3. Analízis modell kidolgozása I.

53 – kedvenc_csapatom

Konzulens:
Potyók Csaba

Csapattagok

Fazekas Bence Mihály	W0TH54	bencemisi2003@gmail.com
Molnár Botond Kristóf	H1XYPA	molnarboti2003@gmail.com
Simon Tímea	SZFS1V	simontiti1109@gmail.com
Juhász Gábor	VA2469	ju.gabo14@gmail.com
Rimmel Botond	DMPMNZ	rimmelbotond@gmail.com

2024.03.03.

3. Analízis modell kidolgozása

3.1 Objektum katalógus

3.1.1 Labirintus

A játék színtere, amely szobákból áll, ezeket számon is tartja.

Képes ajtók megjelenítésére valamint törlésére, emellett szobákat tud egyesíteni és összeolvasztani.

3.1.2 Hallgató

A felhasználó által irányított karakter. Célja, hogy a számítógép által irányított oktatók ne kapják el (ezzel a halálukat okozva) azelőtt, mielőtt megszerzi a Logarlécet, mellyel megnyeri a játékot.

Van neve és számon tartja, hogy melyik szobában tartózkodik. Körönként egyszer akciót hajt végre, mely során egyszer vele szomszédos szobába léphet, ahova vezet ajtó és a létszám engedi. Ezenkívül az akciója során tud felvenni tárgyakat, ezeket szintén számontartja és a maximális számuk 5 lehet, a tárgyakat fel tudja használni, elégetni, valamint letenni.

A Hallgatókra a környezetük is hathat, amely különböző állapotokat idézhet elő, ezek a következők: mérges gáztól védett, oktatótól védett, bénult, eszméletét vesztett.

3.1.3 Oktató

A számítógép által irányított karakter. Célja, hogy megakadályozza a felhasználó/játékos által irányított hallgatót a játék megnyerésében. A vele egy szobában védettséggel nem rendelkező hallgatót/hallgatókat megöli.

Van neve és számon tartja, hogy melyik szobában tartózkodik. Körönként egyszer akciót hajt végre, mely során először egyszer vele szomszédos szobába lép (ha van ilyen), ahova vezet ajtó és a létszám engedi. Ezt követően a szobában lévő összes tárgyat felveszi és egyből el is égeti. Ha a Logarlécet vette fel, azt nem égeti el, hanem néhány kör után a lépése előtt leteszi.

Az Oktatókra a környezetük is hathat, amely különböző állapotokat idézhet elő, ezek a következők: bénult, eszméletét vesztett.

3.1.4 Szoba

A szoba az a színtér, ahol a játék játszódik. A játékteret (Labirintust) számtalan szoba alkotja, amelyeknek különböző befogadóképessége van.

Van szobaszáma és rendelkezik befogadóképességgel, ezenkívül számontartja a benne tartózkodó karaktereket.

Képes osztódásra és egyesülésre, valamint megmondani, hogy egy lépéssel meg szobákba lehet belőle eljutni.

Vannak szobák melyek különleges képességekkel rendelkeznek, ilyenek az elátkozott és a gázos szobák. Az elátkozott szoba az a szoba, amelynek időnként eltűnhet az ajtaja, amin keresztül egy másik szobába átjuthatna a karakter. Az elátkozott szoba esetében az is

előfordulhat, hogy nem lehet oda bejutni, mivel az összes ajtaja eltűnt. A gázos szoba az a szoba, ahol a karakter a bent megtalálható mérges gáz miatt elájul.

Gázos szoba esetében, ha a karakterről lekerül a bénítás, addig élvez védettséget a gázzal szemben, ameddig nem távozik a szobából.

A szobákban tárgyak vannak, amelyeket a karakterek fel tudnak venni.

3.1.5 Logarléc

A logarléc az a tárgy, amelynek a megszerzésével a hallgató megnyeri a játékot. A tárgyat egyébként az oktató is fel tudja venni, azonban 1-2 lépés után megszabadul tőle, azaz visszarakja a játéktérre.

3.1.6 Tranzisztor

Olyan tárgy, amelyből kettőt összekapcsolva a hallgatók az egyik szobából a másikba tudnak teleportálni.

A tranzisztorok számon tartják, hogy van-e párjuk. Ha a hallgató eszköztárába van két tranzisztor, amelynek nincs párja, akkor azok automatikusan összekapcsolódnak.

A teleportáláshoz fel kell használni az egyik összekapcsolt tranzisztort, ezzel az bekapcsolva letevődik a szobában. Ilyenkor már a párját felhasználva automatikusan visszakerülünk abba a szobába, ahol az első tranzisztort otthagytuk. A teleportálás után a két tranzisztor a két szobában kikapcsolt és párosítatlan állapotban lesz. Ilyenkor ezek újra felhasználhatóvá válnak.

3.1.7 Nedves táblatörlő rongy

Olyan tárgy, amit a hallgató tud hatékonyan használni arra, hogy a hallgatóval egy szobában levő oktatókat lebénítsa. Rendelkezik használati idővel, mivel a felvételétől számítva csak adott ideig használható. A használhatósága körönként mindig csökken, és ha már nem használható, magától elég a karakter eszköztárában.

3.1.8 FFP2-es maszk

Olyan tárgy, amit a hallgató a mérges gázzal teli szobákban a gázzal szembeni védelemre tud használni. Rendelkezik használati idővel és számon tartja, hogy használatbalett-e már véve, mivel a legelső használatától számítva csak adott ideig működik. A használhatósága körönként mindig csökken, és ha már nem használható, magától elég a karakter eszköztárában.

3.1.9 TVSZ denevérbőrre nyomtatott példánya

Olyan tárgy, amit a hallgató hatékonyan tud arra használni, hogy megmentse a lelkét attól, hogy azt az oktatók elvegyék. Rendelkezik használati idővel és számon tartja, hogy használatbalett-e már véve, mivel a legelső használatától számítva már csak két alkalommal használható. A használhatósága minden alkalommal csökken, amikor egy oktató megpróbálja elvenni a lelkét, és ha már nem használható, magától elég a karakter eszköztárában.

3.1.10 Dobozolt káposztás camembert

Olyan tárgy, amelyet a karakter tud felhasználni arra, hogy segítségével mérges gázt bocsásson ki a szobában. Ennek hatására a szoba gázos szobává válik.

3.1.11 Szent söröspohár

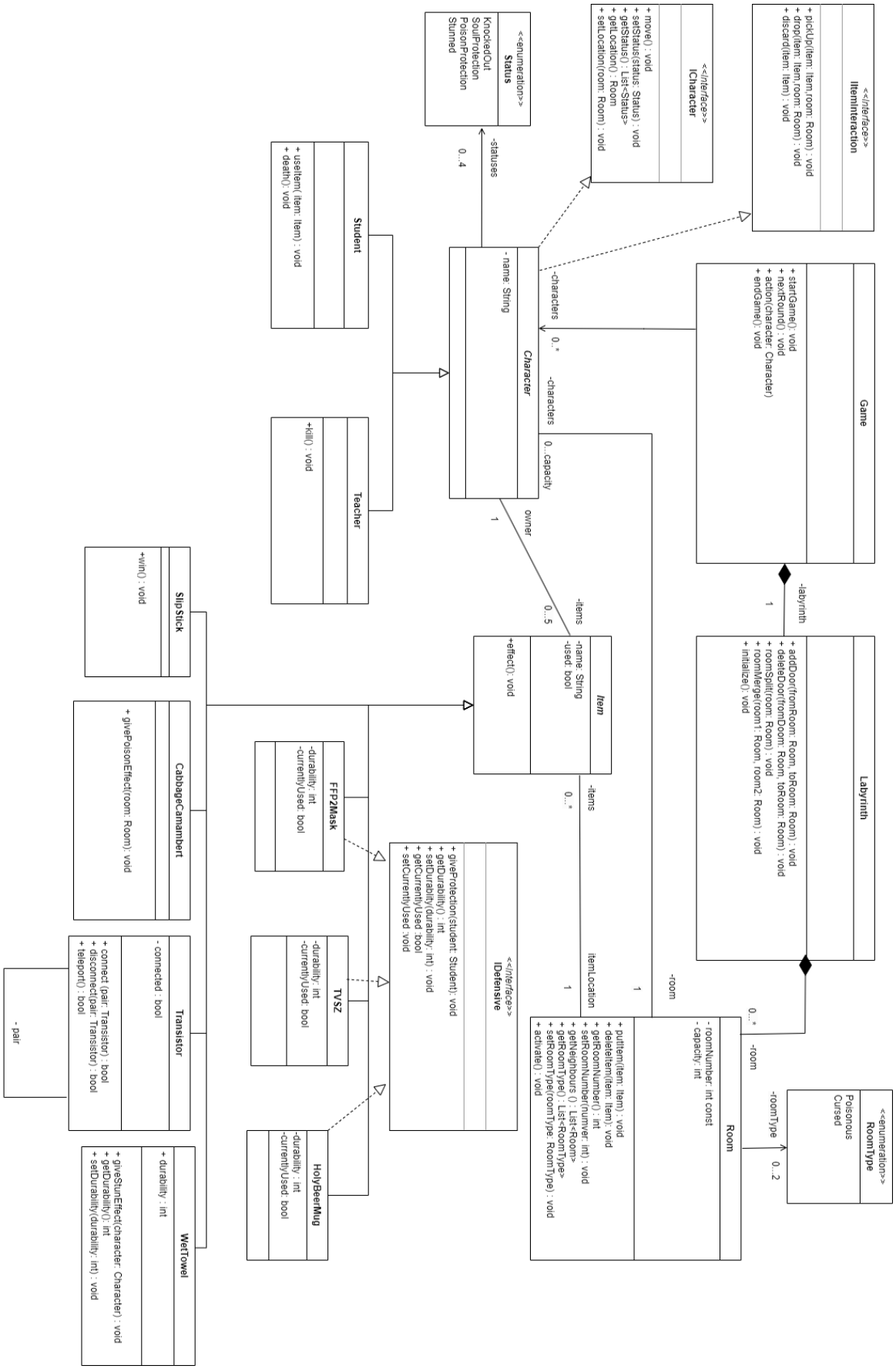
Olyan tárgy, amit a hallgató hatékonyan tud arra használni, hogy megmentse a lelkét attól, hogy azt az oktatók elvegyék. Rendelkezik használati idővel és számon tartja, hogy használatbalett-e már véve, mivel a legelső használatától számítva csak adott ideig működik. A használhatósága körönként mindig csökken, és ha már nem használható, magától elég a karakter eszköztárában.

3.1.12 Játék

A játékmenetet kezeli, vezérli a köröket és vizsgálja, hogy a játék véget ért-e.

Számontartja a játékteret és karaktereket, valamint lehetővé teszi a karakterek számára az akciók végrehajtását.

3.2 Statikus struktúra diagramok



3.3 Osztályok leírása

3.3.1 CabbageCamambert

- **Felelősség**

Az osztály a Dobozos káposztás camembert tárgy viselkedését valósítja meg. Felelőssége, hogy a hallgatók az osztály felhasználásával a szobának a típusát meg tudják változtatni “Poisonous”-re.

- **Ősosztályok**

Nem rendelkezik ősosztállyal.

- **Interfészek**

Az osztály nem valósít meg interfészt.

- **Attribútumok**
- **-name**: a tárgy neve (örökölt attribútum).
- **-used**: azt mutatja, hogy a tárgy használható-e még (örökölt attribútum).
- **-owner:** A tárgyak birtokló játékos (örökölt attribútum).
- **-itemLocation:** Az a szoba amelyben az item található (örökölt attribútum).

- **Asszociációk**
- Nem vesz részt asszociációban.

- **Metódusok**
- **+void givePoisonEffect(Room room):** A metódus a paraméterként kapott szoba típusai közé felveszi a “Poisonous” szobatípust. Amennyiben már eddig szerepelt ez a jelző, akkor nem történik semmi.
- **+void effect() :** A tárgy hatását kifejtő függvény (örökölt metódus).

3.3.2 Character

- **Felelősség**

Ez egy absztrakt osztály, mely az ICharacter és az IItemInteraction interfészeket valósítja meg, illetve a játékben szereplő karaktereknek az absztrakt osztálya. Az interfészekben szereplő függvényeket vagy ő valósítja meg, vagy a leszármazottai.

- **Ősosztályok**

Nem rendelkezik ősosztállyal.

- **Interfészek**

ICharacter, IItemInteraction

- **Attribútumok**
- **-name:** A karakter neve.

- **-location:** Azon szoba amelyben a karakter tartózkodik.
- **-inventory**: A karakternél lévő tárgyak tárolódnak itt.
- **-statuses**: A karakterre érvényben lévő státuszokat tárolja.
- **Asszociációk**
- **egyirányú kapcsolat a Character osztályból a Status osztály felé**: A Character és a Status osztályok közötti kapcsolatot valósítja meg. A karakter tudja magáról milyen státuszai vannak, amelyekből legfeljebb 4 lehet neki.
- **egyirányú kapcsolat a Character osztályból az Item osztály felé**: A Character és az Item osztályok közötti kapcsolatot valósítja meg. A karakter tudja milyen tárgyak vannak az eszköztárában, de egyidőben egy karakternél legfeljebb 5 Item lehet.
- **kétirányú kapcsolat a Room és a Character osztályok közt:** A karakterek tudják melyik szobában vannak, a szobák pedig azt hogy mely karakterek tartózkodnak bennük, és egy karakter egy időben csak egy szobában lehet. Egy szobában egy időben a szoba befogadóképességével megegyező számú karakter lehet.

- **Metódusok**
- **+void move() :** A karakter mozgását valósítja meg(absztrakt).
- **+void setStatus(Status status) :** A karakter státuszát lehet vele állítani(absztrakt).
- **+List<Status> getStatus()** : A karakter státuszát adja vissza(absztrakt).
- **+Room getLocation()** : Visszaadja a karakter mely szobában helyezkedik el.
- **+void setLocation(Room room)** : A karakter tartózkodási helyét lehet vele állítani.
- **+void pickUp(Ttem item, Room room) :** A karakter felveszi a szobában lévő tárgyat, így a tárgy bekerül a karakter eszköztárába.
- **+void drop(Item item, Room room) :** A karakter az eszköztárából abba a szobába teszi le a kiválasztott tárgyat amiben éppen tartózkodik.
- **+void discard(Item item) :** A karakter az eszköztárából elégeti a kiválasztott tárgyat.

3.3.3 FFP2Mask

- **Felelősség**

Az osztály az FFP-es maszk tárgy viselkedését valósítja meg. felelőssége, hogy a hallgatók az osztály felhasználásával a védettséget kapjanak a mérges gáz hatása ellen. A hallgatók status attribútuma változik az osztály felhasználásával.

- **Ősosztályok**

Item

- **Interfészek**

IDefensive

- **Attribútumok**
- **-name**: a tárgy neve (örökölt attribútum).
- **-used**: azt mutatja, hogy a tárgy használható-e még (örökölt attribútum).
- **-owner:** A tárgyak birtokló játékos (örökölt attribútum).
- **-itemLocation:** Az a szoba amelyben az item található (örökölt attribútum).
- **-durability:** használhatóságot jelez, amely hogyha eléri a 0-át, akkor a tárgy használhatatlanná válik.

- **-currentlyUsed:** ezzel az attribútummal vizsgáljuk, hogy az adott tárgy használatban van-e.

- **Asszociációk**
- Nem vesz részt asszociációban.

- **Metódusok**
- **+void giveProtection(Student student):** A metódus a hallgató állapotai közé felveszi az mérges gáz elleni állapotot(PoisonProtection) így a gáz hatása nem érvényes a hallgatóra.
- **+int getDurability():** a “durability” attribútum értékét kapjuk vissza ezen getter metódus által.
- **+void setDurability(int durability):** ezzel a setter metódussal vagyunk képesek a “durability” tagváltozó értéket beállítani.
- **+bool getcurrentlyUsed() :** a “currentlyUsed” attribútum értékét kapjuk vissza ezen getter metódus által.
- **+void setcurrentlyUsed() :** ezzel a setter metódussal vagyunk képesek a “currentlyUsed” tagváltozó kezdőértéket beállítani.
- **+void effect() :** A tárgy hatását kifejtő függvény.(örökölt metódus).

3.3.4 Game

- **Felelősség**

Az osztály a játék indításáért és a körök menedzseléséért felelős, illetve lehetővé teszi a karakterek számára, hogy akciót hajtsanak végre a körön belül (azaz “sorra kerüljenek”). A játék végének kezelése is az osztály felelőssége.

- **Ősosztályok**

Nem rendelkezik ősosztállyal.

- **Interfészek**

Az osztály nem valósít meg interfészt.

- **Attribútumok**
- **-characters:** a játékban részt vevő karakterek
- **+labyrinth:** a játétérként szolgáló labirintus

- **Asszociációk**
- **egyirányú kapcsolat a Game osztályból a Character osztály felé:** a Game osztály számon tartja a játékban részt vevő karaktereket.

- **Metódusok**
- **+void startGame():** játék indítása, kezdőállapot betöltése.
- **+void nextRound():** következő kör indításának megvalósítása.
- **+void endGame():** a játék végének kezelése.
- **+void action(Character character):** a karakterek akciójának megvalósítása.

3.3.5 HolyBeerMug

- **Felelősség**

Az osztály a Szent söröspohár tárgy viselkedését valósítja meg. Felelőssége, hogy ha egy hallgató használja ezt tárgyat, akkor az megvédje őt egy adott ideig attól, hogy egy oktató kiszívja a lelkét.

- **Ősosztályok**

Item

- **Interfészek**

IDefensive

- **Attribútumok**
- **-name**: a tárgy neve (örökölt attribútum).
- **-used**: azt mutatja, hogy a tárgy használható-e még (örökölt attribútum).
- **-owner:** A tárgyak birtokló játékos (örökölt attribútum).
- **-itemLocation:** Az a szoba amelyben az item található (örökölt attribútum).
- **-durability:** használhatóságot jelez, amely hogyha eléri a 0-át, akkor a tárgy használhatatlanná válik.

- **-currentlyUsed:** ezzel az attribútummal vizsgáljuk, hogy az adott tárgy használatban van-e.

- **Asszociációk**
- Nem vesz részt asszociációban.

- **Metódusok**
- **+void giveProtection(Student student):** A metódus a hallgató állapotai közé felveszi az oktató lélek kiszívása elleni állapotot (SoulProtection) így az oktatók támadásai nem hatnak a hallgatóra.
- **+int getDurability():** a “durability” attribútum értékét kapjuk vissza ezen getter metódus által.
- **+void setDurability(int durability):** ezzel a setter metódussal vagyunk képesek a “durability” tagváltozó értéket beállítani.
- **+bool getcurrentlyUsed() :** a “currentlyUsed” attribútum értékét kapjuk vissza ezen getter metódus által.
- **+void setcurrentlyUsed(bool currentlyUsed) :** ezzel a setter metódussal vagyunk képesek a “currentlyUsed” tagváltozó kezdőértéket beállítani.
- **+void effect() :** A tárgy hatását kifejtő függvény.(örökölt metódus).

3.3.6 ICharacter

- **Felelősség**

A játékban részt vevő karaktereknek mozgásához, illetve állapotuk lekérdezéséhez biztosít függvényeket.

- **Ősosztályok**

Nem rendelkezik ősosztállyal.

- **Asszociációk**

Nem vesz részt asszociációban.

- **Metódusok**
- **+void move() :** A karakter mozgását valósítja meg(absztrakt).
- **+void setStatus(Status status)** : A karakter státuszát lehet vele állítani(absztrakt).
- **+List<Status> getStatus()** : A karakter státuszát adja vissza(absztrakt).
- **+Room getLocation()** : Visszaadja a karakter mely szobában helyezkedik el.
- **+void setLocation(Room room)** : A karakter tartózkodási helyét lehet vele állítani.

3.3.7 IDefensive

- **Felelősség**

A védettséget adó tárgyaknak működéséhez, állapotuk lekérdezéséhez biztosít függvényeket.

- **Ősosztályok**

Nem rendelkezik ősosztállyal.

- **Asszociációk**

Nem vesz részt asszociációban.

- **Metódusok**
- **+void giveProtection(Student student):** A metódus a hallgató állapotai közé felveszi a tárgy által meghatározott állapotot, így védelmet ad neki.
- **+int getDurability():** a “durability” attribútum értékét kapjuk vissza ezen getter metódus által.
- **+void setDurability(int durability):** ezzel a setter metódussal vagyunk képesek a “durability” tagváltozó értéket beállítani.
- **+bool getcurrentlyUsed() :** a “currentlyUsed” attribútum értékét kapjuk vissza ezen getter metódus által.
- **+void setcurrentlyUsed(bool currentlyUsed) :** ezzel a setter metódussal vagyunk képesek a “currentlyUsed” tagváltozó kezdőértéket beállítani.

3.3.8 IItemInteraction

- **Felelősség**

A tárgyakkal való játék működéshez biztosít függvényeket, ilyen a tárgy felvétele, letétele és égetése.

- **Ősosztályok**

Nem rendelkezik ősosztállyal.

- **Asszociációk**

Nem vesz részt asszociációban.

- **Metódusok**
- **+void pickUp(Item item, Room room) :** A függvényt megvalósító osztály felveszi a szobában lévő tárgyat, így a tárgy bekerül az osztály eszköztárába.
- **+void drop(Item item, Room room) :** A függvényt megvalósító osztály az eszköztárából a szobába teszi le a kiválasztott tárgyat.
- **+void discard(Item item) :** A függvényt megvalósító osztály az eszköztárából elégeti a kiválasztott tárgyat.

3.3.9 Item

- **Felelősség**

A játékban szereplő tárgyaknak egy absztrakt osztályt biztosít. A tárgyak hatáskifejtéséért is felelős.

- **Ősosztályok**

Nem rendelkezik ősosztállyal.

- **Interfészek**

Az osztály nem valósít meg interfészt.

- **Attribútumok**
- **-name:** A tárgy neve.
- **-used:** Azt jelzi hogy a tárgyat elhasználták-e már.
- **-owner:** A tárgyak birtokló játékos
- **-itemLocation:** Az a szoba amelyben az item található

- **Asszociációk**
- **egyirányú kapcsolat az Character osztályból az Item osztály felé**: A karakter tudja milyen tárgyak vannak az eszköztárában, de egyidőben egy karakternél legfeljebb 5 Item lehet.
- **egyirányú kapcsolat a Room osztályból az Item osztály felé:** A szobában elérhetőek a benne lévő tárgyak. Egy szobában akármennyi tárgy lehet.

- **Metódusok**

- **+void effect() :** A tárgy hatását kifejtő függvény (absztrakt).

3.3.10 Labyrinth

- **Felelősség**

A szobák és azok szomszédainak nyilvántartásáért és az esetleges szoba változások kezelésért (ajtó megjelenés, ajtó eltűnés, szoba osztódás, szoba egyesülés) felelős.

- **Ősosztályok**

Nem rendelkezik ősosztállyal.

- **Interfészek**

Az osztály nem valósít meg interfészt.

- **Attribútumok**
- **-playground :** A szobákat és szomszédait tartja nyilván.

- **Asszociációk**

Nem vesz részt asszociációban.

- **Metódusok**
- **+void addDoor(Room fromRoom, Room toRoom) :** Kezeli azt az esetet amikor a szobának új ajtaja keletkezik(pl osztódáskor az osztódó szobák egymásba nyílnak), vagy egy régi jelenik meg.
- **+void deleteDoor(Room fromRoom, Room toRoom) :** Kezeli azt az esetet amikor a szobának ajtaja tűnik el.
- **+void roomSplit(Room room) :** Két szoba osztódását valósítja meg.
- **+void roomMerge(Room room, Room room) :** Két szoba egyesülését valósítja meg.
- **+void initialie():** A labirintus felépítését, a kezdeti állapotának beállítását valósítja meg.

3.3.11 Room

- **Felelősség**

A labirintust felépítő szobák és azok viselkedésének megvalósításáért felelős.

- **Ősosztályok**

Nem rendelkezik ősosztállyal.

- **Interfészek**

Az osztály nem valósít meg interfészt.

- **Attribútumok**
- **-roomNumber:** A szobaszám, amely minden szobának egyedi.
- **-capacity:** A szobában tartózkodó karakterek maximális számát adja meg.
- **-roomType**: A szoba képességet meghatározó típusát/típusait tárolja.
- **-characters**: A szobában lévő karaktereket tartalmazza.
- **-items**: A szobában lévő tárgyakat tartalmazza.

- **Asszociáció**
- **egyirányú kapcsolat a Room osztályból a RoomType osztály felé**: A szoba tudja magáról milyen képességet adó típusa/típusai vannak, amelyekből legfeljebb 2 van neki.
- **egyirányú kapcsolat a Room osztályból az Item osztály felé**: A szobaban elérhetőek a benne lévő tárgyak. Egy szobában akár mennyi tárgy lehet.
- **kétirányú kapcsolat a Room és a Character osztályok közt:** A karakterek tudják melyik szobában vannak, a szobák pedig azt hogy mely karakterek tartózkodnak bennük. Egy karakter egy időben egy szobában tartózhat. Egy szobában egy időben egyszerre a szoba befogadóképességével megegyező karakterek lehet.
- **kompozíció a Room osztályból a Labyrinth osztály a felé:** A labirintus szobákból áll, egy szoba egy labirintushoz tartozhat. Ha a labirintus megsemmisül akkor a tartalmazott szobák is megszűnnek. Egy labirintus akárhány szobából állhat.

- **Metódusok**

- **+void putItem(item: Item) :** A szobában új tárgy helyeződik el.

- **+void deleteItem(item: Item):** A szobából egy - a szobában lévő- tárgyat felvettek a karakterek.
- **+int getRoomNumber() :** A szoba szobaszámát adja vissza.
- **+void setRoomNumber(numver: int) :** A szoba szobaszámát állítja be.
- **+List<Room> getNeighbours () :** Visszaadja a szoba szomszédait.
- **+List<RoomType> getRoomType() :** Visszaadja a szoba típusát/típusait.
- **+void setRoomType(roomType: RoomType) :** A szoba típusa állítható vele.
- **+void activate() :** A szoba a típusából / típusaiból következő extra hatását érvényesíti.

3.3.12 RoomType

- **Felelősség**

Olyan enumerációs osztály, melynek felelőssége az, hogy a segítségével a szobák típusait nyilván tudjuk tartani.

- **Ősosztályok**

Nem rendelkezik ősosztállyal.

- **Interfészek**

Az osztály nem valósít meg interfészt.

- **Attribútumok**
- **Poisonous:** azt jelenti, hogy az adott szoba mérges gázzal teli, és ha egy karakter belép egy ilyen típusú szobába, akkor elveszti az eszméletét.
- **Cursed:** azt jelenti, hogy a szoba elátkozott, emiatt bizonyos időegységenként eltűnik vagy megjelenik egy ajtaja.

- **Asszociáció**
- **egyirányú kapcsolat a Room osztályból a RoomType osztály felé**:A szoba tudja magáról milyen képességet adó típusa/típusai vannak.Egy szobának egy időben egyszerre 2 típusa lehet.

3.3.13 SlipStick

- **Felelősség**

Az osztály a Logarléc tárgy viselkedését valósítja meg. Felelőssége, hogy ha a hallgatók eszköztárába kerül a Logarléc nevű tárgy, akkor sikeresen teljesítették a küldetésüket és megnyerték a játékot.

- **Ősosztályok**

Item

- **Interfészek**

Az osztály nem valósít meg interfészt.

- **Attribútumok**
- **-name**: a tárgy neve (örökölt attribútum).
- **-used:** Azt jelzi hogy a tárgyat elhasználták-e már.(örökölt attribútum)
- **-owner:** A tárgyak birtokló játékos (örökölt attribútum).
- **-itemLocation:** Az a szoba amelyben az item található (örökölt attribútum).

- **Asszociációk**

Nem vesz részt asszociációban.

- **Metódusok**
- **+void effect() :** A tárgy hatását kifejtő metódus (örökölt metódus).
- **+void win():** Ez a metódus akkor fut le, ha egy hallgató eszköztárába kerül a Logarléc nevezetű tárgy, és ennek hatására megnyerik a hallgatók a játékot.

3.3.14 Status

- **Felelősség**

Olyan enumerációs osztály, melynek felelőssége az, hogy a segítségével a karakterek státuszait nyilván tudjuk tartani.

- **Ősosztályok**

Nem rendelkezik ősosztállyal.

- **Interfészek**

Nem valósít meg interfészt.

- **Attribútumok**
- **KnockedOut:** azt jelenti, hogy az adott karakter eszméletét vesztette, és ennek a következtében elejti minden tárgyát és bizonyos időegységig képtelen akciót végrehajtani.
- **SoulProtection:** azt jelenti, hogy egy oktató képtelen annak a hallgatónak a lelkét kiszívni, aki ilyen státusszal rendelkezik.
- **PoisonProtection:** azt jelenti, hogy az adott karakter, aki ezzel a státusszal rendelkezik, nem veszti el az eszméletét, amikor belép egy mérges gázzal teli szobába.
- **Stunned:** azt jelenti, hogy az adott karakter lebénult, és emiatt nem képes bizonyos időegységig akciót végrehajtani.

- **Asszociációk**
- **egyirányú kapcsolat a Character osztályból a Status osztály felé**: A Character és a Status osztályok közötti kapcsolatot valósítja meg. A karakter tudja magáról milyen státuszai vannak, amelyből legfeljebb 4 van neki.

3.3.14 Student

- **Felelősség**

Ez az osztály felel a hallgatók viselkedésének a megvalósításáért.

- **Ősosztályok**

Character

- **Interfészek**

ICharacter, IItemInteraction

- **Attribútumok**
- **-name**: a hallgató neve (örökölt attribútum).

- **Asszociációk**

Nem vesz részt asszociációban.

- **Metódusok**
- **+void move() :** A hallgató mozgását megvalósító metódus.
- **+void setStatus(status: Status)** : A hallgató státuszát lehet beállítani vele, valamilyen esemény után.
- **+List<Status> getStatus()** : A hallgató státuszait adja vissza.
- **+Room getLocation()** : Visszaadja azt, hogy a hallgató mely szobában helyezkedik el.
- **+void setLocation(Room room)** : A hallgató tartózkodási helyét lehet vele állítani.
- **+void pickUp(Item item, Room room) :** A függvény segítségével képes egy hallgató felvenni a szobában lévő tárgyat, így a tárgy bekerül az eszköztárába.
- **+void drop(Item item, Room room) :** A függvény segítségével képes egy hallgató, az eszköztárából a szobába letenni a kiválasztott tárgyat.
- **+void discard(Item item, Room room) :** A függvény segítségével képes egy hallgató, az eszköztárából elégetni a kiválasztott tárgyat.
- **+void death():** Ha a hallgatónak a lelkét kiszívják, akkor ez a metódus felel a hallgató játékból való kieséséért.
- **+void useItem(Item item):** Ennek a függvénynek a segítségével képes egy hallgató, az eszköztárában szereplő tárgyak közül egyet használni.

3.3.15 Teacher

- **Felelősség**

Ez az osztály felel az oktatók viselkedésének a megvalósításáért.

- **Ősosztályok**

Character

- **Interfészek**

ICharacter, IItemInteraction

- **Attribútumok**
- **-name**: az oktató neve (örökölt attribútum).

- **Asszociációk**

Nem vesz részt asszociációban.

- **Metódusok**
- **+void move() :** Az oktató mozgását megvalósító metódus.
- **+void setStatus(Status status)** : Az oktató státuszát lehet beállítani vele, valamilyen esemény után.
- **+List<Status> getStatus()** : Az oktató státuszait adja vissza.
- **+Room getLocation()** : Visszaadja azt, hogy az oktató mely szobában helyezkedik el.
- **+void setLocation(Room room)** : Az oktató tartózkodási helyét lehet vele állítani.

- **+void kill():** Ezzel a metódussal képes egy oktató egy hallgató lelkét kiszívni, ha ugyanabban a szobában tartózkodnak, és ha a hallgató nem rendelkezik megfelelő védelemmel.
- **+void pickUp(Item item, Room room) :** A függvény segítségével képes egy oktató felvenni a szobában lévő tárgyat, így a tárgy bekerül az oktató eszköztárába.
- **+void drop(Item item, Room room) :** A függvény segítségével képes egy oktató, az eszköztárából a szobába letenni a paraméterben szereplő tárgyat.
- **+void discard(Item item, Room room) :** A függvény segítségével képes egy oktató, az eszköztárából elégetni a paraméterben szereplő tárgyat.

3.3.16 Transistor

- **Felelősség**

Az osztály a Tranzisztor tárgy viselkedését valósítja meg. Felelőssége, hogy biztosítja azon hallgatók számára a teleportálást, akiknek van két összekötött tranzisztoruk és ezek közül az egyik már azon szobába le van helyezve, ahová a hallgató teleportálni szeretne.

- **Ősosztályok**

Item

- **Interfészek**

Az osztály nem valósít meg interfészt.

- **Attribútumok**
- **-connected:** ezzel az attribútummal jelezzük, hogy egy tranzisztor össze van-e már kötve egy másikkal.
- **-name:** a tárgy neve (örökölt attribútum).
- **-used:** azt mutatja, hogy a tárgy használható-e még (örökölt attribútum).
- **-owner:** A tárgyak birtokló játékos (örökölt attribútum).
- **-itemLocation:** Az a szoba amelyben az item található (örökölt attribútum).
- **-pair:** összekapcsolt tranzisztor esetén a tranzisztor párja (egy másik tranzisztor)

- **Asszociációk**
- **asszociáció a Transistor osztállyal:** összekapcsolt tranzisztorok esetén a tranzisztor párja egy másik tranzisztor. Ezt a kapcsolatot fejezi ki ez az asszociáció.

- **Metódusok**
- **+bool connect(Transistor pair):** ezzel a függvénnyel kapcsoljuk össze a tranzisztorunkat a paraméterben szereplővel, és a függvény visszatérési értékéből tudjuk meg, hogy ez a művelet sikeres volt-e vagy sem.
- **+bool disconnect(Transistor pair):** ezzel a függvénnyel szétkapcsoljuk a tranzisztorunkat a paraméterben szereplőtől, és a függvény visszatérési értékéből tudjuk meg, hogy ez a művelet sikeres volt-e vagy sem.

- **+bool teleport():** ezen metódussal történik meg a hallgató elteleportálása a már letett és összekapcsolt tranzisztorhoz, és szintén a metódus visszatérési értékéből tudjuk meg, hogy ez a művelet sikeres volt-e vagy sem.
- **+void effect() :** A tárgy hatását kifejtő függvény (örökölt metódus).

3.3.17 TVSZ

- **Felelősség**

Az osztály a TVSZ denevérbőrre nyomtatott példánya tárgy viselkedését valósítja meg. Felelőssége, hogy ha egy hallgató használja a "TVSZ denevérbőrre nyomtatott példánya" nevű tárgyat, akkor az megvédje őt 3-szor attól, hogy egy oktató kiszívja a lelkét.

- **Ősosztályok**

Item

- **Interfészek**

IDefensive

- **Attribútumok**
- **-durability:** használhatóságot jelez, amely hogyha eléri a 0-át, akkor a tárgy használhatatlanná válik.
- **-currentlyUsed:** ezzel az attribútummal vizsgáljuk, hogy az adott tárgy használatban van-e.
- **-name**: a tárgy neve (örökölt attribútum).
- **-used**: azt mutatja, hogy a tárgy használható-e még (örökölt attribútum).
- **-owner:** A tárgyak birtokló játékos (örökölt attribútum).
- **-itemLocation:** Az a szoba amelyben az item található (örökölt attribútum).

- **Asszociációk**

Nem vesz részt asszociációban.

- **Metódusok**
- + **void giveProtection(Hallgato hallgato):**A metódus a hallgató állapotai közé felveszi az oktató lélekkiszívása elleni állapotot(SoulProtection) így az oktatók támadásai nem hatnak a hallgatóra.
- + **int getDurability():** a "durability" attribútum értékét kapjuk vissza ezen getter metódus által.
- **+ void setDurability(int durability):** ezzel a setter metódussal vagyunk képesek a "durability" tagváltozó értéket beállítani.
- + **bool getcurrentlyUsed():** a "currentlyUsed" attribútum értékét kapjuk vissza ezen getter metódus által.
- + **void setcurrentlyUsed(int currentlyUsed):** ezzel a setter metódussal vagyunk képesek a "currentlyUsed" tagváltozó értéket beállítani.

- **+void effect() :** A tárgy hatását kifejtő metódus (örökölt metódus).

3.3.18 WetTowel

- **Felelősség**

Az osztály a Nedves táblatörlő rongy tárgy viselkedését valósítja meg. Felelőssége, hogy a hallgató a használata során képes legyen egy oktatót megbénítani, és ez által kisebb előnyre tegyen szert a játék során.

- **Ősosztályok**

Item

- **Interfészek**

Az osztály nem valósít meg interfészt.

- **Attribútumok**
- **-durability:** használhatóságot jelez, amely hogyha eléri a 0-át, akkor a tárgy használhatatlanná válik.
- **-name**: a tárgy neve (örökölt attribútum).
- **-used**: azt mutatja, hogy a tárgy használható-e még (örökölt attribútum).
- **-owner:** A tárgyak birtokló játékos (örökölt attribútum).
- **-itemLocation:** Az a szoba amelyben az item található (örökölt attribútum).

- **Asszociációk**

Nem vesz részt asszociációban.

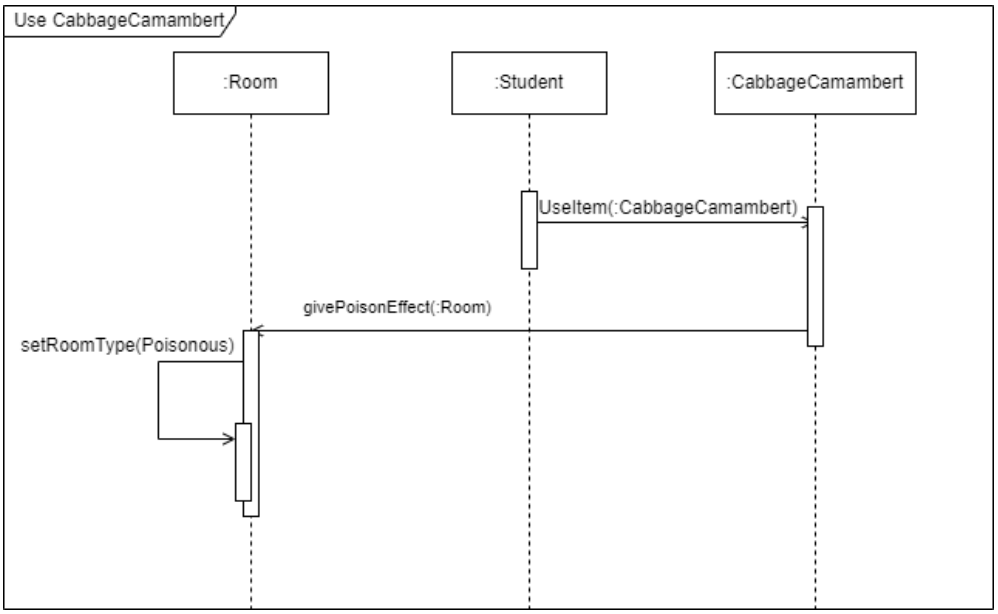
- **Metódusok**
- **+void giveStunEffect(Character character):** ennek a függvénynek a segítségével állítjuk be azon character “Status”-át “Stunned”-ra, amely a paraméterben szerepel.

- **+int getDurability():** a “durability” attribútum értékét kapjuk vissza ezen getter metódus által.

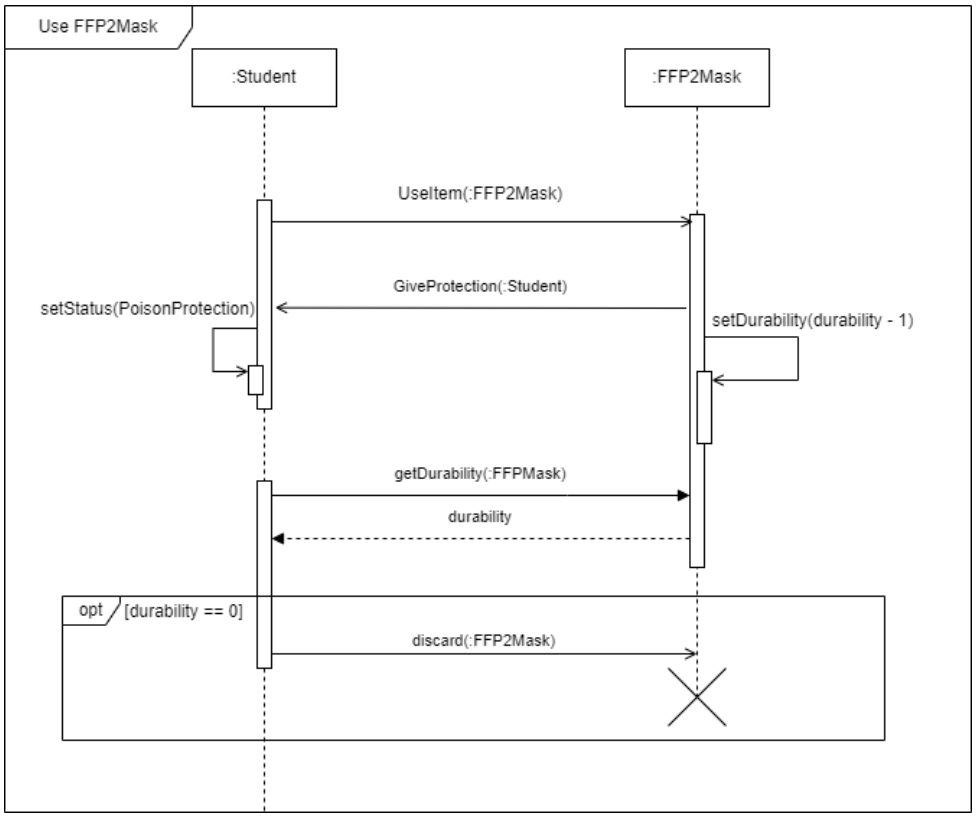
- **+void setDurability(int durability):** ezzel a setter metódussal vagyunk képesek a “durability” tagváltozó értékét beállítani.
- **+void effect() :** A tárgy hatását kifejtő függvény (örökölt metódus).

3.4 Szekvencia diagramok

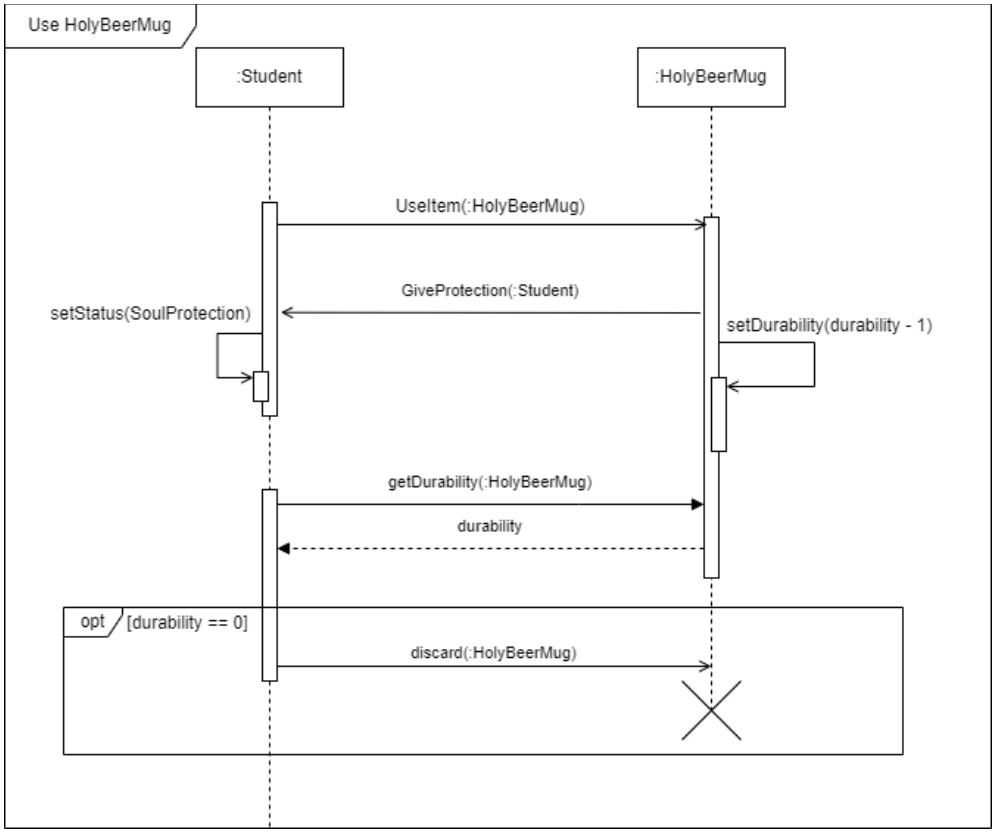
3.4.1 Use CabbageCamambert



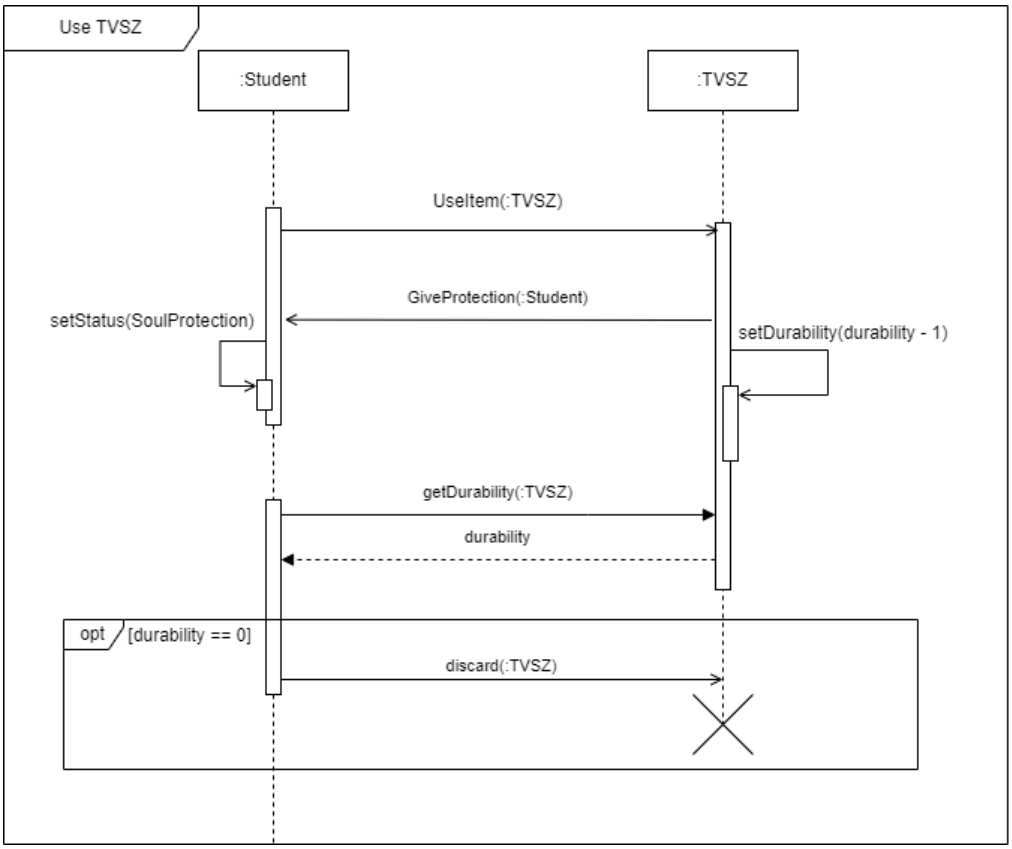
3.4.2 Use FFP2Mask



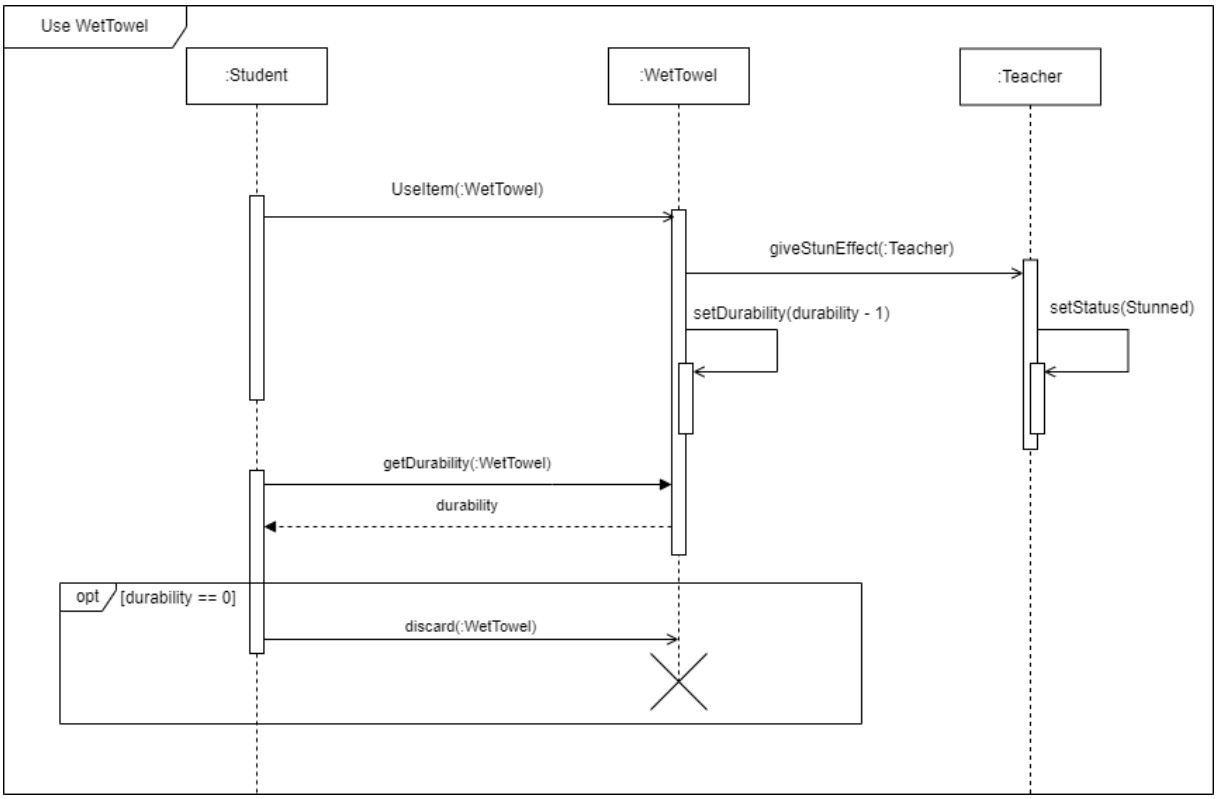
3.4.3 Use HolyBeerMug



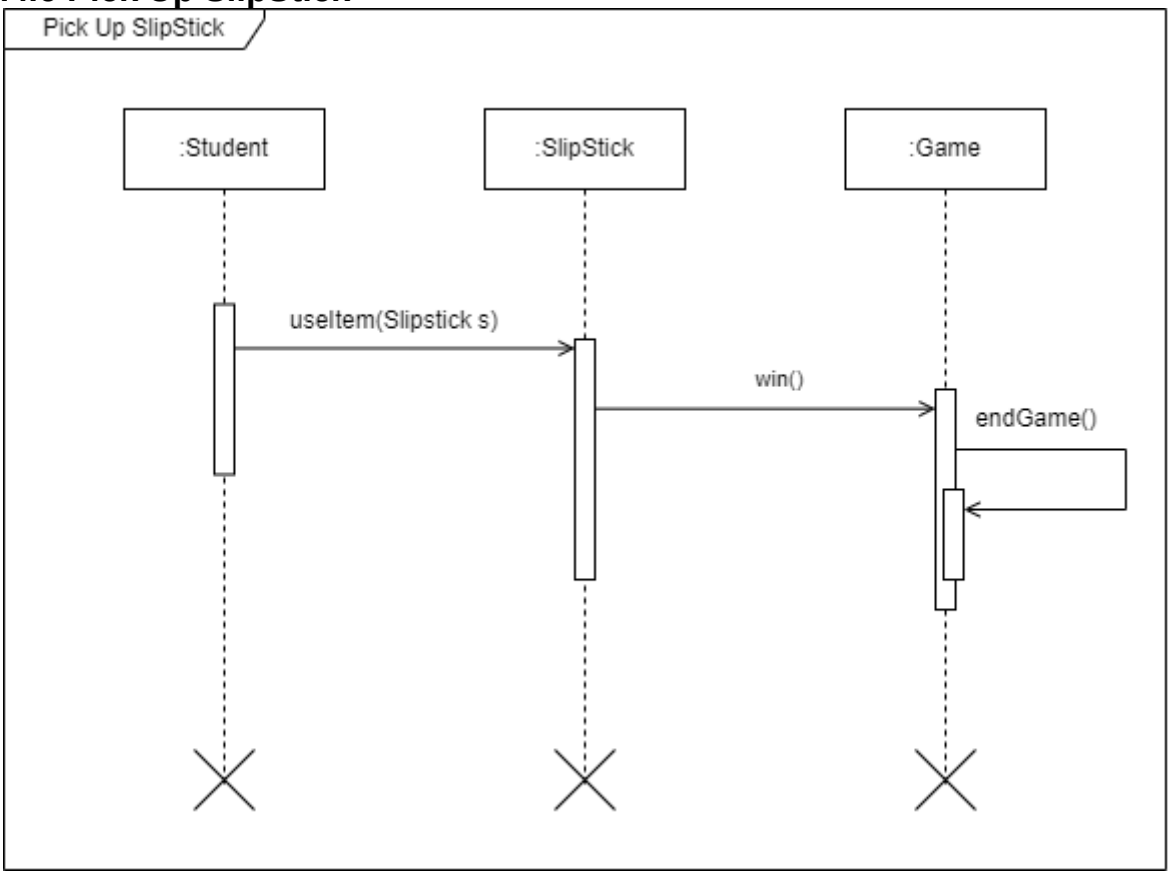
3.4.4 Use TVSZ



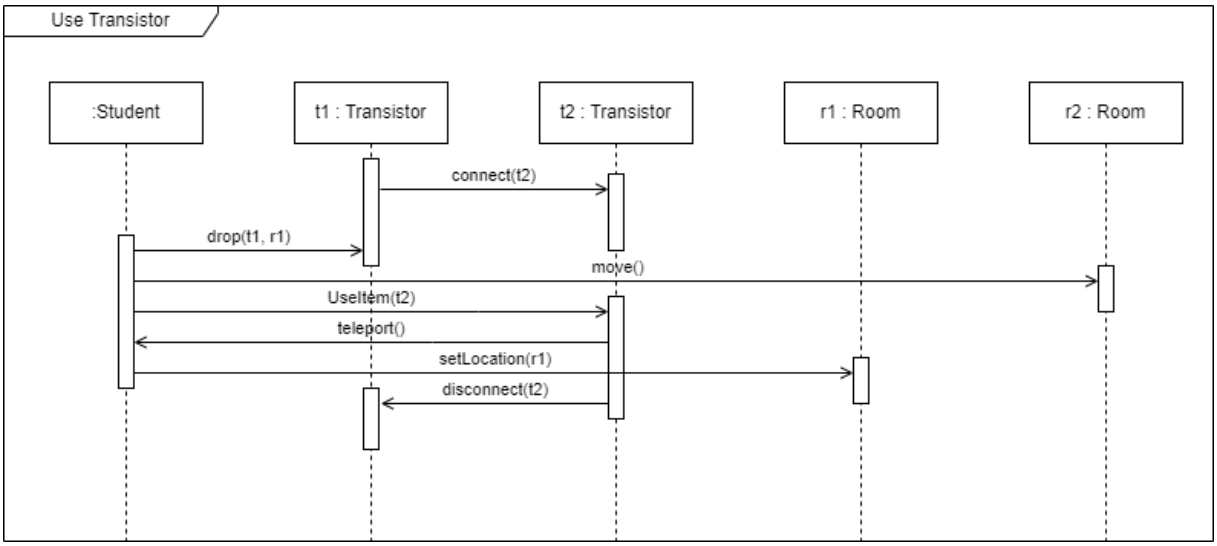
3.4.5 Use WetTowel



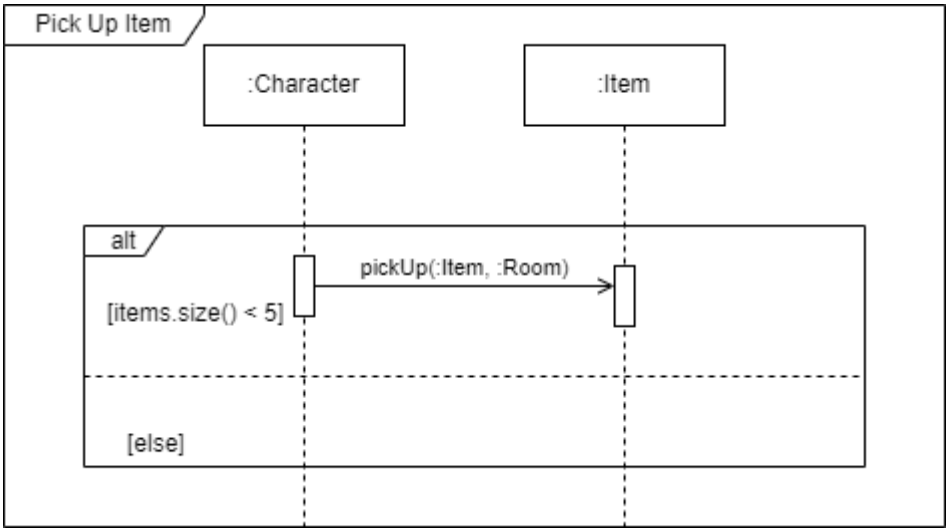
3.4.6 Pick Up SlipStick



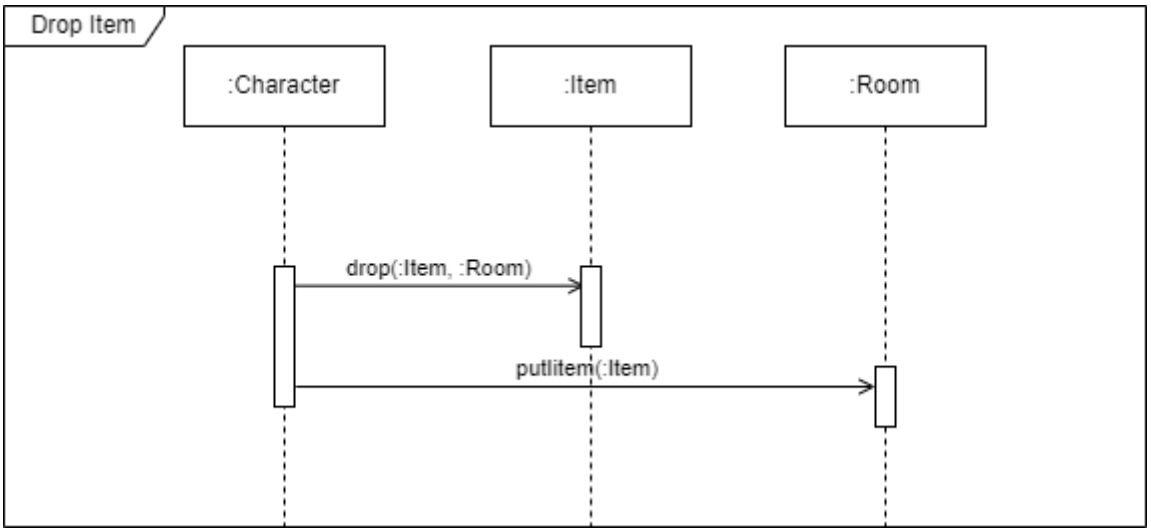
3.4.7 Use Transistor



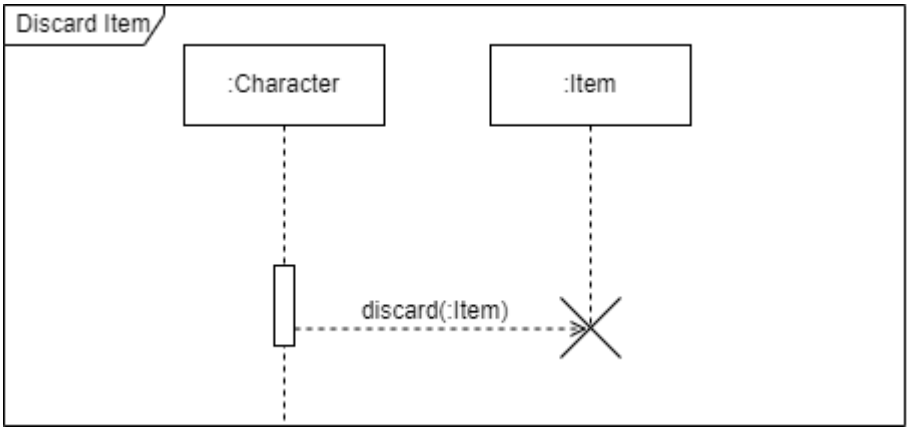
3.4.8 Pick Up Item



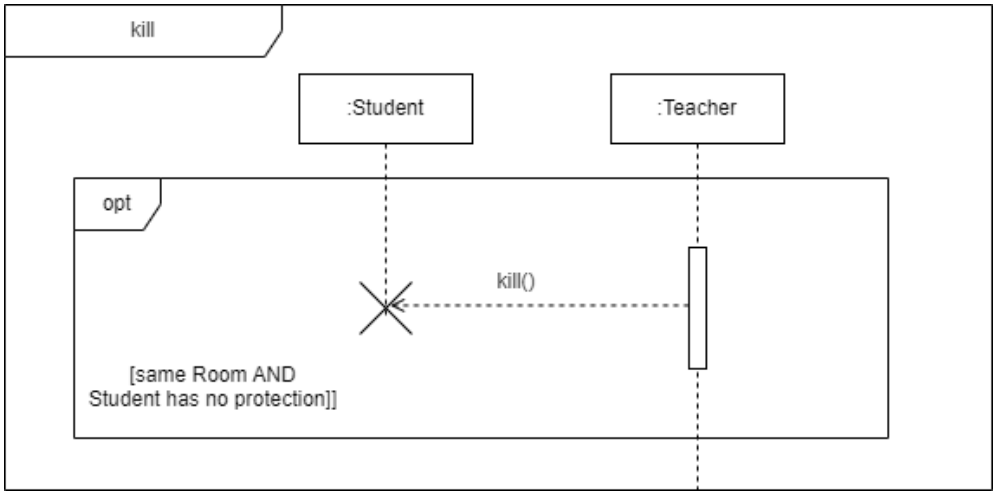
3.4.9 Drop Item



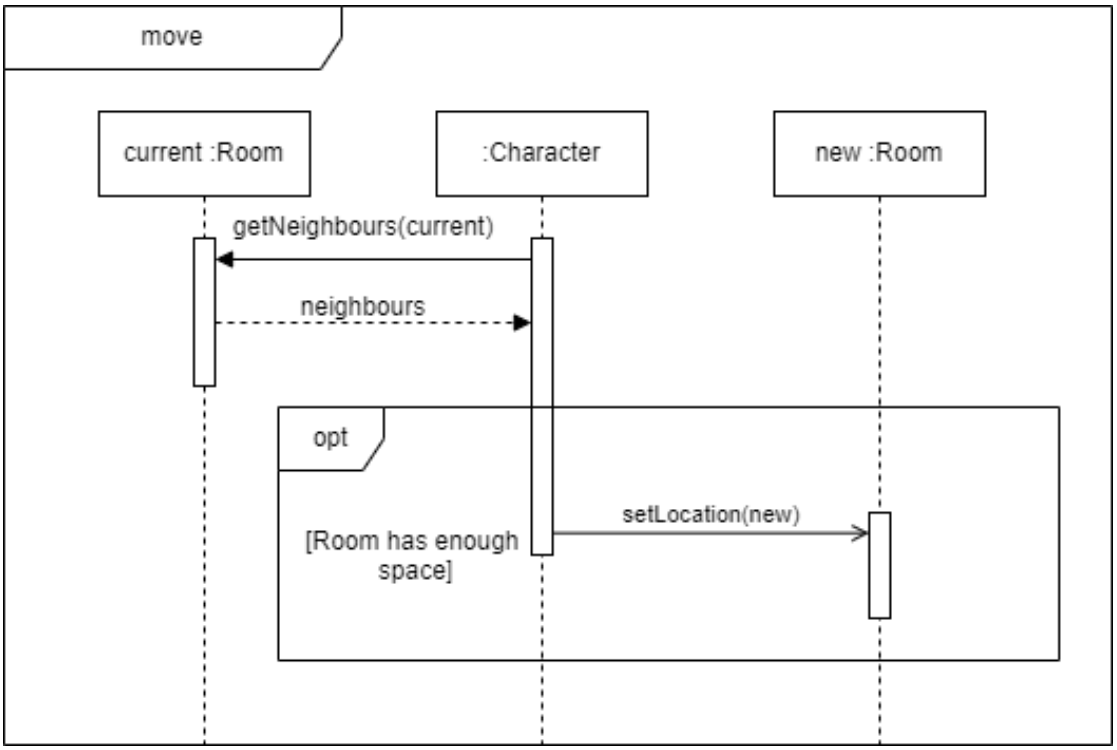
3.4.10 Discard Item



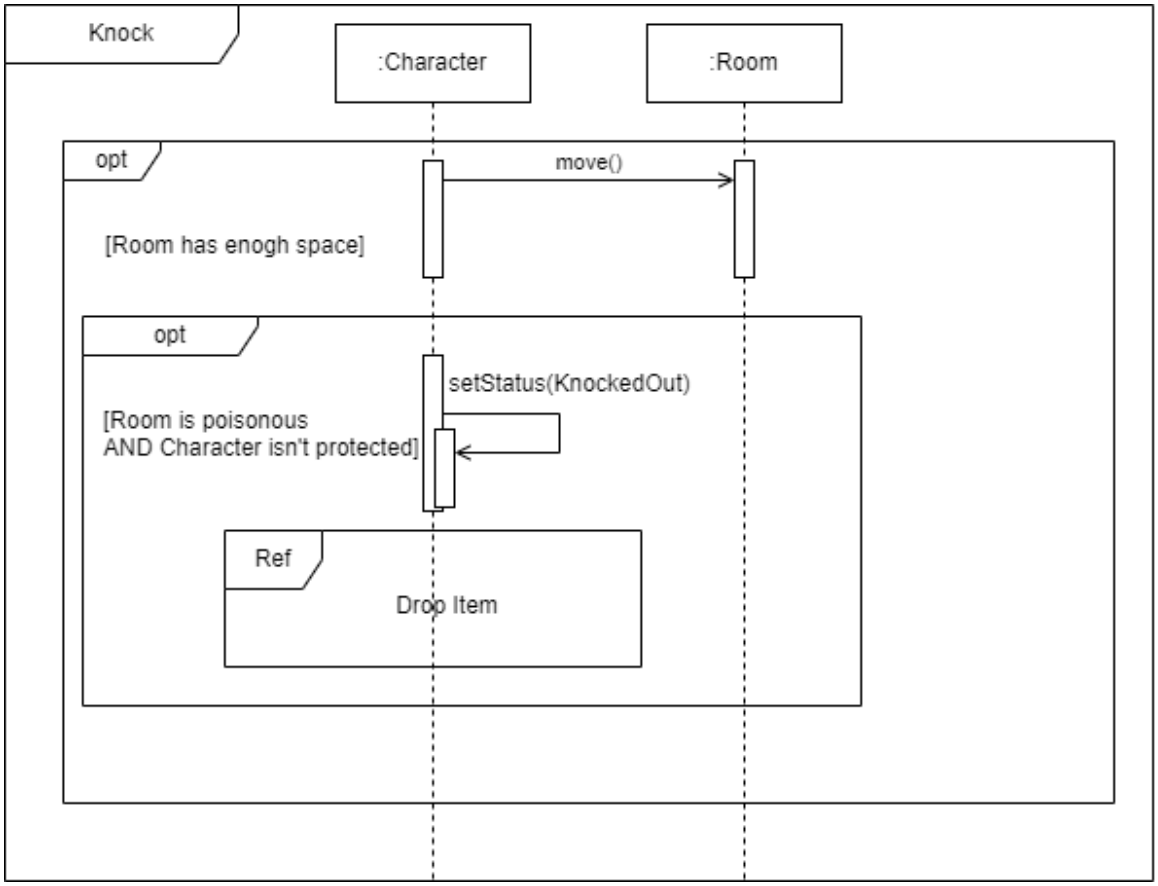
3.4.11 Kill Student



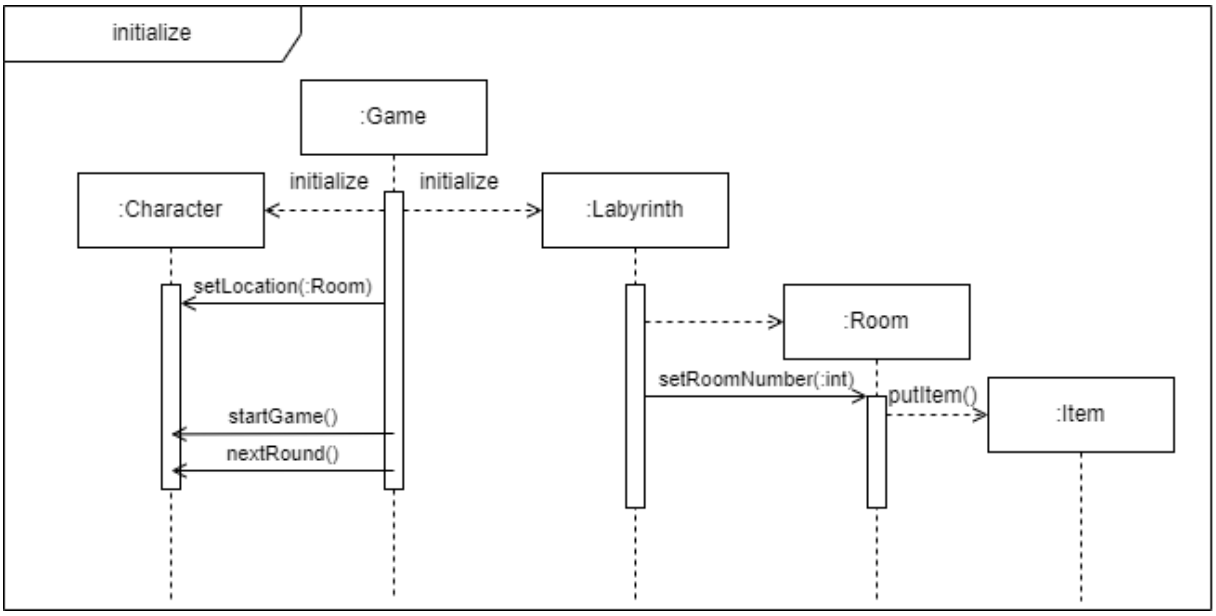
3.4.12 Move



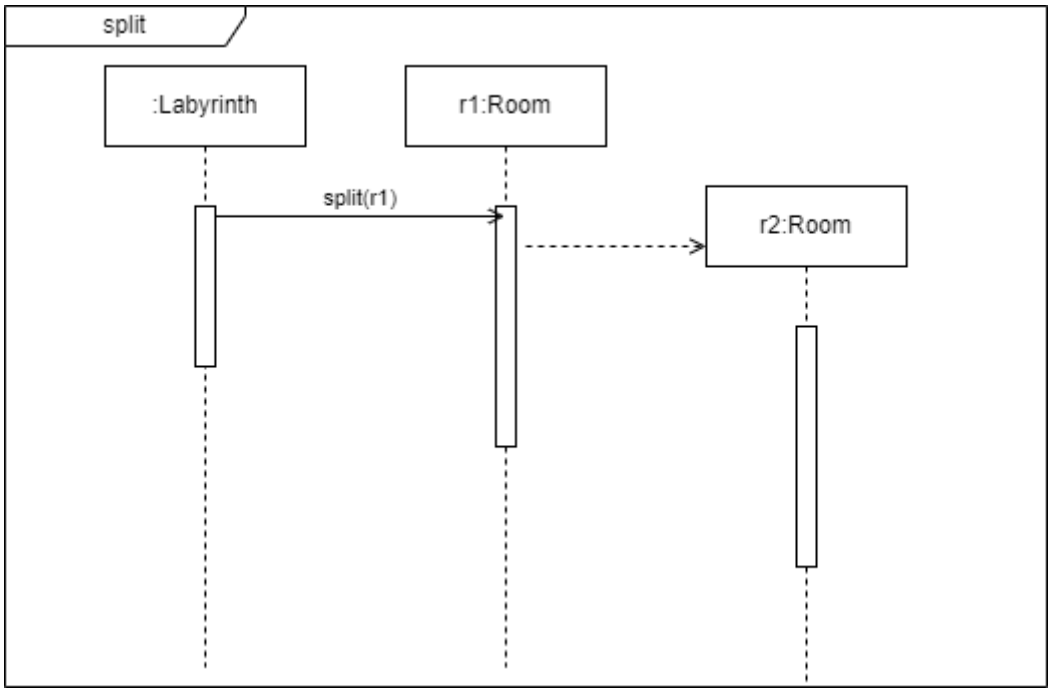
3.4.13 Knock



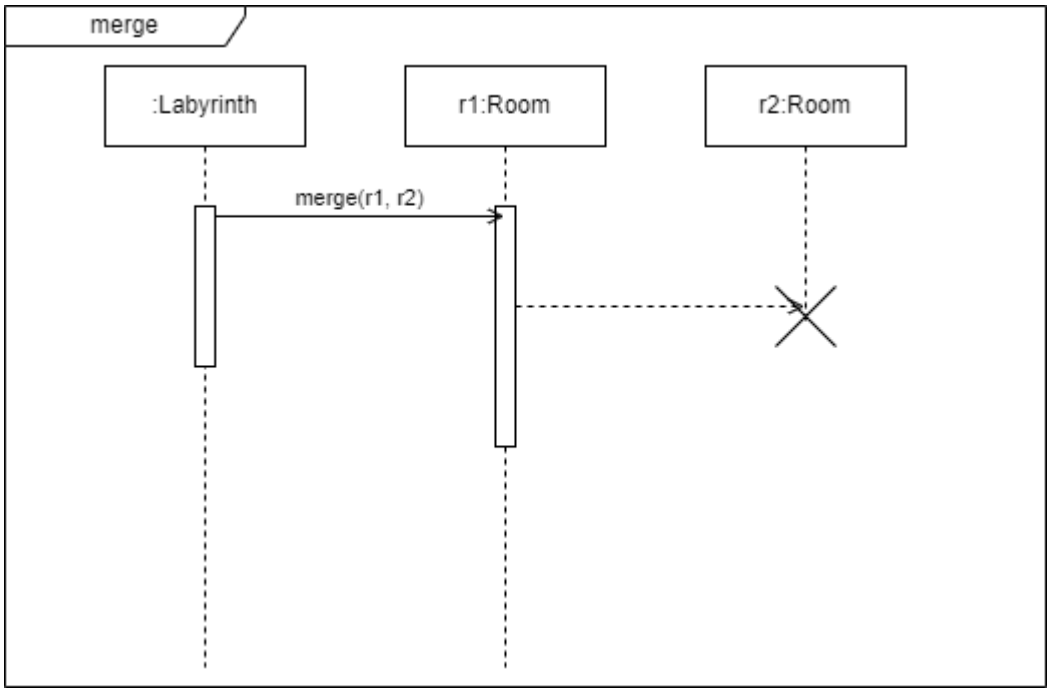
3.4.14 Initialize



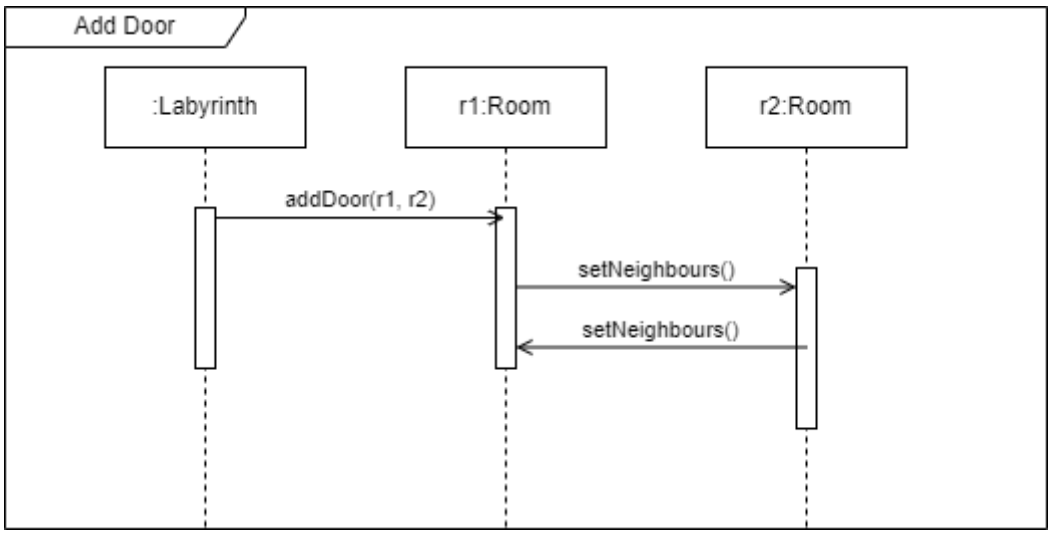
3.4.15 Split Rooms



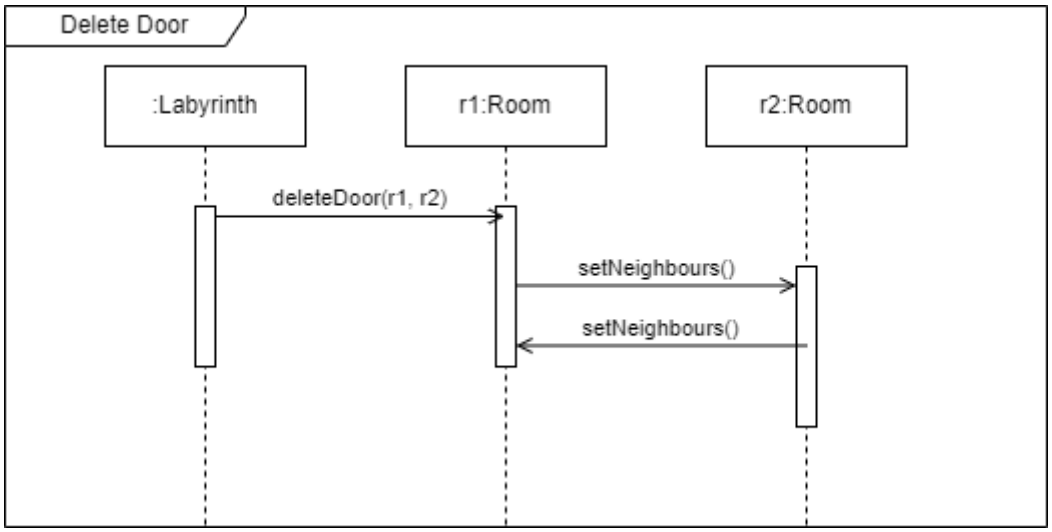
3.4.16 Merge Rooms



3.4.17 Add Door

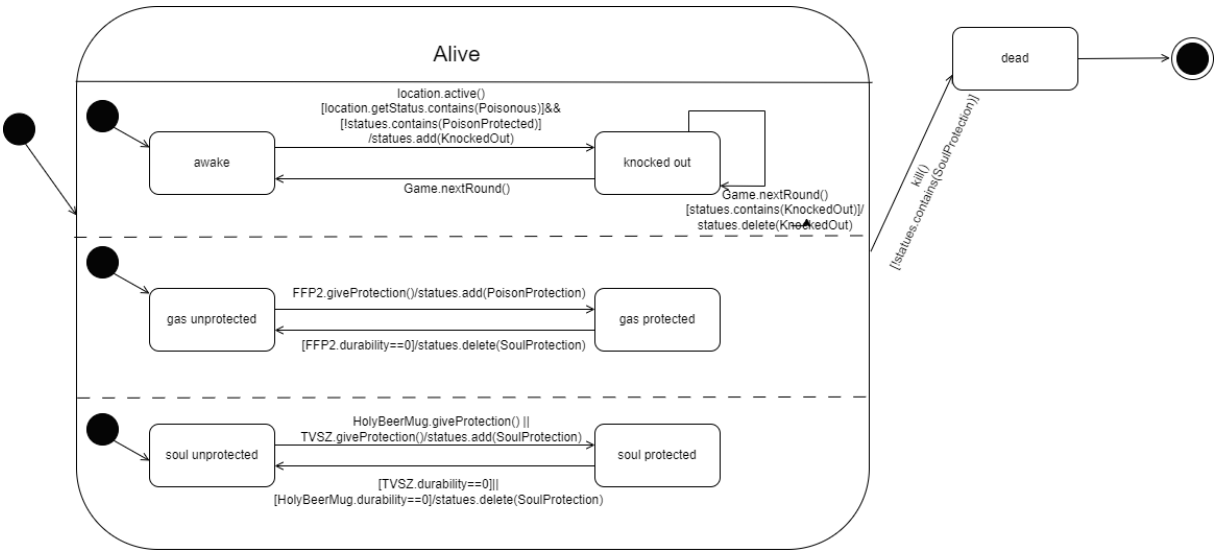


3.4.18 Delete Door

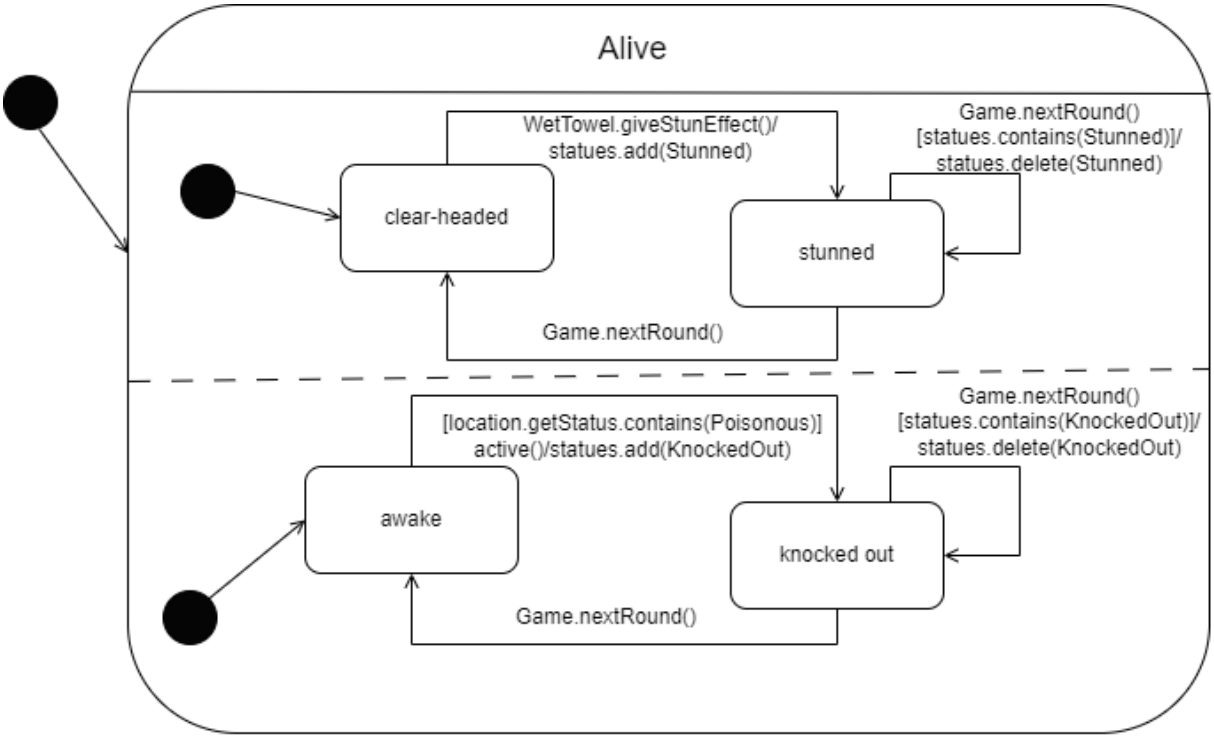

Delete Door
:Labyrinth
r1:Room
r2:Room
deleteDoor(r1, r2)
setNeighbours()
setNeighbours()

3.5 State-chartok

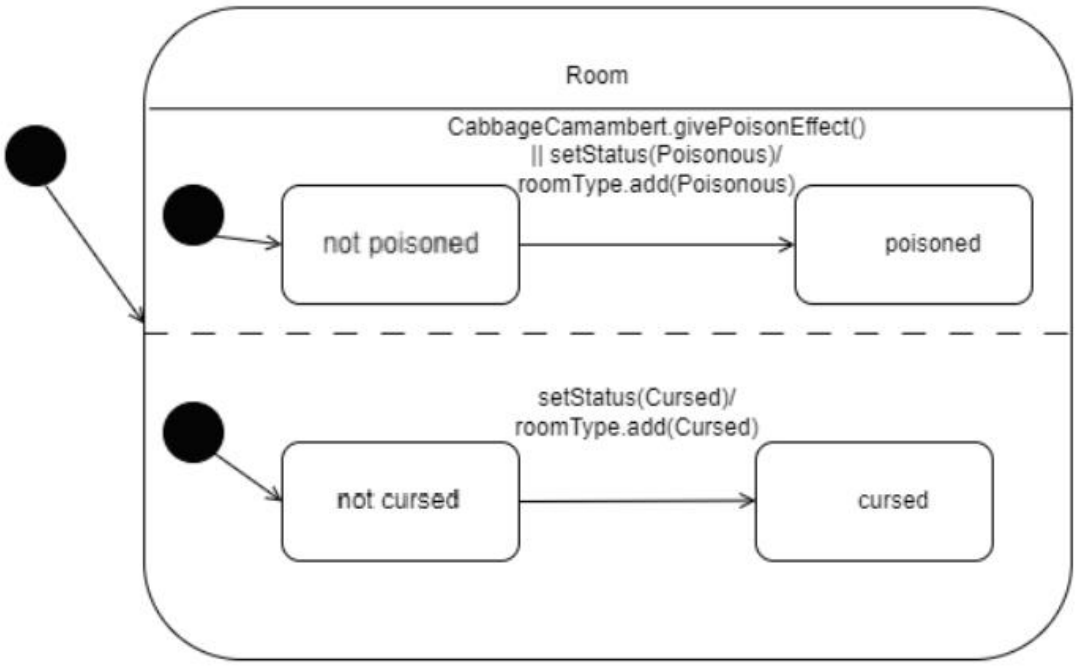
3.5.1 Student States



3.5.2 Teacher States



3.5.3 Room States


Room
CabbageCamambert.givePoisonEffect()
|| setStatus(Poisonous)/
roomType.add(Poisonous)
not poisoned
poisoned
setStatus(Cursed)/
roomType.add(Cursed)
not cursed
cursed

3.6 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2024.03.06. 19:00	3 óra	Fazekas Molnár Simon Juhász Rimmel	Értekezlet: Osztálydiagram átgondolása, szerkesztése
2024.02.28. 8:00	3 óra	Molnár	Osztálydiagram vázának elkészítése
2024.02.28. 9:30	2 óra	Fazekas	Az osztálydiagram ellenőrzése, kezdeti hibák azonosítása, azok javítása
2024.02.28. 19:00	3 óra	Fazekas	Az osztálydiagram továbbfejlesztése, modellezés folytatása
2024.02.29. 17:00	2,5 óra	Molnár	Osztálydiagram feladat leírásnak való ellenőrzése, hibák kijavítása
2024.02.29. 20:00	3 óra	Fazekas Molnár Simon Rimmel	Értekezlet: Osztálydiagramban meghozott döntések átbeszélése, további feladatok kiosztása
2024.02.29. 23:30	2 óra	Fazekas	Értekezleten megbeszélt változtatások implementálása az osztálydiagrammon
2024.03.01 17:00	2,5 óra	Molnár	Osztályleírások készítése
2024.03.01 18:30	1 óra	Fazekas	Osztályleírások befejezése
2024.03.01. 20:00	3 óra	Juhász	Osztálydiagram és osztályleírások áttekintése, hibák és hiányosságok javítása
2024.03.01. 21:00	1,5 óra	Rimmel	Szekvencia diagram, state-chart
2024.03.02. 10:00	2 óra	Rimmel	Diagramok készítése
2024.03.02 17:00	3 óra	Juhász	Osztálydiagram és osztályleírások további áttekintése, a megmaradt hibák és hiányosságok javítása

2024.03.03 11:00	5 óra	Juhász	Szekvenciadiagramok létrehozása
2024.03.03 14:00	4 óra	Simon	State-chartok készítése
2024.03.03 22:00	1 óra	Rimmel	Napló rendezése, dokumentum ellenőrzése