**4. Analízis modell kidolgozása II.**

10 – Extra

Konzulens:

Dobos-Kovács Mihály

Csapattagok

|  |  |  |
| --- | --- | --- |
| **Draskóczi Dóra Boglárka** | **CTWF8V** | **ddb.dora@gmail.com** |
| Halász Máté | ZINVFE | matejjhalasz@gmail.com |
| Molnár Martina | BFCYOE | 1011molnarmartina@gmail.com |
| Pigler András | XERZQ0 | [andras.pigler@gmail.com](mailto:andras.pigler@gmail.com) |
| Serfőző Dávid | F05MHO | [serdavid3010@gmail.com](mailto:serdavid3010@gmail.com) |

2024. 03. 11.

# Analízis modell kidolgozása

## Objektum katalógus

### Labirintus

Ez lesz a játék színtere. Szobákból épül fel, melyeket ajtók kapcsolnak össze egymással, felépítése időnként változhat. A játékosok itt játszák le a teljes játékot.

### Karakter

Eltérő tulajdonságokkal és célokkal rendelkező játékos, aminek döntéseit vagy a felhasználók, vagy a gép határozza meg. Képes tárgyakat felvenni és letenni.

### Oktató

Célja, hogy akadályozza a hallgatókat a Logarléc megszerzésében. Véletlenszerűen mozog a szobák között és véletlenszerűen tudja a szobákban levő tárgyakat felvenni/letenni. Ha egy vagy több hallgatóval kerül egy szobába és azok nem tudnak védekezni ellene valamilyen tárggyal, akkor az ott lévő hallgatók elveszítik a játékot.

### Hallgató

A felhasználó irányítja, célja a Logarléc megtalálása és felvétele adott időn belül. Tud szobák között mozogni, képes a bennük lévő tárgyakat felvenni, azokat – a megengedett mennyiségig – tárolni, illetve használni/aktiválni. Ha oktatóval találkozik és nincsen nála használható tárgy, elveszti a játékot.

### Szoba

A játék szobákból épül fel, ezek közt mozoghatnak a hallgatók és oktatók. Egy-egy szobából legalább egy, de esetenként sok másik szobába is nyílhat ajtó. A szobákban különféle tárgyak lehetnek. Vannak tulajdonságaik, amelyek közt van olyan ami mindegyik szobára érvényes - ilyen például a befogadóképességük -, és vannak olyanok melyek csak egyesekre igazak: mérges gázos vagy elátkozott szoba.

Mérges gázos szobába belépő hallgatók és oktatók egy rövid időre eszméletüket vesztik és a náluk lévő tárgyakat elejtik. Ha van náluk *FFP2-es maszk* nevezetű tárgy (3.1.11), az megvédi őket az ájulástól.

Elátkozott szobák szomszédaihoz vezető ajtók időközönként eltűnnek/megjelennek. A szobák szomszédai ekkor nem változnak meg.

A szobák képesek egyesülni és osztódni, de az előbbi csak már szomszédos szobák közt lehetséges. Szobák egyesülésekor mindkét szoba tulajdonságával és szomszédaival az új szoba is rendelkezni fog, befogadóképessége pedig a nagyobb befogadóképességel rendelkező szobáéval lesz azonos.Szoba osztódásakor, a két szoba szomszédos lesz egymással és mindkét szoba rendelkezni fog az eredeti tulajdonságaival. A szomszédokat és a bennük lévő tárgyakat a szobák véletlenszerűen fogják megkapni.

### Tárgyak

Mind az oktató, mind a hallgató fel tudja venni ezeket a szobákban. Különleges és különféle erőkkel bírnak, melyekkel befolyásolni tudják környezetüket vagy elősegíthetik a hallgatókat és hátráltatják az oktatókat céljaik elérésében. Általában aktiválni lehet őket, hogy fel lehessen képességüket használni. Egyes tárgyaknak ’lejárati idejük’ is van, aminek letelte után elveszítik adottságukat és eltűnnek.

### Logarléc

A játék során a hallgatók célja megtalálni ezt az ereklyét. Amint felvették, megnyerték a játékot.

### Tranzisztorok

Teleportálni tudják aktiválójukat egyik szobából a másikba. A szobák szomszédossága nem releváns a működésükre. Használatuk előtt páronként csatlakoztatni kell őket, összekapcsolás nélkül nincs különleges képességük. Az összekapcsolt pár egy tagját a hallgató leteheti egy szobába, majd haladhat tovább (a tranzisztorok korlátlan ideig használhatók, így bármennyi idő eltelhet a képesség felhasználása nélkül). Ha a hallgató a nála maradó tranzisztort bekapcsolja és leteszi, akkor a másik tranzisztor szobájába kerül, a bekapcsolt tranzisztor pedig kikapcsol. Kikapcsoláskor a két tranzisztor közti kapcsolat megszűnik.

### Szent Söröspohár

A tárgy adott ideig védettséget nyújt a hallgatóknak az oktatók ellen.

### TVSZ denevérbőrre nyomtatott példányai

A tárgy három alkalommal menti meg a hallgató életét az oktatóval való találkozás során.

### Nedves táblatörlő rongy

A tárgy képessége, hogy míg ki nem szárad, teljesen megbénítja a vele egy szobában levő oktatókat, így azok sem lelket elvenni, sem mozogni, sem tárgyakat felvenni vagy eldobni nem képesek.

### Dobozolt káposztás camembert

A tárgy képessége annak környezetét befolyásolja. Kinyitása után a szoba egyik tulajdonsága (ha eddig nem volt gázos) megváltozik és gázos lesz (3.1.1). Használata után (miután ki lett nyitva), a doboz eltűnik.

### FFP2-es maszk

A tárgy képessége védelmet biztosít gázos szobák ellen. Nem kell aktiválni, ha a hallgató belép egy ilyen tulajdonsággal rendelkező szobába, automatikusan felveszi azt. Használat során azonban veszít az erejéből, amint teljesen el lett használva, eltűnik.

## Statikus struktúra diagramok

## Osztályok leírása

### Beer

#### Felelősség

Egy tárgyat valósít meg. Védettséget nyújt a hallgatóknak oktatókkal szemben.

#### Ősosztályok Item → TimeLimitedItem

#### Interfészek

Nem valósít meg interface-t.

#### Asszociációk

#### Attribútumok

* **int time**:   
  Tárolja, hogy mennyi ideig nyújt védettséget oktatókkal szemben

#### Metódusok

* **void ProtectStudent(Student s):**Adott ideig megvédi a hallgatót az oktatóval szemben.
* **void use():**   
  További függvényeit hívja meg, hogy védelmet biztosítson oktatók ellen a hallgatók számára az oktatók ellen.
* **void accept(**ItemVisitor ch):  
  A visitor minta megfelelő függvényét használja.

### Camambert

#### Felelősség

Egy tárgyat valósít meg. Megbénítja a vele egy szobában lévő, védettséggel nem bíró karaktereket.

#### Ősosztályok

Item

#### Interfészek

#### Asszociációk

#### Attribútumok

#### Metódusok

* **void gassing(Room r):**   
  Elgázosítja a szobát, amelyben a hallgató található, aki használja.
* **void use()**:   
  További függvényeket hív meg, hogy elgázosítsa az adott szobát, amiben kinyitották.
* **void accept(ItemVisitor**: ch):  
  A visitor minta megfelelő függvényét használja

### Character

#### Felelősség

Absztrakt ősosztály, a hallgatóknak és az oktatóknak.

#### Ősosztályok

#### Interfészek

* ItemVisitor

#### Asszociációk

* **asszociáció:** Túloldali szereplő: szoba. Több az egyhez kapcsolat.   
  Minden karakter minden pillanatban egy szobában tartózkodik.
* **irányított asszociáció:** Túloldali szereplő: Item. Egy a többhöz kapcsolat.  
  Minden karakter legfeljebb 5 tárggyal rendelkezhet.
* **kompozíció:** Túloldali szereplő: Tranzisztor. Egy a többhöz kapcsolat.  
  Minden karakternél legfeljebb 2 tranzisztor lehet.

#### Attribútumok

* **int itemCapacity:** Tárolja, hogy mennyi item lehet egyszerre a karakternél
* **bool unconscious:** Tárolja, hogy az adott karakter éppen eszméletlen-e vagy sem.

#### Metódusok

* **void pickUp(Item item):**   
  Absztrakt függvény. A karakterek ezzel vehetnek fel egy-egy tárgyat.
* **void putDown(Item item):**Absztrakt függvény. Ezzel tesznek le egy-egy tárgyat a karakterek.
* **void move(Room r):**Absztrakt függvény. Az egyes karakterek szobák között mozognak. A paraméterként kapott szobába kerül át az adott karakter.
* **void faint():**  
  Ha a karakter ájult állapotba kerül, ez a függvény bénítja meg, dobatja el a tárgyait.
* **void Notify():**
* Értesíti a karaktereket, hogy egy Oktató lépett a szobába.
* **void Step(int round):** Lépteti az őt megvalósító Character-eket
* **void GetParalyzed():** Megbénítja az őt megvalósító Character-eket
* **void GetProtectedAgainstTeacher():** A megfelelő tárgyak használatakor ez állítja be a védettséget az őt megvalósító Character-ekre (Student-ekre) a Teacher-ek ellen
* **void GetProtectedAgainstGas():** A megfelelő tárgy (FFP2) használatakor ez állítja be a védettséget az őt megvalósító Character-ekre gázos szobák ellen

### FFP2

#### Felelősség

Egy tárgyat valósít meg. Védelmet nyújt a hallgatóknak adott ideig, néhány alkalommal, ha van a birtokukban ilyen tárgy.

#### Ősosztályok Item

#### Interfészek

Nem valósít meg interface-t.

#### Asszociációk

#### Attribútumok

* **int lives**:   
  Tárolja, hogy hány alkalommal képes megvédeni egy karaktert a játék hátralévő részében.
* **int time**:   
  Tárolja, hogy az adott alkalommal mennyi ideig képes védeni a karaktert.

#### Metódusok

* **void ProtectStudent():**   
  Megvédi a hallgatót adott ideig.
* **use():**További függvényeket hív meg, hogy elgázosítsa az adott szobát, amiben kinyitották.
* **void accept(**ItemVisitor ch):  
  A visitor minta megfelelő függvényét használja

### GameController

#### Felelősség

A játék belső működéséért felelős, figyeli, hogy tart-e még a játék, lépteti az egyes köröket, elindítja és befejezi a játékot.

#### Ősosztályok -

#### Interfészek -

#### Asszociációk

#### kompozíció: Túloldali szereplő: Room, eltárolja a szobákat

#### Attribútumok

#### int currentRound: Tárolja az adott kör sorszámát

#### Metódusok

#### void StartGame(): Elindítja a játékot.

#### void EndGame: Befejezi a játékot, ha a hallgatók megszerezték a logarlécet.

* **void Step()**: Lépteti a köröket a játék során.
* **void SetCurrentRound():** Beállítja az adott kör sorszámát
* **void AnyStudentsAlive():** Ellenőrzi, hogy van-e még életben Student, ha nincs, akkor vége a játéknak
* **void StudentsHaveLogarlec():** Ellenőrzi, hogy valamelyik Student megszerezte-e a Logarlécet, ha igen akkor vége a játéknak

### Item

#### Felelősség

Az egyes tárgyak közös ősosztálya, amelyeket a karakterek felvehetnek a játék során. Céljuk eltérő, a legtöbb védelmet biztosít a hallgatóknak valamilyen formában.

#### Ősosztályok -

#### Interfészek -

#### Asszociációk

* **asszociáció**: a túloldali szereplő: karakter. Több az egyhez kapcsolat.  
  A karakterek tárgyakat vehetnek magukhoz és a játék során használhatják azokat, vagy le is tehetik.
* **asszociáció**: a túloldali szereplő: szoba. Több az egyhez kapcsolat.  
  A szobákban helyezkednek el a tárgyak, ahol a hallgatók és oktatók hozzájuk férnek.

#### Attribútumok

* **-**

#### Metódusok

* **void use():**A tárgyak felhasználhatók a játék során, amikor ez megtörténik, működésbe lépnek.
* **void OnPickedUpBy(Teacher t):**
* Akkor hajtódik végre, ha egy Oktató veszi fel az adott tárgyat
* **void OnPickedUpBy(Student s):**
* Akkor hajtódik végre, ha egy Hallgató veszi fel az adott tárgyat
* **void Accept(ItemVisitor ch):**Absztrakt függvény, azért felel, hogy a visitor mintának megfelelően minden tárgynál annak megfelelő függvényét hívja.

### ItemVisitor

#### Felelősség

Egy interface, amit mindenkinek meg kell valósítania, aki itemeket vehet fel. Ez azért szükséges, mert az Item accept Függvénye azt a típusának megfelelő visit függvényt fogja hívni, amit az ItemVisitor deklarál.

#### Ősosztályok

-

#### Asszociációk

-

#### Attribútumok

-

#### Metódusok

* **void Visit(TVSZ i):**A karakter meghívja az ismeretlen Item accept függvényét és átadja neki saját magát. Ekkor a mögötte lévő valós tárgy (TVSZ) acceptje hívódik meg, ami meghívja a megkapott ItemVisitor visit-jét saját magával. Ez a függvény fogja átadni magát a TVSZ tárggyal.
* **void Visit(Beer i):**A karakter meghívja az ismeretlen Item accept függvényét és átadja neki saját magát. Ekkor a mögötte lévő valós tárgy (Beer) acceptje hívódik meg, ami meghívja a megkapott ItemVisitor visit-jét saját magával. Ez a függvény fogja átadni magát a Beer tárggyal.
* **void Visit(Sponge i):**A karakter meghívja az ismeretlen Item accept függvényét és átadja neki saját magát. Ekkor a mögötte lévő valós tárgy (Sponge) acceptje hívódik meg, ami meghívja a megkapott ItemVisitor visit-jét saját magával. Ez a függvény fogja átadni magát a Sponge tárggyal.
* **void Visit(FFP2 i):**   
  A karakter meghívja az ismeretlen Item accept függvényét és átadja neki saját magát. Ekkor a mögötte lévő valós tárgy (FFP2) acceptje hívódik meg, ami meghívja a megkapott ItemVisitor visit-jét saját magával. Ez a függvény fogja átadni magát a FFP2 tárggyal.
* **void Visit(Camembert i):**A karakter meghívja az ismeretlen Item accept függvényét és átadja neki saját magát. Ekkor a mögötte lévő valós tárgy (Camembert) acceptje hívódik meg, ami meghívja a megkapott ItemVisitor visit-jét saját magával. Ez a függvény fogja átadni magát a Camembert tárggyal.
* **void Visit(Logarlec i):**A karakter meghívja az ismeretlen Item accept függvényét és átadja neki saját magát. Ekkor a mögötte lévő valós tárgy (Logarlec) acceptje hívódik meg, ami meghívja a megkapott ItemVisitor visit-jét saját magával. Ez a függvény fogja átadni magát a Logarlec tárggyal.
* **void Visit(Transistor i):**A karakter meghívja az ismeretlen Item accept függvényét és átadja neki saját magát. Ekkor a mögötte lévő valós tárgy (Transistor) acceptje hívódik meg, ami meghívja a megkapott ItemVisitor visit-jét saját magával. Ez a függvény fogja átadni magát a Transistor tárggyal.

### Logarléc

#### Felelősség

Egy tárgyat valósít meg, amint egy hallgató magához veszi a logarlécet, megnyeri a játékot.

#### Ősosztályok

Item

#### Interfészek

-

#### Asszociációk

-

#### Attribútumok

-

#### Metódusok

* **void accept(ItemVisitor ch):**  
  A visitor minta megfelelő függvényét használja.
* **onPickedUpBy(Teacher t):**  
  Ha egy oktató veszi fel a logarlécet, még nem jelenti automatikusan a játé végét.
* **onPickedUpBy(Student s):**Amikor egy hallgató veszi fel a lécet, és ezt sikeresen véghez tudja vinni, (mert befér még a tárgyai közé), akkor megnyeri a játékot.

### Room

#### Felelősség

A ”labirintus” szobáit megvalósító osztály. Egy-egy szobából egy vagy akár több másikba is nyílhat ajtó, ismeri a szomszédait. Ezen felül tárgyakat tárol, amiket a karakterek felvehetnek és mozgásuk során átvihetnek egy másik szobába. Közöttük közlekednek a karakterek.

#### Ősosztályok

#### Interfészek

#### Asszociációk

* **aggregáció:** Túl oldali szereplő: GameController  
  A GameController tárolja a szobákat.
* **asszociáció:** Egy a többhöz kapcsolat. Túl oldali szereplő: Room, azaz saját maga.  
  A szobák ismerik egymás szomszédjait. Egy szobának lehet több szomszédja is, de akár egyetlen egy sem.
* **asszociáció:** Egy a többhöz kapcsolat. Túl oldali szereplő: Character.  
  A karakterek szobák között mozognak, így valamelyikben állandóan benne vannak. Egy szobában több karakter is lehet.
* **aggregáció:** Több a többhöz kapcsolat. Túl oldali szereplő: Transistor.   
  Egy szobában legfeljebb 1 tranzisztor lehet
* **asszociáció:** Egy a többhöz. Túloldali szereplő: Item.  
  Egy szobában számtalan tárgy lehet, melyeket a karakterek felvehetnek.

#### Attribútumok

* **int maxCapacity:** A szoba befogadóképessége.
* **int capacity:** A szobában jelenleg tartózkodó karakterek létszáma.
* **bool poisonous:** Számon tartja, hogy az adott szoba éppen elgázosított-e vagy sem.
* **bool cursed:** Számon tartja, hogy az szoba elátkozott-e vagy sem, azaz, nyílnak-e belőle ajtók vagy sem.

#### Metódus

* **void Remove(Character: ch)**:   
  Eltávolítja a szobában lévő játékosokat, ha azok át akarnak menni egy másik szobába, vagy teleportálnak egy tranzisztorral, vagy amennyiben kénytelenek átkerülni máshová, mert nem férnek be az adott szobába.
* **void Accept(Character ch):**Beengedi a karaktereket a szobákba.
* **void Step(int round):**   
  A szobát lépteti minden körben
* **void SeparateRoom():**

Szétválasztja azt a szobát, aminek a SeparateRoom függvénye meghívódott.

* **void MergeRooms():**  
  Egyesíti a két szobát.
* **void GetPoisioned():**Megadja, hogy a szoba elgázosított-e.
* **void GetCursed():**Megadja, hogy a szoba átkozott-e.
* **void Notify():**Értesíti a hallgatót, hogy nem került át, másik szobába.

### Sponge

* **Felelősség**

Egy tárgyat valósít meg, megbénítható vele egy oktató adott időre.

#### Ősosztályok

Item

#### Interfészek

-

#### Asszociációk

-

#### Attribútumok

* + - **int time:** Tárolja, hogy mennyi ideig nyújt védettséget oktatókkal szemben.

#### Metódusok

* + - **paralyze(Student s):**Amikor egy hallgató felhasználja, megbénít adott időre egy oktatót.
    - **use():**További függvényeket hív meg, hogy elgázosítsa az adott szobát, amiben kinyitották.
    - **void accept(**ItemVisitor ch):  
      A visitor minta megfelelő függvényét használja

### Student

#### Felelősség

Egy karaktert valósít meg, amit a játékosok tudnak irányítani, akiknek céljuk felvenni a logarlécet és ezáltal megnyerni a játékot. Felvehetnek tárgyakat, használhatják őket, letehetik azokat.

#### Ősosztályok

Character

#### Interfészek

-

#### Asszociációk

* **Függőség:** Túloldali szereplő: Teacher  
  Egy oktató képes elvenni egy hallgató lelkét, így közvetlen hatása lehet rá a játék során.

#### Attribútumok

* **bool protectedAgainstTeacher:**Amennyiben egy hallgató felhasznál egy sörös poharat vagy a TVSZ-t, akkor védetté válik a vele egy szobában lévő oktatóval szemben.
* **bool protectedAgainstGas:**   
  Amennyiben a hallgatónál van FFP2-es maszk, védetté válik egy gázos szobában.
* **bool hasLogarlec:**

Igaz, ha valamelyik hallgatónál van a logarléc.

* **int numberOfStudents:**   
  Az életben lévő Studentek száma

#### Metódusok

* **void PickUp(Item item):**A hallgató a játék során dönthet úgy, hogy felvesz egy tárgyat abban a szobában, amiben éppen tartózkodik
* **void PutDown(Item item):**A felvételhez hasonlóan, le is tehet egy tárgyat egy hallgató abban a szobában, ahol éppen tartózkodik.
* **void UseItem():**A hallgató felhasználhatja a nála lévő tárgyakat.
* **void Die():**Amennyiben a hallgató egy oktatóval kerül egy szobába, és nincs nála a védelmét szolgáló eszköz, az oktató elveszi a hallgató lelkét, ekkor ő kiesik a játékból.
* **void SetLogarlec():**
* **void getProtectedAgainstTeacher():**Védetté teszi az oktatóval szemben a hallgatót.
* **void getProtectedAgainstGas():**Védetté teszi a hallgatót a gázosított szobán belül, ha van nála eszköz hozzá és szabadon mozoghat tovább

### Teacher

#### Felelősség

Egy karaktert valósít meg.

#### Ősosztályok

Character

#### Interfészek

-

#### Asszociációk

**Függőség:** Túloldali szereplő: Student  
Egy oktató képes elvenni egy hallgató lelkét, így közvetlen hatása lehet rá a játék során.

#### Attribútumok

* **bool paralized:** Tárolja, hogy az adott oktató ájult állapotban van-e vagy eszméleténél van.

#### Metódusok

* **void pickUp(Item item):**   
  Az oktatók is képesek tárgyakat felvenni abban a szobában, amiben tartózkodnak, ez náluk véletlenszerűen működik.
* **void putDown(Item item):**Hasonlóan a felvételhez, véletlenszerűen le is tehetnek tárgyakat az oktatók abban a szobában, ahol éppen tartózkodnak.
* **void Step(int round):**   
  Az oktatók is mozognak a szobák között, ám ez véletlenszerűen körönként történik.
* **void getParalized():**  
  Beállítja, hogy az oktató el legyen kábítva.

### Transistor

#### Felelősség

Egy tárgyat valósít meg, amit a karakterek felvehetnek, kettőt összekapcsolhatnak, és teleportálhatnak vele bizonyos szobák között.

#### Ősosztályok

Item

#### Interfészek

-

#### Asszociációk

* **asszociáció:** Túloldali szereplő: egy másik tranzisztor.   
  Egy tranzisztor összekapcsolható egy másikkal.
* **aggregáció:** Nulla vagy kettő az egyhez. Túl oldali szereplő: Character  
  Egy karakternél semennyi vagy legfeljebb kettő tranzisztor lehet egyszerre.
* **aggregáció:** Nulla vagy egy a nulla vagy egyhez. Túl oldali szereplő: Room.  
  Ha letesznek egy olyan tranzisztort valamelyik szobába, amit előtte összekapcsoltak egy másik tranzisztorral, akkor annak ismernie kell a hozzákapcsolta tranzisztort és hogy melyik szobában tették le, mert oda kell majd tudnia teleportálni a hallgatónak, ha leteszi a másik tranzisztort.

#### Attribútumok

* **bool connected:**Igaz, ha össze van kapcsolva egy másik tranzisztorral az adott tranzisztor.
* **bool activated:**  
  Igaz, ha bekapcsolják a tranzisztort, miután össze lett kötve egy másikkal

#### Metódusok

* **void accept(ItemVisitor ch):**A visitor minta megfelelő függvényét használja.
* **void connectTo(Transistor t):**Egy tranzisztor összekapcsolható egy másik tranzisztorral.
* **void removeConnection():**   
  Megszünteti a kapcsolatot két tranzisztor között.

### TVSZ

#### Felelősség

Egy tárgyat valósít meg. Védelmet biztosít a hallgatóknak.

#### Ősosztályok

Item.

#### Interfészek

-

#### Asszociációk

-

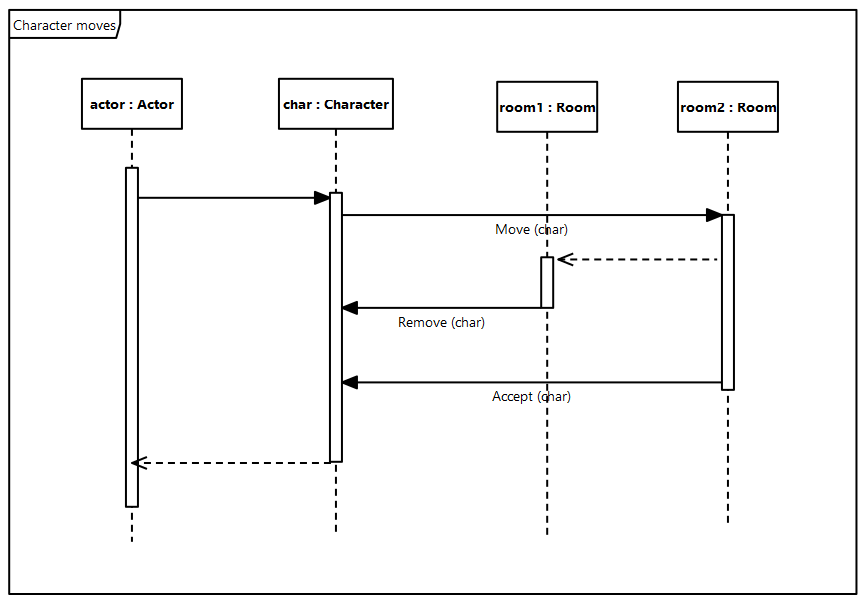
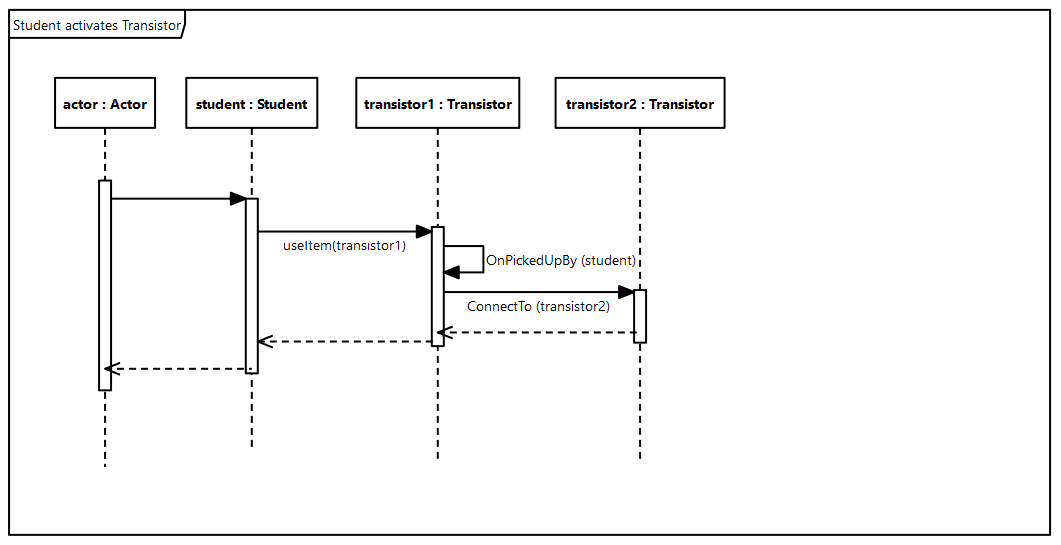
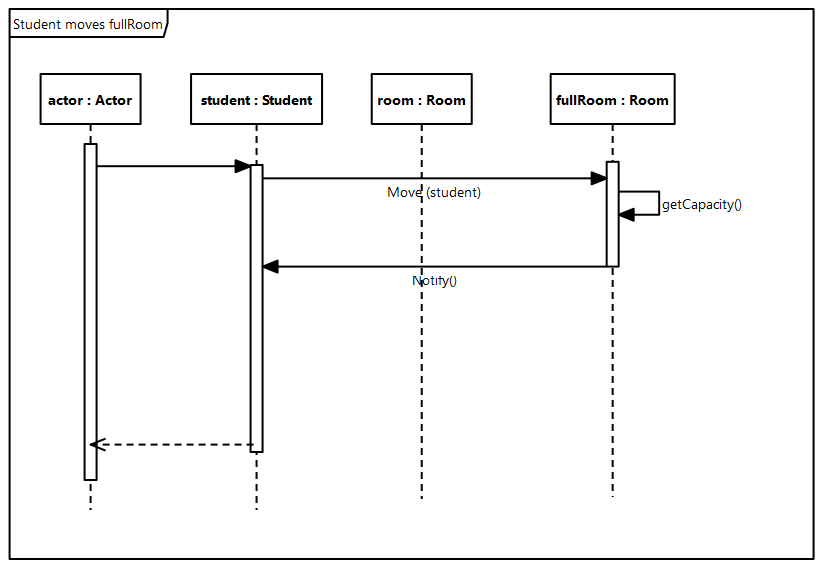
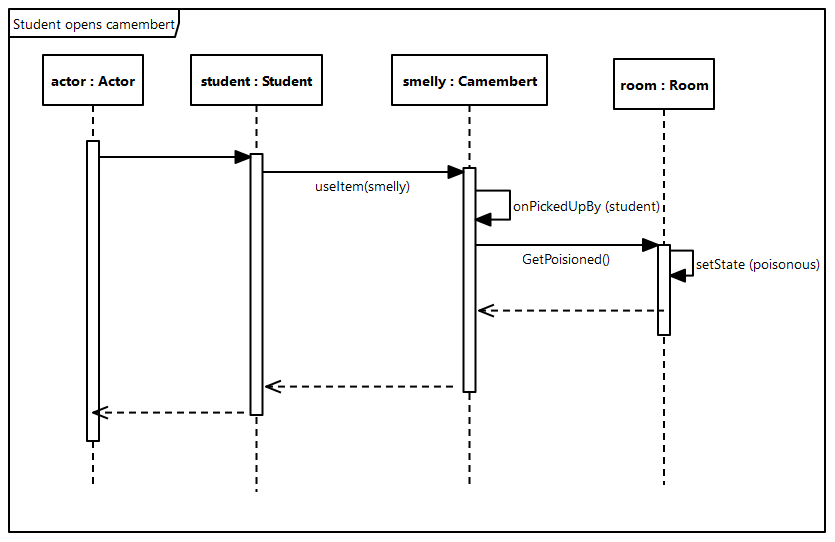
#### Attribútumok

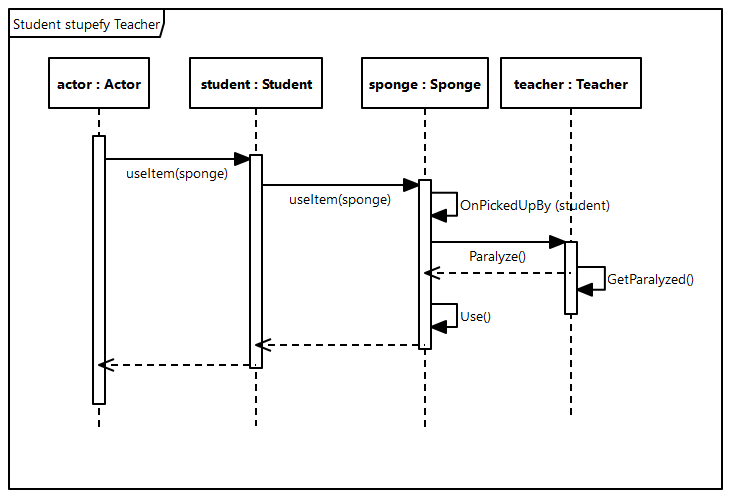
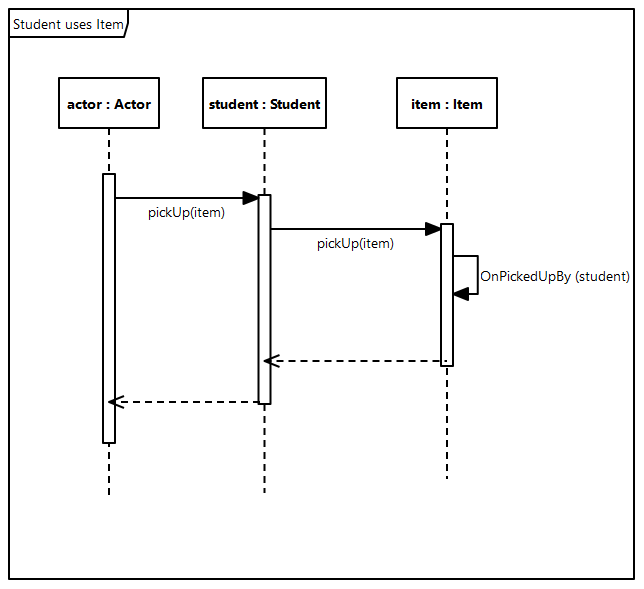
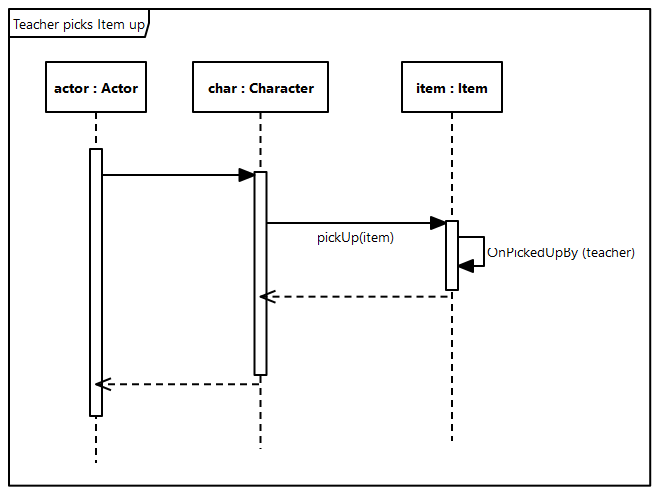
* **int lives:** Tárolja, hogy az adott tárgy hányszor nem lett még felhasználva, mivel csupán adott számú alkalommal védhet meg egy hallgatót.

#### Metódusok

* **void accept(ItemVisitor):**A visitor minta megfelelő függvényét használja.
* **void use():**További függvényeket hív meg, hogy elgázosítsa az adott szobát, amiben kinyitották.
* **void ProtectStudent(Student s):**Ha egy hallgató egy szobába kerül egy oktatóval, akkor megvédi a hallgatót attól, hogy az oktató el tudja venni a lelkét.

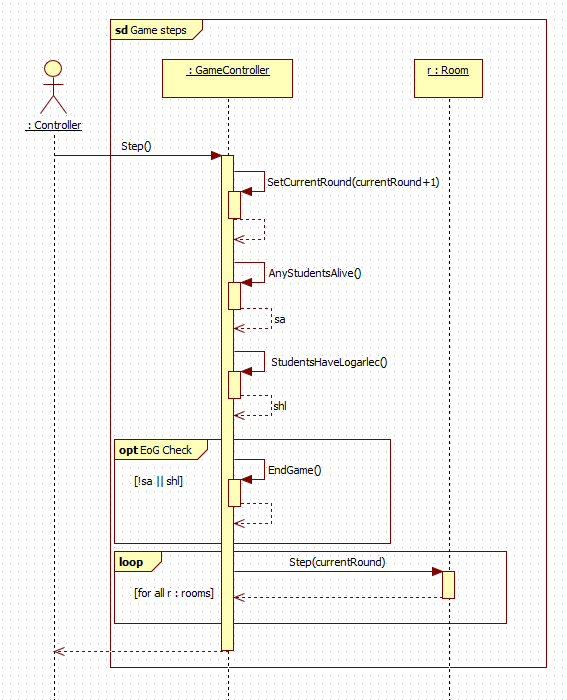
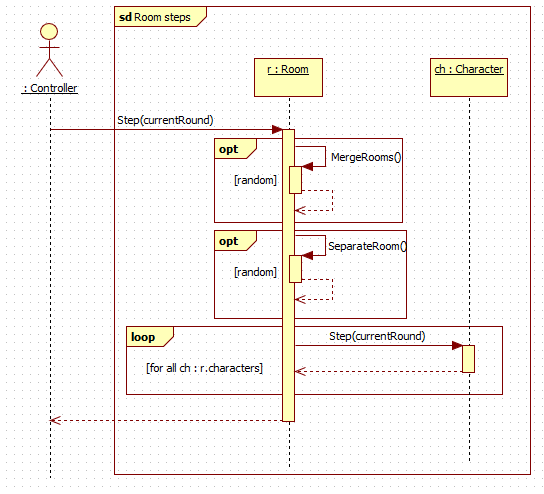
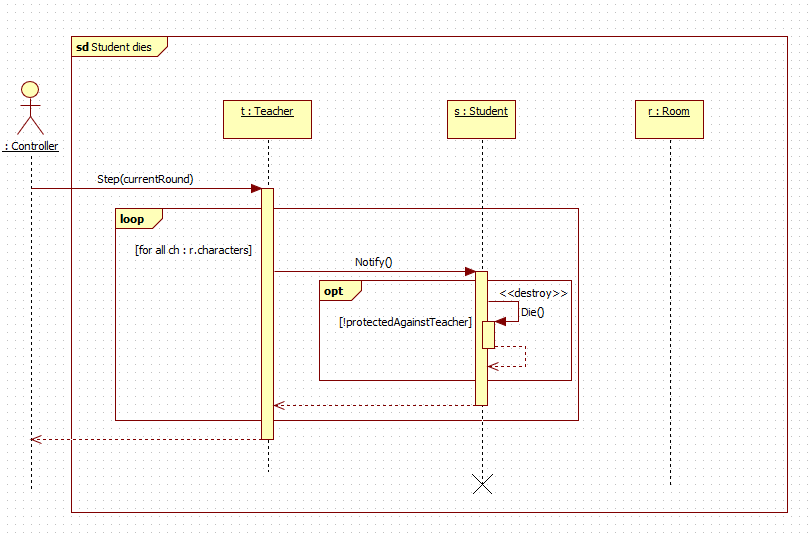
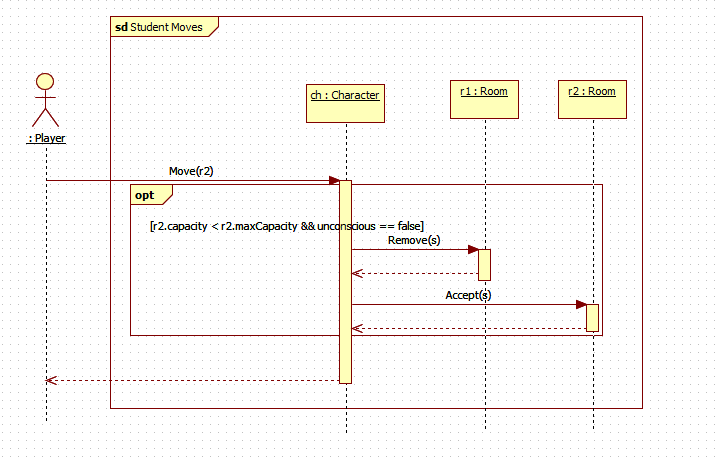
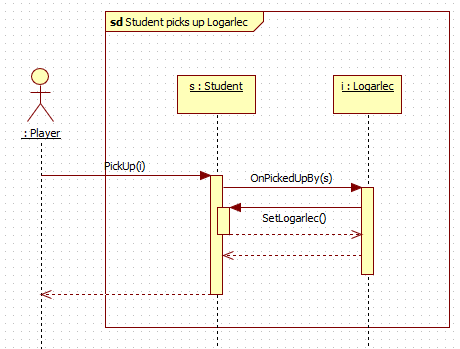
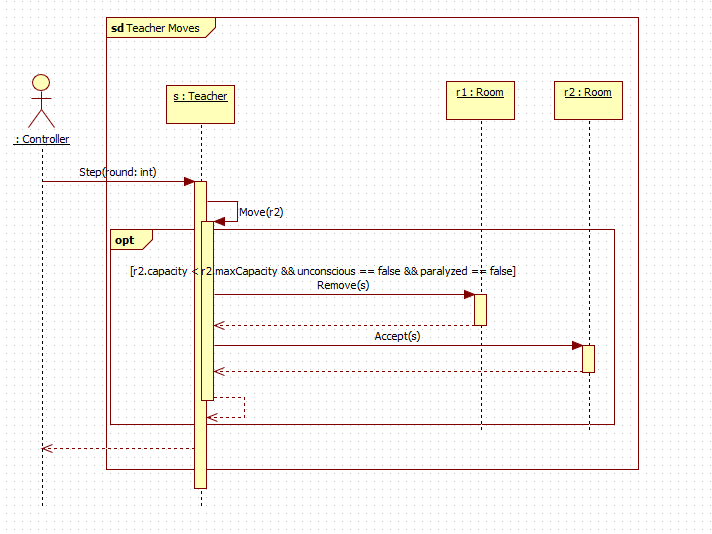
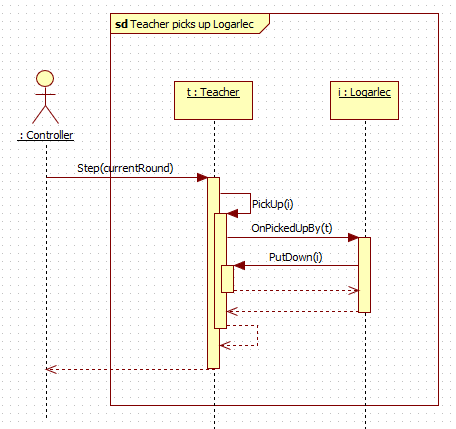
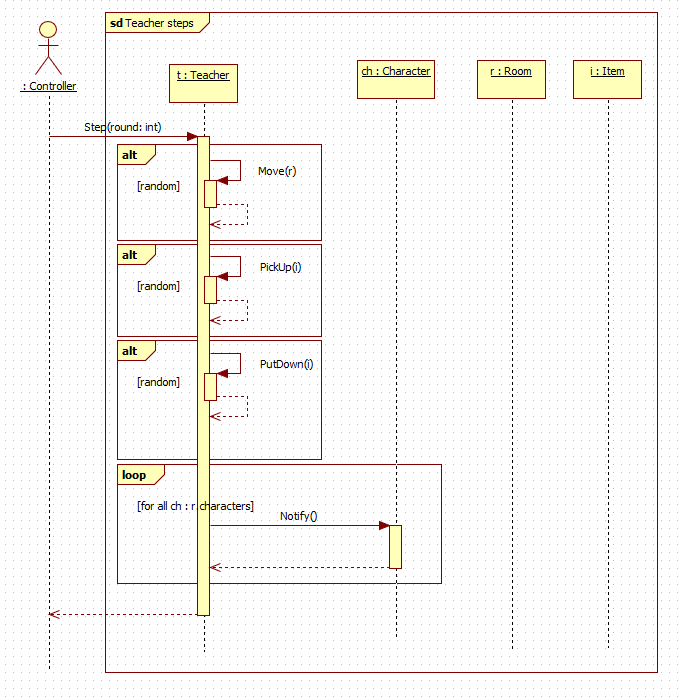
## Szekvencia diagramok

* Character moves
* Student activates transistor
* Student már tele lévő szobába próbál belépni
* Student opens camambert
* A képen szöveg, diagram, sor, Párhuzamos látható

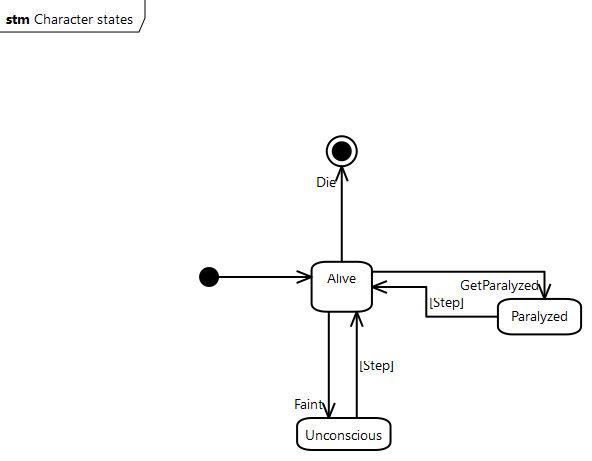
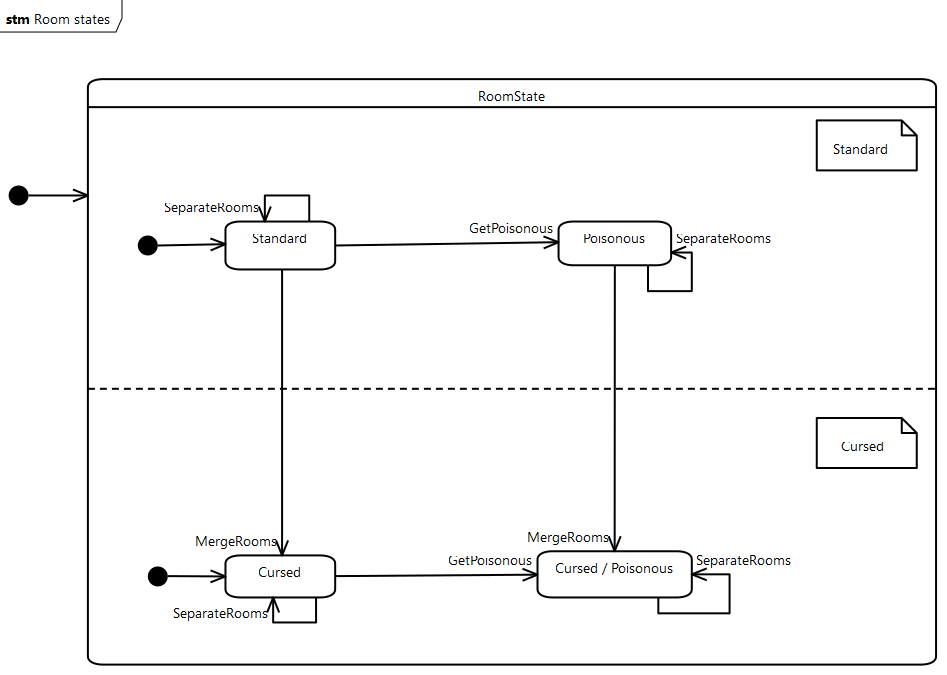
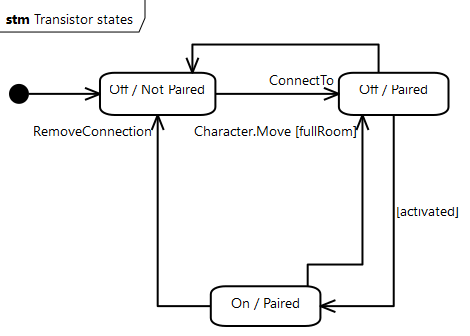
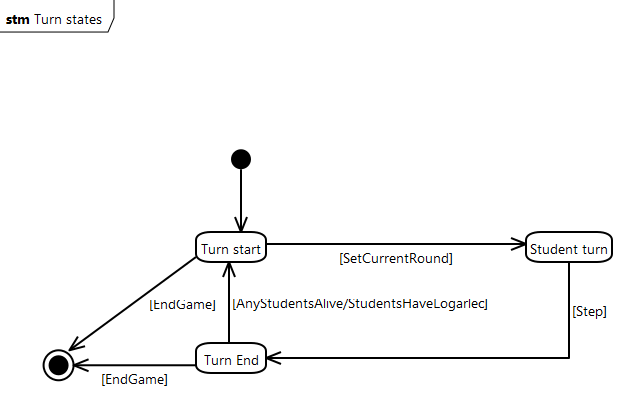
  Automatikusan generált leírásStudent picks item up
* Student stupefy Teacher
* Student uses Item
* Teacher picks item up
* Character in deadly room

A képen szöveg, Párhuzamos, diagram, sor látható

Automatikusan generált leírás

* Game steps
* Room steps
* Student dies
* Student moves
* Student picks up logarléc
* Teacher moves
* Teacher picks up logarléc
* Teacher steps

## State-chartok

* Character
* Room
* Transistor
*   
  GameController

## Napló

|  |  |  |  |
| --- | --- | --- | --- |
| **Kezdet** | **Időtartam** | **Résztvevők** | **Leírás** |
| 2024. 03. 06. | 1,5 óra | Draskóczi,  Molnár,  Serfőző | Konzi |
| 2024. 03. 08. 19:00 | 2 óra | Draskóczi,  Molnár,  Pigler,  Serfőző | Értekezlet.  Döntés:  Molnár és Draskóczi megcsinálják a szekvenciadiagramokat 03. 09. délutánra.  Serfőző módosítja az osztálydiagramot, és a módosítások kapcsán felmerülő szekvenciadiagramokat elkészíti 03. 10-re.  Pigler segíti a munkát menet közben, ha kérdés merül fel, illetve megcsinálja az osztályok leírásának módosítását, miután az osztálydiagram elkészül 03. 10-e estére, valamint az állapotdiagramokat. |
| 2024. 03. 08. 21:00 | 4,5 óra | Molnár | Molnár elkészíti a szekvencia diagramok egyik felét. |
| 2024. 03. 08. 21:00 | 3 óra | Draskóczi | Draskóczi elkészíti a szekvencia diagramok másik felét. |
| 2024. 03. 09. 14:00 | 2 óra | Serfőző | Serfőző elkészíti az osztálydiagramot és az esetlegesen kimaradt szekvencia diagramokat. |
| 2024. 03. 09. 15:00 | 1,5 óra | Pigler | Pigler ellenőrzi az osztálydiagramot és annak leírásait. |
| 2024. 03. 10. 10:00 | 4 óra | Serfőző | Serfőző elvégzi a módosításokat az osztálydiagramon és az esetlegesen kimaradt szekvencia diagramokat elkészíti. |
| 2024. 03. 10. 16:00 | 2 óra | Pigler | Pigler elkészíti az állapot diagramokat. |
| 2024. 03. 10. 19:00 | 2 óra | Molnár | Molnár ellenőrzi és javítja az állapotdiagramokat. Ellenőrzi, hogy konzisztensek-e a szekvenciadiagramokkal. |
| 2024. 03. 10.  19:00 | 3 óra | Draskóczi | Draskóczi összesíti a dokumentumot és elkészíti a naplót, dokumentál, módosításokat hajt végre, hogy a konzisztencia megmaradjon az egyes diagramok között. |