekvencia diagramm tervezés és osztálydiagramm pontosítás

2024.03.10. 15:00	2 óra	Papp	Szekvencia diagramm készítés
2024.03.10. 15:00	2 óra	Nagy	Leírások elkészítésének elkezdése, szekvencia diagrammok pontosítása
2024.03.10. 18:00	10 óra	Nagy Papp	Osztály leírások elkészítése. Szekvencia, osztály és állapot diagrammok véglegesítés. Dokumentum szerkesztés simítások.