3. Analízis modell kidolgozása 1

10 – itee_team

Konzulens:
Budai Péter

Csapattagok:

Elekes Tamás Csaba	E30C8Z	elekestamas22@gmail.com
Seres Márk Dániel	EUQ8V5	seres.dani@gmail.com
Rédey Bálint Attila	DAVRIZ	botvinnik09@gmail.com
Nagy András	VWBG06	nagyandrasgall@gmail.com
Fuksz Domonkos	GIT0NQ	fukszdomonkos@gmail.com

2014. március 3.

3. Analízis modell kidolgozása 1

3.1. Objektum katalógus

3.1.1. Akadály (Obstacle)

Az akadály (Obstacle) objektum felelőssége egyrészt az, hogy amikor áthalad rajta egy ellenség (Enemy) lelassítsa. Másfelől felelőssége az is, hogy egy-egy ellenség áthaladtával amortizálódjon, valamint ha már teljesen elhasználódott, értesítse azt az út elemet (Path), amelyiken áll.

3.1.2. Ellenség (Enemy)

Egy ellenséget (tünde, hobbit, törp vagy ember) megvalósító objektum. Nyilvántartja a saját életét, sebességét, és azt is, hogy pillanatnyilag melyik mezőn (Field) tartózkodik. Az ő felelőssége még, hogy egy adott lövedék (Bullet) hatására, mennyire sebződjön, vagy ha már sokat sebződött, akkor haljon meg.

3.1.3. GemStat

Ez az osztály azért felel, hogy egy toronyról (Tower) egyszerűen le tudjuk kérdezni a tulajdonságait. Milyen kristályokat vett már rá a játékos, az egész torony értéke mennyi.

3.1.4. Játék (Game)

A játék (Game) objektum felelőssége többek közt a játék ütemezése, az idő múlásának kontrollálása. Ezenkívül az inicializálás, vagyis a játék kezdeti állapotának felvétele, továbbá a modell állapotának folyamatos változása miatti frissítés, valamint ennek a grafikus felületen való megjelenítése.

3.1.5. Kristály (Gem)

Ha a játékos vesz a toronyra/akadályra valamilyen kristályt, akkor jön létre, megkapja a torony, és beépíti magába. Felelőssége, hogy általa érvényre jussanak a fejlesztések.

3.1.6. Lövedék (Bullet)

A tornyok egy lövedéket tárolnak, amit minden lövésnél átadnak a lövő függvénynek. Ennek a lövedéknek a feladata, hogy az ellenségnek megmondja mennyit sebez rajta.

3.1.7. Mező (Field)

A Field osztály a Cell osztály leszármazottja. A nem út típusú cellákat (mező) reprezentálja. Egy mezőre egy torony helyezhető.

3.1.8. Pálya (Map)

A Map osztály a játéktér elemeit, mint cellák tárolja, egy két dimenziós tömbben. Megadja minden egyes cellához, a szomszédjai referenciáját. A pályák egy külső XML fájlban kerülnek tárolásra. A pálya ebből a fájlból töltődik be. Az XML fájlban tárolódik a pálya neve, nehézségi szintje, és a térkép struktúrája. Megadja, hogy a tartalmazott cella út vagy mező típusú-e. Minden út cella tartalmaz egy tulajdonságot, ami következő cella irányát adja meg.

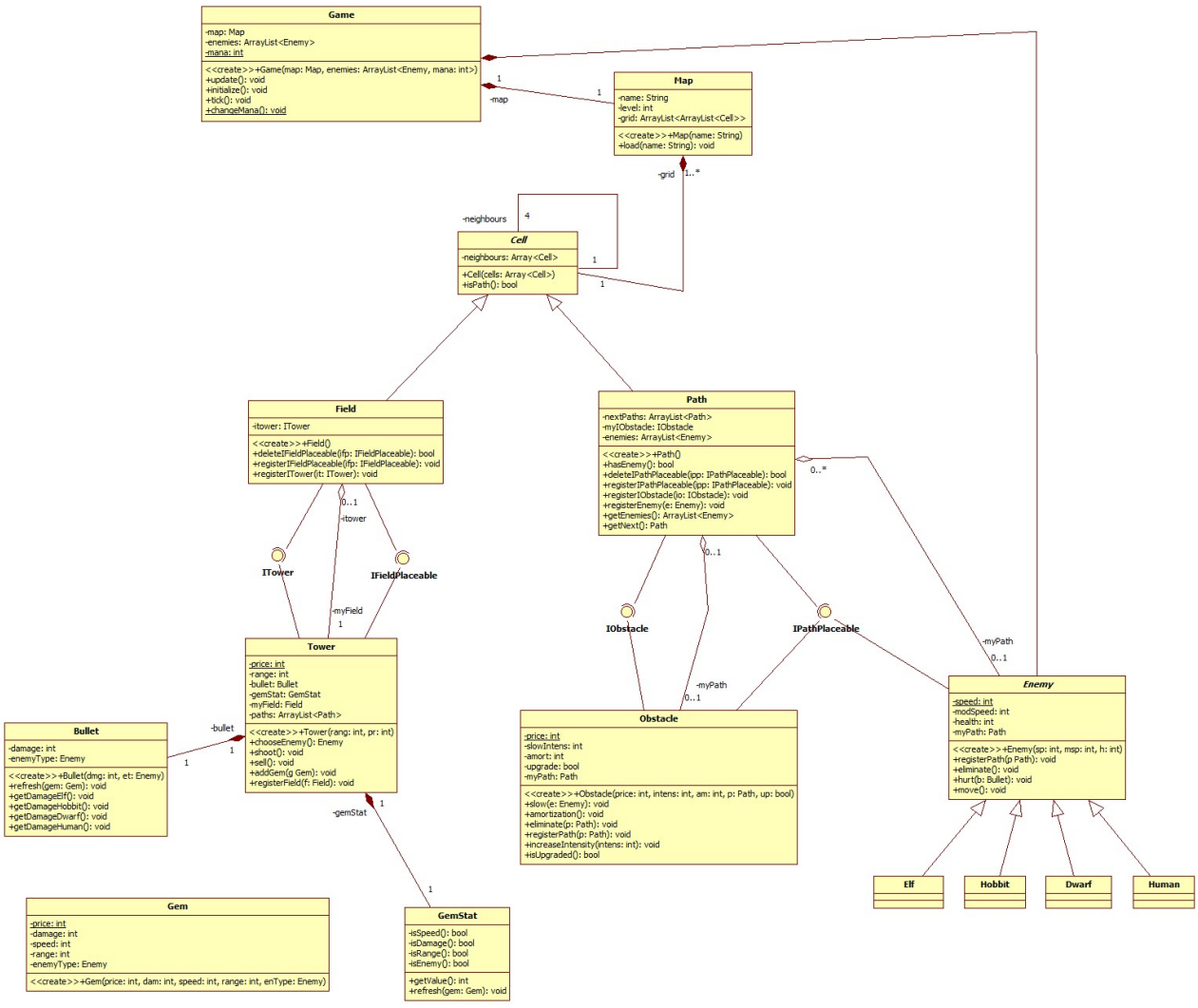
3.1.9. Torony (Tower)

Az egyetlen tervezett toronytípus, le lehet rakni a pályán az úton kívül bárhova. A hatósugarába belépett ellenségekre lőnie kell, lehet fejleszteni lövési sebességét, erejét, újratöltési idejét és egy ellenségtípusra még hatásosabbá tenni a lövedékeit. A játékos varázserőért tud lerakni, illetve eladni tornyokat. Ez a legfontosabb eszköz amivel a játékos meg tudja akadályozni az ellenségek célbajutását.

3.1.10. Út (Path)

A Path a Cell osztály leszármazottja. Az út típusú cellákat reprezentálja. Tartalmazza a rajta lévő ellenségeket és esetleg akadályt.

3.2. Statikus struktúra diagramok



3.1. ábra. Osztálydiagram

3.3. Osztályok leírása

3.3.1. Bullet

- Felelősség
  Ellenség kapja meg, és ebből tudja meg mennyire sebződik.
- Ősosztályok
  Object
- Interfészek
  Nincs
- Attribútumok
  - int damage: alapsebzés
  - Enemy enemyType: a torony itt tárolja, hogy melyik ellenség típusra erősebb a sebzése
- Metódusok
  - void refresh(Gem gem): frissíti kristály vásárlás után a sebzési értékeket.
  - int getHobbitDamage(): ha hobbitot sebez, ezzel a függvénnyel kérdezi le a sebzés értékét.
  - int getHumanDamage(): ha embert sebez, ezzel a függvénnyel kérdezi le a sebzés értékét.
  - int getDwarfDamage(): ha törpöt sebez, ezzel a függvénnyel kérdezi le a sebzés értékét.
  - int getElfDamage(): ha tündét sebez, ezzel a függvénnyel kérdezi le a sebzés értékét
  - Bullet(int damage, Enemy enemyType): konstruktor

3.3.2. Enemy

- Felelősség
  Tudja, hogy mennyi élete van még, milyen sebességgel haladt eredetileg, és milyen sebességgel halad most. Ez egy absztrakt ősosztály, ami összefogja a 4 ellenségtípust (Hobbit, Elf, Dwarf, Human).
- Ősosztályok
  Object
- Interfészek
  IPathPlaceable
- Attribútumok
  - int speed: a normál sebessége
  - int modSpeed: a módosított sebessége (ha egy akadályon halad keresztül)
  - health: élete
  - myPath: az a mező, ahol tartózkodik
- Metódusok
  - Metódusok
  - registerPath(Path p): új mezőre léptetik

  - eliminate(): meghal
  - hurt(Bullet): sebződik (abstract method)
  - move(): mozog, a következő path-ra lép, cellát vált
  - Enemy(int sp, int msp, int h): konstruktor

3.3.3. Enemy subclasses: Elf, Hobbit, Dwarf, Human

- Felelősség
  Sebződés: egy Bullet alapján a saját életét csökkenteni, és ha kell, meghalni.Tehát felüldefiniálja az Enemy ősosztály hurt metódusát.
- Ősosztályok
  Object → Enemy
- Interfészek
  IPathPlaceable
- Metódusok
  - hurt(Bullet): sebződik

3.3.4. ITower

- Felelősség
  A torony funkciói vannak benne.
- Metódusok
  - void setPaths(): a saját cellájából kiindulva a hatósugarával lefedett területen felkeresi, és beregisztrálja a paths listába a path cellákat.
  - void shoot(): A torony akkor lő, ha letelt az újratöltési idő, ekkor megnézi, hogy lőtávon belül van-e ellenség, és ha van meghívja a sebzés függvényét, átadva paraméterként a lövedékét.
  - void addGem(Gem gem): paraméterként megkapja a kiválasztott kristályt, a gameStat-ot frissíti, és a bullet-et is.
  - void sell(): A játékos eladhatja a tornyot, amiért a fejlesztések árának felét kapja meg.A torony eltűnik és felszabadul az alatta lévő hely. Az értékét a gemStat változóból nyeri ki.

3.3.5. Game

- Felelősség
  Lásd objektum katalógus.
- Ősosztályok
  Object
- Interfészek
  Nincs
- Attribútumok
  - Map map: játék térképe

  - List<Enemy> enemies: ellenségek listája
  - static int mana: maradék varázserő

- Metódusok

  - void update(): frissíti a modellt, grafikát
  - void initialize(): kezdeti állapotot felveszi
  - void tick(): ütemező függvény
  - static void changeMana(): mana jóváírására, csökkentésére (fejlesztés/eladás bekövetkeztekor)
  - Game(Map map, ArrayList<Enemy> enemies, int mana): konstruktor

3.3.6. GemStat

- Felelősség
  Lásd objektumkatalógus.
- Ősosztályok
  Object
- Interfészek
  Nincs
- Attribútumok

  - boolean isDamage: vásároltak-e már sebzésnövelő kristályt a toronyra
  - boolean isRange: vásároltak-e már hatósugárnövelő kristályt a toronyra
  - boolean isSpeed: vásároltak-e már lövés sebességnövelő kristályt a toronyra
  - boolean isEnemy: vásároltak-e már ellenségre specializáló kristályt a toronyra

- Metódusok

  - void refresh(Gem gem): egy Kristályt kap, amit beépít a toronyba
  - int getValue: kiszámolja a torony értékét

3.3.7. Gem

- Felelősség
  A kristály osztály frissíti a torony GemStat-ját és Bullet-jét.
- Ősosztályok
  Object
- Interfészek
  Nincs
- Attribútumok

  - int damage: sebzés növelés értéke
  - int range: hatósugár növelés értéke

  - int speed: két lövés közt eltelt idő
  - static int price: az ára, amennyi manába kerül
  - Enemy enemyType: ellenség típus amire erősíti a tornyot

- Metódusok

  - Gem(int damage, int range, int speed, int price, Enemy enemyType): konstruktor

3.3.8. IObstacle

- Felelősség
  Olyan metódusok használatát teszi lehetővé, amelyek az Obstacle típusú elemek viselkedését modellezik.
- Ősosztályok
  Nincs
- Metódusok

  - void slow(int intesity, Path p): szól p-nek, hogy lassítsa le az ellenséget intensity-vel
  - void amortization(): amortizál

3.3.9. Obstacle

- Felelősség
  Lásd objektum katalógus.
- Ősosztályok
  Nincs
- Interfészek
  IObsacle, IPathPlaceable
- Attribútumok

  - int slowIntens: lassítás mértéke
  - Path myPath: a mező, amin rajta van
  - int amort: az elhasználódottság mértéke
  - static int price: az ára
  - bool upgrade: volt-e fejlesztve

- Metódusok

  - void slow(int intesity, Path p): lelassítja a p-n lévő ellenséget intesity-vel
  - void amortization(): csökkenti az amort értékét, ha nulla lesz, hívja az eliminate-et
  - void eliminate(Path p): szól a p-nek, hogy távolítsa el az akadályt
  - void registerPath(Path p): myPath-t beállítja p-re
  - void increaseIntensity(int intensity): fejleszztéskor megnöveli a lassytás mértékét
  - bool isUpgraded(): upgrade-l tér vissza
  - Obstacle(int intens, Path p, int amort, int price, bool up): konstruktor

3.3.10. Tower

- Felelősség
  Lásd objektumkatalógus
- Ősosztályok
  Nincs
- Interfészek
  ITower, IFieldPlaceable
- Attribútumok
  - Bullet bullet: A torony tárol egy lövedéket, mindig ezt lövi ki.
  - GemStat gemStat: A megvásárolt kristályokról tárol statisztikát.
  - List<Path> paths: Hatósugárba eső út cellák.
  - Field myField: mező, amin áll
  - static int price: az ára
  - int range: lőtáv, hatókör
- Metódusok
  - Enemy chooseEnemy(): A torony tárolja a környező path cellákat. Minden tick-ben végig megy rajtuk, és kiválaszt egyet, amelyiken van ellenség, és oda fog lőni. Azzal tér vissza, hogy sikerült-e ellenséget találni.
  - void register(Field field): beregisztrálja, hogy melyik mezőn van
  - setPaths(): beállítja a paths-t
  - shoot(): enemy kiválasztása után rátüzel
  - sell(): tornyot eladjuk: töröljük a mezőjéről és jóváírjuk az érte kapott összeget
  - addGem(Gem g): fejlesztéskor a kapott g Gem alapján beállítjuk a range-t és a bullet tagváltozóit
  - Tower(int rang, int pr): konstruktor

3.3.11. Map

- Felelősség
  Ld. objektum katalógus
- Ősosztályok
  Nincs
- Interfészek
  Nincs
- Attribútumok
  - String name: a pálya neve, egyben az azonosítója
  - int level: a pálya szintje
  - Array<Array<Cell» grid: A cellákat tartalmazó 2 dimenziós tömb

- Metódusok
  - Map(string name): az osztály konstruktora, a paraméterként megadott névvel rendelkező fájlból betölti a pálya térképét
  - void load(string name): megnyitja a paraméterként kapott nevű XML fájlt, és abból betölti a pálya celláinak tulajdonságait, felépíti a pályát.

3.3.12. Cell

- Felelősség
  A Cell a pálya egy egységét reprezentáló osztály. Létrehozásakor megkapja a 4 szomszédja referenciáját. Maga a cella nem tudja, hogy hol van a térképen. A cella tárolja a rajta éppen tartózkodó ellenségek referenciáit. A Cell osztály absztrakt.
- Ősosztályok
  Nincs
- Interfészek
  Nincs
- Attribútumok
  - Array<Cell> neighbours: 4 elemű tömb, tárolja 4 irányban a szomszédjai referenciáját.
- Metódusok
  - Cell(Array<Cell>): konstruktor, paraméterként kapja a szomszédos mezők referenciáit.
  - bool isPath(): olyan értékkel tér vissza amilyen típusú a cella

3.3.13. Field

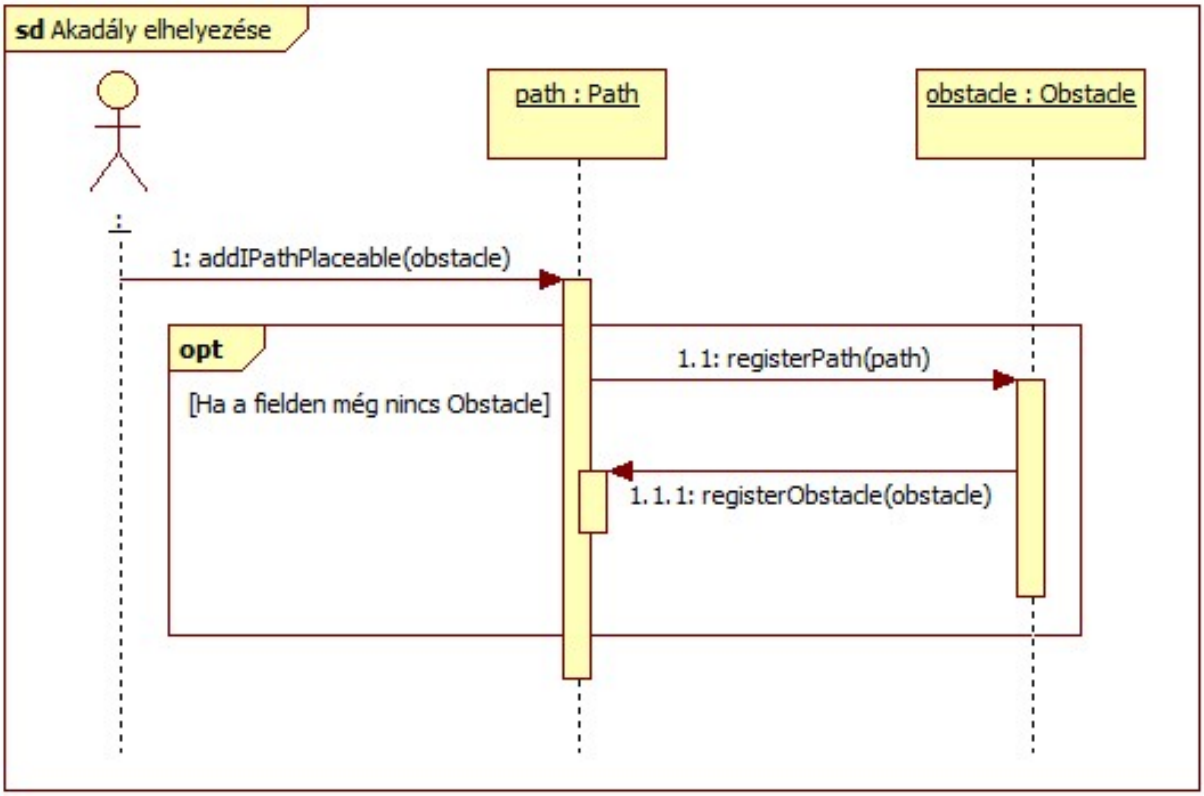
- Felelősség
  Ld. objektum katalógus
- Ősosztályok
  Object $\rightarrow$ Cell
- Interfészek
  Nincs
- Attribútumok
  - ITower itower: a mezőn álló torony interfészű elem tárolása
- Metódusok
  - bool isPath(): hamis értékkel tér vissza
  - void addIFieldPlaceable(): egy új tornyot ad hozzá a mezőhöz
  - void deleteIFieldPlaceable(IFieldPlaceable ifield): eltávolítja a tornyot a mezőről
  - void registerITower(itower ITower): beteszi ifieldbe a kapott tornyot
  - Field(): konstruktor

3.3.14. Path

- Felelősség
  Ld objektum katalógus
- Ősosztályok
  Object → Cell
- Interfészek
  Nincs
- Attribútumok
  - IObstacle iobstacle: az esetleg az úton levő akadályt tárolja
  - ArrayList<Enemy> enemies: az éppen áthaladó ellenségek listája
  - ArrayList<Path> paths: következő path-ok címei
- Metódusok
  - ArrayList<Enemy> hasEnemy(): visszaadja a rajta lévő ellenségek listáját
  - bool isPath(): igaz értékkel tér vissza
  - void deleteIPathPlaceable(IPathPlaceable ipath): kitörli a tárolójából a paraméterként kapott referenciával megegyező tárolt referenciát
  - void registerIPathPlaceable(IPathPlaceable ipath): beregisztrálja a paraméterként kapott objektumot, mint saját magán tartózkodó ellenség
  - bool hasEnemy(): megmutatja, hogy van-e a cellán ellenség
  - void registerEnemy(Enemy e): a kapott ellenséget beteszi az enemies-be
  - void registerObstacel(Obstacle o): a o kapott akadály lesz az obstacle
  - ArrayList<Enemy> getEnemies(): visszatér az enemies-el
  - Path getNext(): paths-ből ad vissza egy elemet

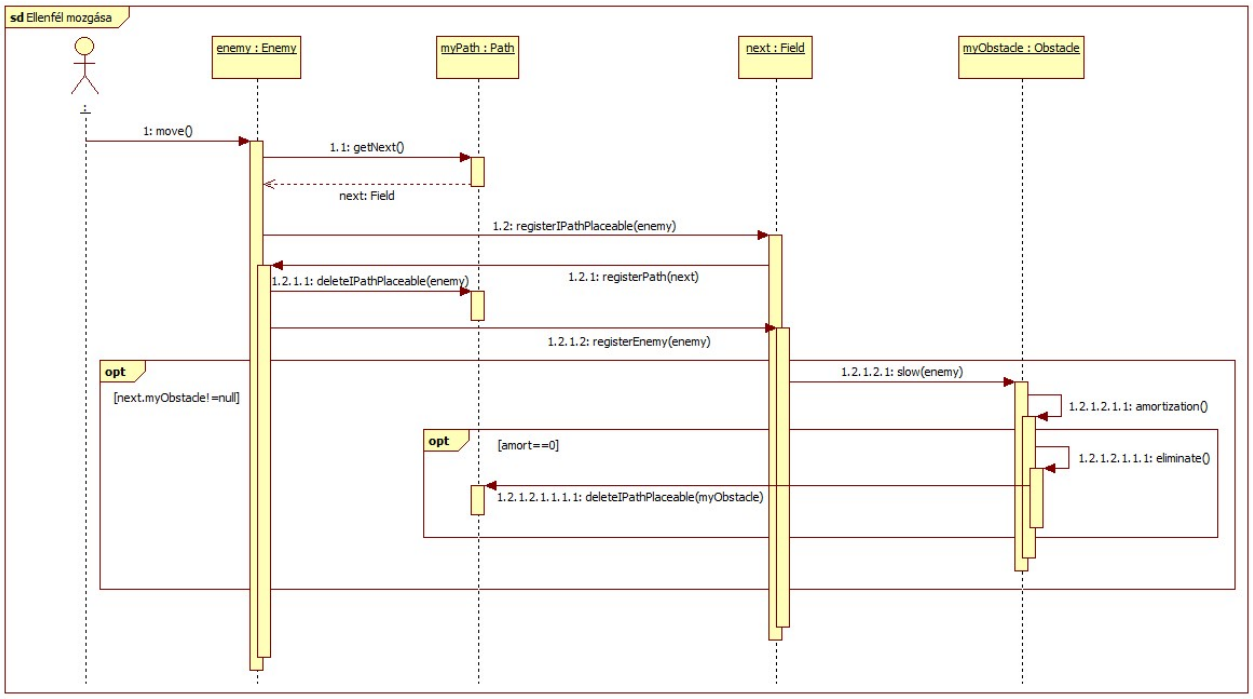
3.4. Szekvencia diagramok

3.4.1. Akadály elhelyezése



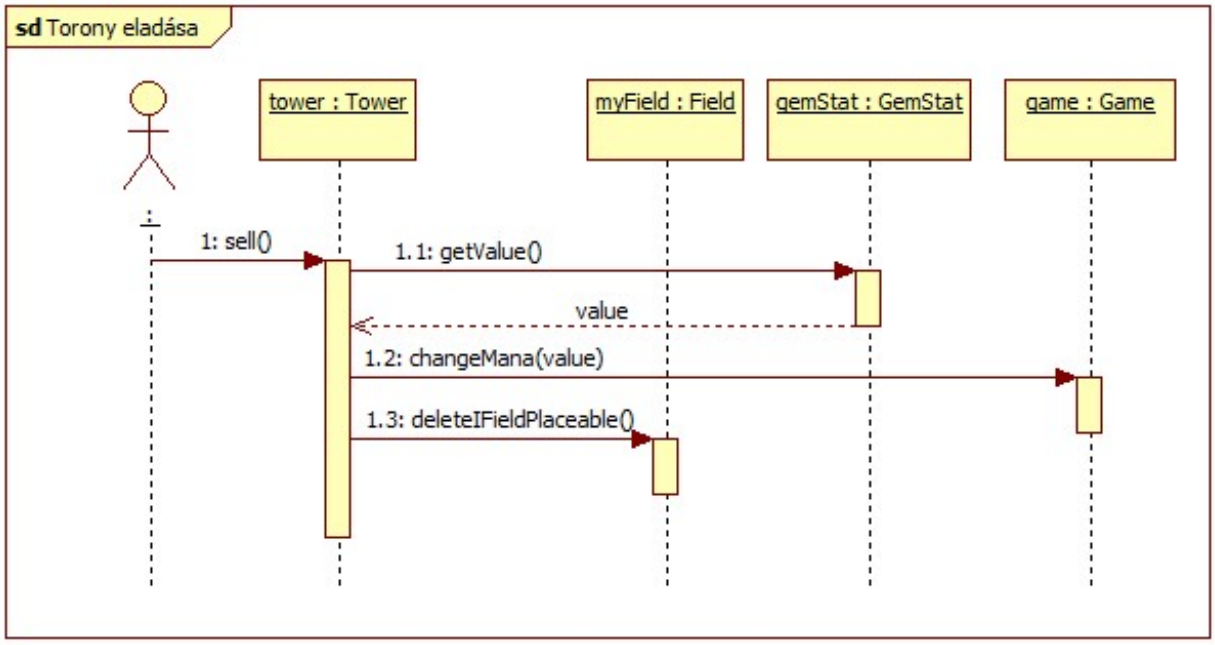
3.2. ábra. Akadály elhelyezése szekvenciadiagram

3.4.2. Ellenfél mozgása



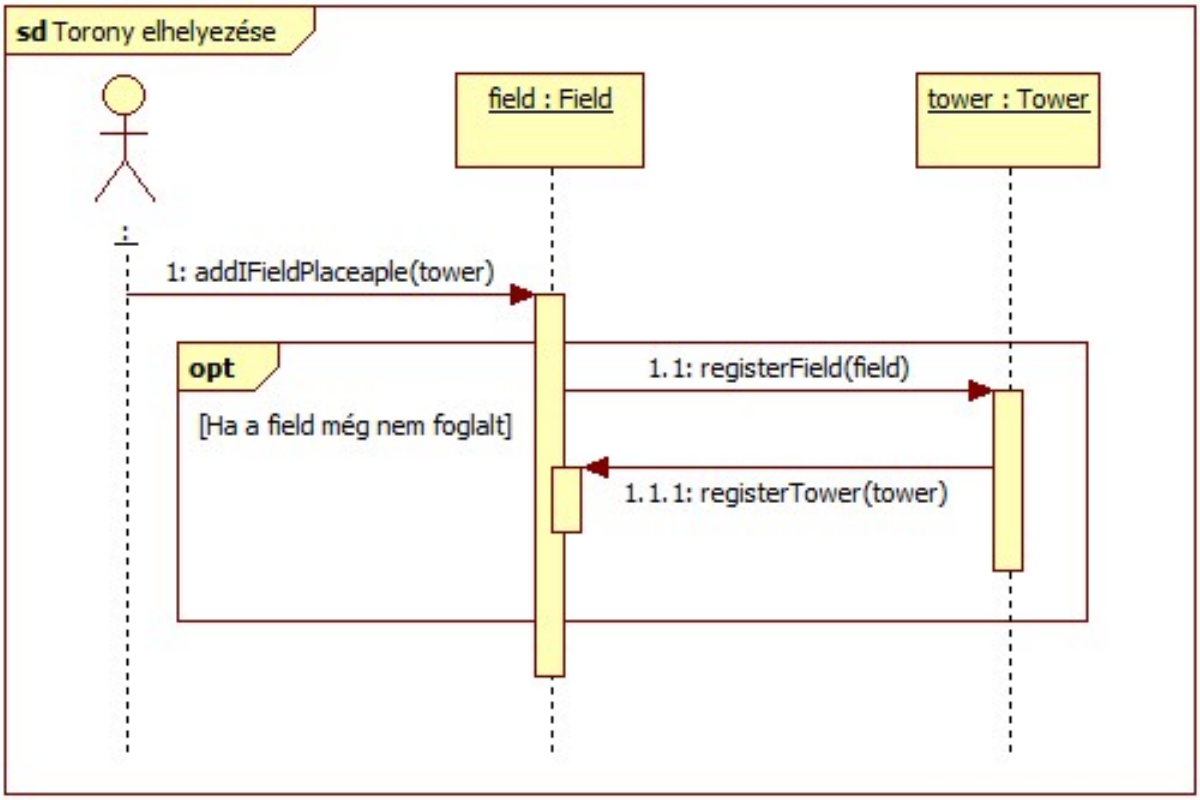
3.3. ábra. Ellenfél mozgása szekvenciadiagram

3.4.3. Torony eladása



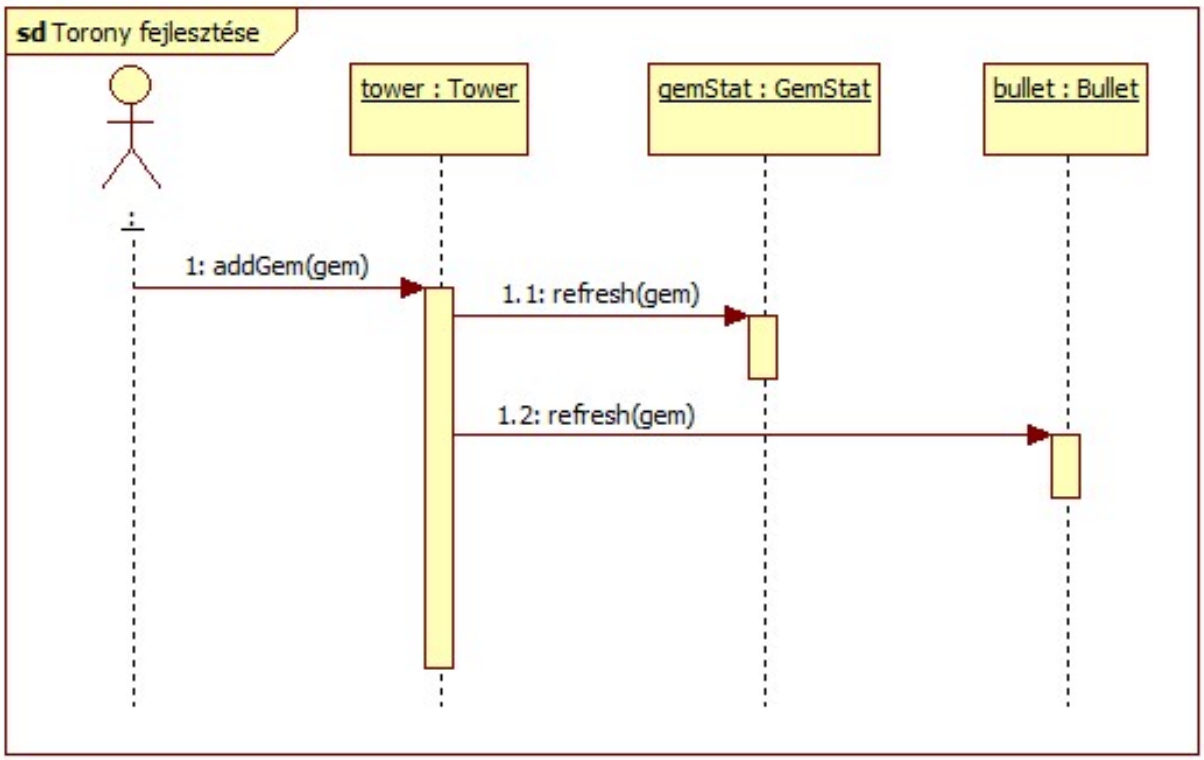
3.4. ábra. Torony eladása szekvenciadiagram

3.4.4. Torony elhelyezése



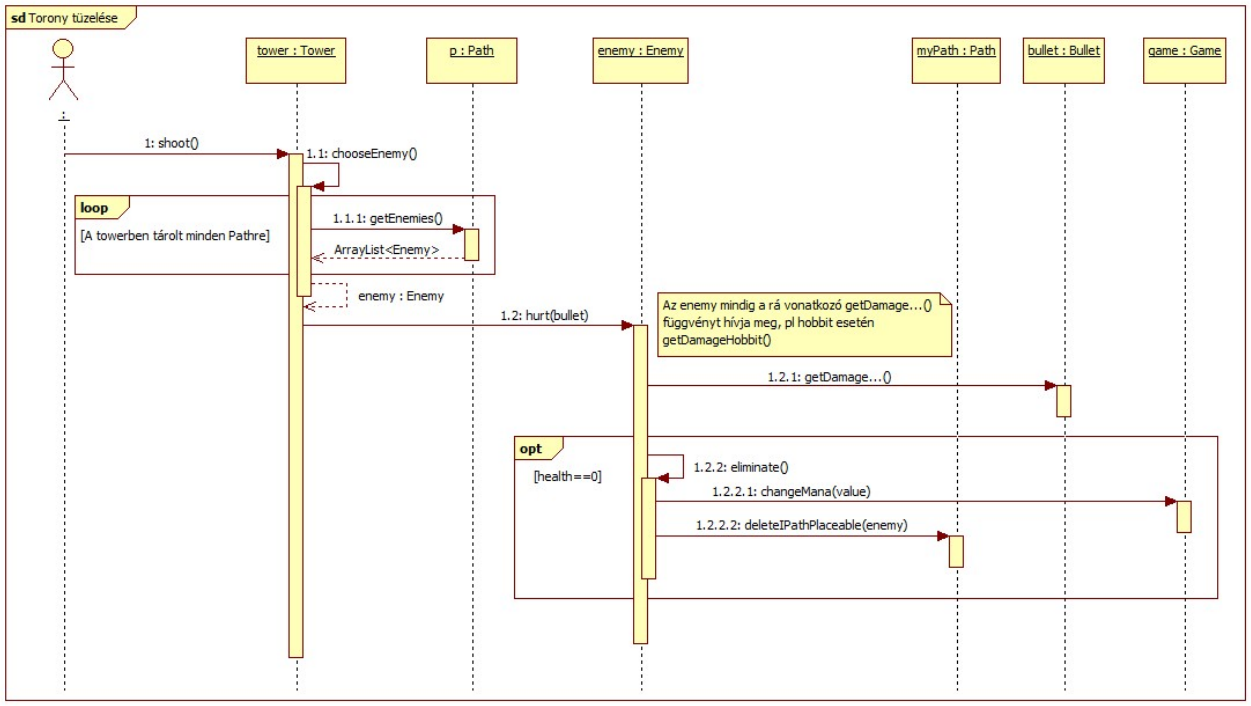
3.5. ábra. Torony elhelyezése szekvenciadiagram

3.4.5. Torony fejlesztése



3.6. ábra. Torony fejlesztése szekvenciadiagram

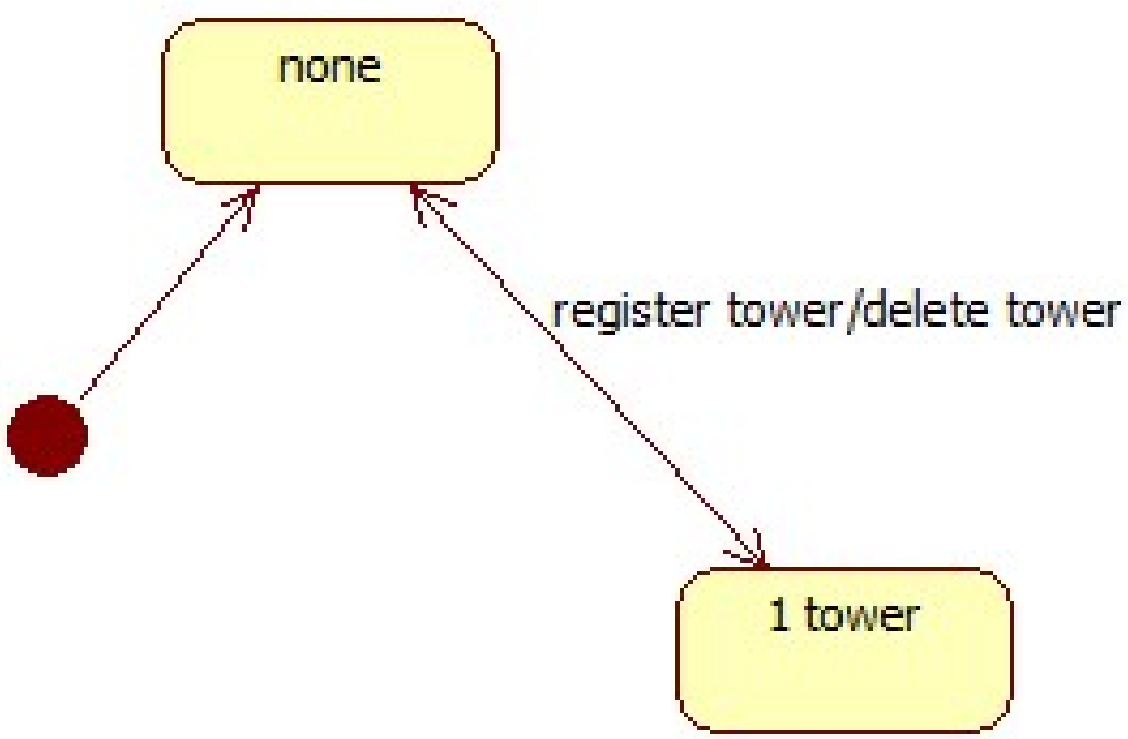
3.4.6. Torony tüzelése



3.7. ábra. Torony tüzelése szekvenciadiagram

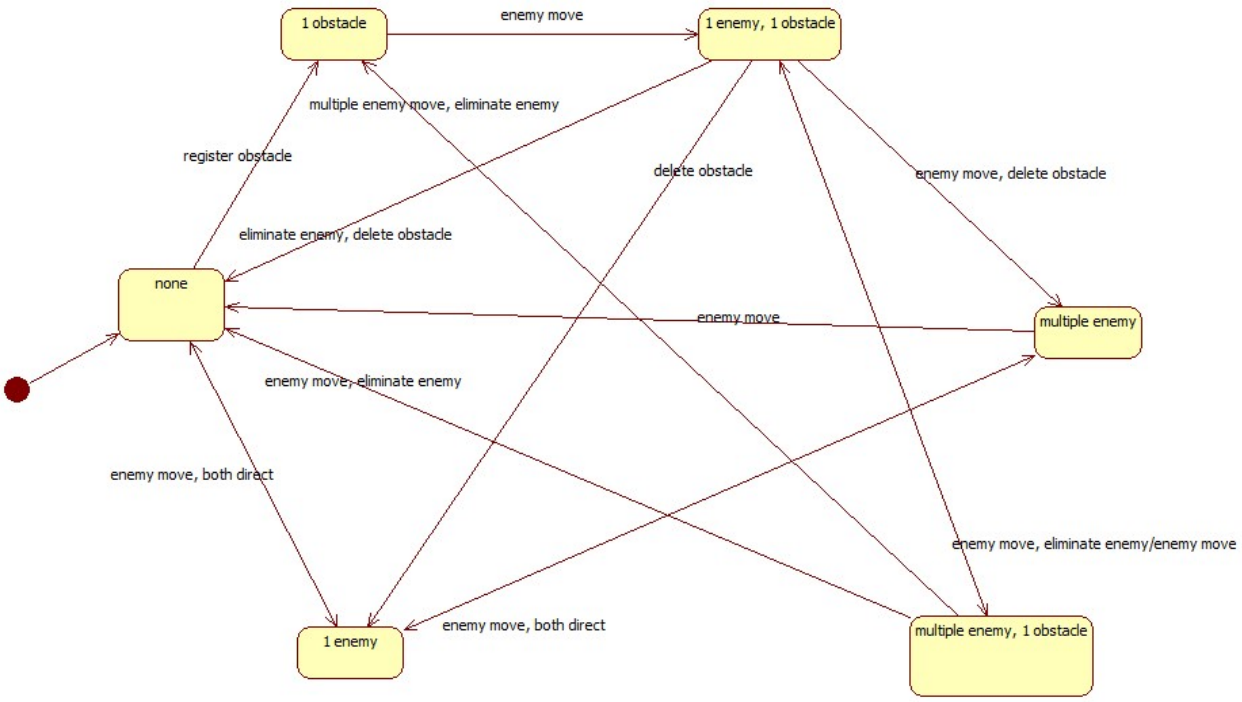
3.5. State-chartok

3.5.1. Field állapota



3.8. ábra. Field állapotdiagram

3.5.2. Path állapota



3.9. ábra. Path állapotdiagram

3.6. Napló

Kezdet	Időtartam	Résztvevők	Leírás
2014.02.26. 08:00	1,5 óra	**Elekes Fuksz Nagy Seres Rédey**	Konzultáció
2014.02.27. 13:00	1,5 óra	**Fuksz Elekes**	Megbeszélés az analízis modellről. Use-casek összeírása.
2014.02.27. 17:00	30 perc	**Fuksz**	2. dokumentáció kisebb részeinek javítása
2014.02.27. 18:00	1 óra	**Elekes**	2. dokumentáció kiegészítés 3.3.3 Tower objektum leírása
2014.02.28. 14:00	5 óra	**Elekes Fuksz Nagy Seres Rédey**	Osztályleírások, hozzájuk tartozó use casek, szekvenciadiagrammok szövegesen: Tower, ITower, Bullet, Gem, GemStat: Elekes Enemy, és leszármazottai: Nagy Map, Cell, Field, IField, Path, IPath: Fuksz Obstacle, IObstacle Game: Rédey use case, szekvenciadiagrammok: Seres, Rédey
2014.03.01. 10:00	3 óra	**Fuksz**	2. dokumentáció LaTeX formára hozása
2014.03.01. 9:20	2,5 óra	**Rédey**	Obstacle, IObstacle, Game osztályok elkészítése

Kezdet	Időtartam	Résztvevők	Leírás
2014.03.01. 12:30	2,5 óra	**Rédey**	Osztálydiagram rajzolása a meglévő osztályokkal
2014.03.01. 13:00	2 óra	**Elekes**	Bullet, ITower, Gem, GemStat leírások, és katalógusok megírása
2014.03.01. 15:00	2,5 óra	**Rédey Fuksz Elekes**	Skype értekezlet. Use case-k kifejtése.
2014.03.01 15:00	1,5 óra	**Rédey**	Osztálydiagram szerkesztése az értekezlet alapján
2014.03.01 17:00	1 óra	**Rédey**	Use casek szerkesztése az osztálydiagram alapján, osztálydiagram és a dokumentum szinkronba hozatala
2014.03.02. 7:00	4 óra	**Seres**	Szekvencia diagramok elkészítése
2014.03.02. 11:00	2 óra	**Rédey, Seres**	Osztálydiagram szerkesztése, use casek megbeszélése
2014.03.02. 13:00	2 óra	**Rédey, Seres, Elekes, Fuksz, Nagy**	Skype konferencia, Enemy és leszármazottai, osztálydiagramm szerkesztése, use case-ek rendszeresítése, szekvenciák rendszeresítése
2014.03.02. 19:30	3,5 óra	**Rédey**	State Chartok elkészítése, dokumentum szinkronizálása az osztálydiagrammal, osztálydiagramm véglegesítése
2014.03.03. 01:00	2,5 óra	**Fuksz**	Dokumentáció véglegesítése