

Szoftver laboratórium 4.

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. május 13.

Tartalomjegyzék

2 Követelmény, projekt, funkcionalitás	10
2.1. Bevezetés	10
2.1.1. Cél	10
2.1.2. Szakterület	10
2.1.3. Definíciók, rövidítések	10
2.1.4. Hivatkozások	10
2.1.5. Összefoglalás	10
2.2. Áttekintés	10
2.2.1. Általános áttekintés	10
2.2.2. Funkciók	11
2.2.3. Felhasználók	12
2.2.4. Korlátozások	12
2.2.5. Feltételezések, kapcsolatok	12
2.3. Követelmények	12
2.3.1. Funkcionális követelmények	12
2.3.2. Erőforrásokkal kapcsolatos követelmények	13
2.3.3. Átadással kapcsolatos követelmények	13
2.3.4. Egyéb nem funkcionális követelmények	13
2.4. Lényeges use-case-ek	14
2.4.1. Use-case leírások	14
2.4.2. Use-case diagram	16
2.5. Szótár	16
2.6. Projekt terv	18
2.6.1. Határidők	18
2.6.2. Átadás	18
2.6.3. Szerepkörök	18
2.6.4. Felhasznált fejlesztőeszközök	19
2.7. Napló	19
3 Analízis modell kidolgozása 1	21
3.1. Objektum katalógus	21
3.1.1. Akadály (Obstacle)	21
3.1.2. Ellenség (Enemy)	21
3.1.3. GemStat	21
3.1.4. Játék (Game)	21
3.1.5. Kristály (Gem)	21
3.1.6. Lövedék (Bullet)	21
3.1.7. Mező (Field)	21
3.1.8. Pálya (Map)	21
3.1.9. Torony (Tower)	22
3.1.10. Út (Path)	22
3.2. Statikus struktúra diagramok	22
3.3. Osztályok leírása	23
3.3.1. Bullet	23
3.3.2. Enemy	23
3.3.3. Enemy subclasses: Elf, Hobbit, Dwarf, Human	24

3.3.4.	ITower	24
3.3.5.	Game	24
3.3.6.	GemStat	25
3.3.7.	Gem	25
3.3.8.	IObstacle	26
3.3.9.	Obstacle	26
3.3.10.	Tower	27
3.3.11.	Map	27
3.3.12.	Cell	28
3.3.13.	Field	28
3.3.14.	Path	29
3.4.	Szekvencia diagramok	30
3.4.1.	Akadály elhelyezése	30
3.4.2.	Ellenfél mozgása	31
3.4.3.	Torony eladása	32
3.4.4.	Torony elhelyezése	33
3.4.5.	Torony fejlesztése	34
3.4.6.	Torony tüzelése	35
3.5.	State-chartok	36
3.5.1.	Field állapota	36
3.5.2.	Path állapota	37
3.6.	Napló	37

4 Analízis modell kidolgozása 2 39

4.1.	Objektum katalógus	39
4.1.1.	Akadály (Obstacle)	39
4.1.2.	Ellenség (Enemy)	39
4.1.3.	Játék (Game)	39
4.1.4.	Kristály (Gem)	39
4.1.5.	Lövedék (Bullet)	39
4.1.6.	Mező (Field)	39
4.1.7.	Pálya (Map)	39
4.1.8.	Torony (Tower)	39
4.1.9.	Út (Path)	40
4.1.10.	Obstaclehez tartozó kristályok: Intensity, Repair	40
4.1.11.	Towerhez tartozó krsítályok: Range, Speed, Damage, EnemyType	40
4.2.	Statikus struktúra diagramok	41
4.3.	Osztályok leírása	41
4.3.1.	Bullet	41
4.3.2.	Enemy	42
4.3.3.	Enemy subclasses: Elf, Hobbit, Dwarf, Human	43
4.3.4.	Game	43
4.3.5.	Controller	44
4.3.6.	IGame	44
4.3.7.	Gem	45
4.3.8.	IObstacle	45
4.3.9.	Obstacle	45
4.3.10.	ITower	46
4.3.11.	Tower	46
4.3.12.	Map	47
4.3.13.	Cell	47

4.3.14. Field	48
4.3.15. Path	48
4.3.16. Towerhez tartozó krsitályok: Range, Speed, Damage, EnemyType	49
4.3.17. Obstaclehez tartozó kristályok: Intensity, Repair	50
4.3.18. IOGem	50
4.3.19. ITGem	50
4.3.20. IFieldPlaceable	50
4.3.21. IPathPlaceable	51
4.4. Szekvencia diagramok	51
4.4.1. Akadály elhelyezése	51
4.4.2. Ellenfél mozgása	52
4.4.3. Torony eladása	53
4.4.4. Torony elhelyezése	54
4.4.5. Torony fejlesztése	55
4.4.6. Torony tüzelése	56
4.4.7. Inicializálás	57
4.4.8. Játék menete	58
4.5. State-chartok	59
4.6. Napló	59

5 Szkeleton tervezése 60

5.1. A szkeleton modell valóságos use-case-ai	60
5.1.1. Use-case diagram	60
5.1.2. Use-case leírások	61
5.2. A szkeleton kezelői felületének terve, dialógusok	63
5.2.1. Kezelőfelület	63
5.2.2. Teszesetek	63
5.3. Szekvencia diagramok a belső működésre	65
5.3.1. 1. Akadály elhelyezése ellenség által foglalt útra	65
5.3.2. 2. Akadály elhelyezése másik akadály által foglalt útra	66
5.3.3. 3. Akadály elhelyezése üres útra	67
5.3.4. 4. Akadály fejlesztése	68
5.3.5. 5. Ellenfél mozgása	69
5.3.6. 6. Fejlesztetlen torony eladása	70
5.3.7. 7. Fejlesztett torony eladása	71
5.3.8. 8. Inicializálás	72
5.3.9. 9. Torony elhelyezése foglalt mezőre	73
5.3.10. 10. Torony elhelyezése üres mezőre	74
5.3.11. 11. Torony fejlesztése minden gemből eggel	75
5.3.12. 12. Torony hatókörében nincs ellenség tüzeléskor	76
5.3.13. 13. Torony lelő egy ellenséget fejlesztetlenül	77
5.3.14. 14. Torony tüzel egy ellenségre fejlesztetlenül	77
5.4. Kommunikációs diagramok	78
5.4.1. 1. Akadály elhelyezése ellenség által foglalt útra	78
5.4.2. 2. Akadály elhelyezése másik akadály által foglalt útra	78
5.4.3. 3. Akadály elhelyezése üres útra	79
5.4.4. 4. Akadály fejlesztése	79
5.4.5. 5. Ellenfél mozgása	80
5.4.6. 6. Fejlesztetlen torony eladása	80
5.4.7. 7. Fejlesztett torony eladása	81
5.4.8. 8. Inicializálás	81

5.4.9. 9. Torony elhelyezése foglalt mezőre	82
5.4.10. 10. Torony elhelyezése üres mezőre	82
5.4.11. 11. Torony fejlesztése minden gemből eggel	83
5.4.12. 12. Torony hatókörében nincs ellenség tüzeléskor	83
5.4.13. 13. Torony lelő egy ellenséget fejlesztetlenül	84
5.4.14. 14. Torony tüzel egy ellenségre fejlesztetlenül	85
5.5. Napló	85
6 Szkeleton beadás	87
6.1. Fordítási és futtatási útmutató	87
6.1.1. Fájllista	87
6.1.2. Fordítás	87
6.1.3. Futtatás	88
6.2. Értékelés	88
6.3. Napló	88
7 Prototípus koncepciója	89
7.0. Specifikáció változás	89
7.0.1. Módosítás	89
7.0.2. Módosított osztálydiagram	90
7.0.3. Módosított szekvenciadiagramok	90
7.1. Prototípus interface-definíciója	92
7.1.1. Az interfész általános leírása	92
7.1.2. Bemeneti nyelv	92
7.1.3. Kimeneti nyelv	94
7.2. Összes részletes use-case	96
7.3. Tesztelési terv	98
7.4. Tesztelést támogató segéd- és fordítóprogramok specifikálása	98
7.5. Napló	99
8 Részletes tervez	100
8.1. Osztályok és metódusok tervei	100
8.1.1. Bullet	100
8.1.2. Enemy	100
8.1.3. Enemy subclasses: Elf, Hobbit, Dwarf, Human	101
8.1.4. Game	101
8.1.5. Controller	102
8.1.6. IGame	103
8.1.7. Gem	103
8.1.8. IObstacle	104
8.1.9. Obstacle	104
8.1.10. ITower	105
8.1.11. Tower	105
8.1.12. Map	106
8.1.13. Cell	106
8.1.14. Field	107
8.1.15. Path	108
8.1.16. Towerhez tartozó krsitályok: Range, Speed, Damage, EnemyType	108
8.1.17. Obstaclehez tartozó kristályok: Intensity, Repair	109
8.1.18. IOGem	109
8.1.19. ITGem	109

8.1.20. IFieldPlaceable	110
8.1.21. IPathPlaceable	110
8.2. A tesztek részletes tervei, leírásuk a teszt nyelvén	110
8.2.1. Ellenség mozgás és sebzés	110
8.2.2. Akadály teszt	111
8.2.3. Kód, és ellenség kettészelés	112
8.2.4. Ellenség elágazáshoz ér	113
8.3. A tesztelést támogató programok tervei	115
8.4. Napló	115
10 Prototípus beadása	116
10.1. Fordítási és futtatási útmutató	116
10.1.1. Fájllista	116
10.1.2. Fordítás	117
10.1.3. Futtatás	117
10.2. Tesztek jegyzőkönyvei	118
10.2.1. Teszteset1	118
10.2.2. Teszteset2	118
10.2.3. Teszteset3	118
10.2.4. Teszteset4	119
10.3. Értékelés	119
10.4. Napló	119
11 Grafikus felület specifikációja	121
11.1. A grafikus interfész	121
11.2. A grafikus rendszer architektúrája	122
11.2.1. A felület működési elve	122
11.2.2. A felület osztály-struktúrája	123
11.3. A grafikus objektumok felsorolása	124
11.3.1. GPath	124
11.3.2. GField	124
11.3.3. GGame	125
11.3.4. GEnemy és leszármazottai: GHobbit, GHuman, GElf, GDwarf	125
11.3.5. Controller	126
11.3.6. GController	126
11.3.7. IView	126
11.3.8. Graphic	127
11.3.9. Cell	127
11.3.10. Game és IGame	127
11.4. Kapcsolat az alkalmazói rendszerrel	128
11.4.1. Akadály elhelyezése	128
11.4.2. Akadály javítása	129
11.4.3. Akadály törlődése	129
11.4.4. Betöltés	130
11.4.5. Ellenség beérkezik	131
11.4.6. Ellenség halála	131
11.4.7. Ellenség lépése	132
11.4.8. Ellenség létrehozása	133
11.4.9. Ellenség sebzése	134
11.4.10. Torony elhelyezése	135
11.4.11. Torony fejlesztése ellenség típusra	135

11.4.12.Torony törlődése	136
11.4.13.Varázserő változása	136
11.5. Napló	136
13 Grafikus felület specifikációja	138
13.1. Fordítási és futtatási útmutató	138
13.1.1. Fájllista	138
13.1.2. Fordítás	139
13.1.3. Futtatás	139
13.2. Értékelés	139
13.3. Napló	140
14 Összefoglalás	141
14.1. Projekt összegzés	141

Ábrák jegyzéke

2.1. Grafikus interfész	16
3.1. Osztálydiagram	22
3.2. Akadály elhelyezése szekvenciadiagram	30
3.3. Ellenfél mozgása szekvenciadiagram	31
3.4. Torony eladása szekvenciadiagram	32
3.5. Torony elhelyezése szekvenciadiagram	33
3.6. Torony fejlesztése szekvenciadiagram	34
3.7. Torony tüzelése szekvenciadiagram	35
3.8. Field állapotdiagram	36
3.9. Path állapotdiagram	37
4.1. Osztálydiagram	41
4.2. Akadály elhelyezése szekvenciadiagram	51
4.3. Ellenfél mozgása szekvenciadiagram	52
4.4. Torony eladása szekvenciadiagram	53
4.5. Torony elhelyezése szekvenciadiagram	54
4.6. Torony fejlesztése szekvenciadiagram	55
4.7. Torony tüzelése szekvenciadiagram	56
4.8. Inicializálás szekvenciadiagram	57
4.9. Játék menete szekvenciadiagram	58
5.1. Játékos use-case diagram 1	60
5.2. Játékos use-case diagram 2	60
5.3. Akadály elhelyezése ellenség által foglalt útra szekvenciadiagram	65
5.4. Akadály elhelyezése másik akadály által foglalt útra szekvenciadiagram	66
5.5. Akadály elhelyezése üres útra szekvenciadiagram	67
5.6. Akadály fejlesztése szekvenciadiagram	68
5.7. Ellenfél mozgása szekvenciadiagram	69
5.8. Fejlesztetlen torony eladása szekvenciadiagram	70
5.9. Fejlesztett torony eladása szekvenciadiagram	71
5.10. Inicializálás szekvenciadiagram	72
5.11. Torony elhelyezése foglalt mezőre szekvenciadiagram	73
5.12. Torony elhelyezése üres mezőre szekvenciadiagram	74
5.13. Torony fejlesztése minden gemből egygyel szekvenciadiagram	75
5.14. Torony hatókörében nincs ellenség tüzeléskor szekvenciadiagram	76
5.15. Torony lelő egy ellenséget fejlesztetlenül szekvenciadiagram	77
5.16. Torony tüzel egy ellenségre fejlesztetlenül szekvenciadiagram	77
5.17. Akadály elhelyezése ellenség által foglalt útra kommunikáviós diagram	78
5.18. Akadály elhelyezése másik akadály által foglalt útra kommunikáviós diagram	78
5.19. Akadály elhelyezése üres útra kommunikáviós diagram	79
5.20. Akadály fejlesztése kommunikáviós diagram	79
5.21. Ellenfél mozgása kommunikáviós diagram	80
5.22. Fejlesztetlen torony eladása kommunikáviós diagram	80
5.23. Fejlesztett torony eladása kommunikáviós diagram	81
5.24. Inicializálás kommunikáviós diagram	81

5.25. Torony elhelyezése foglalt mezőre kommunikáviós diagram	82
5.26. Torony elhelyezése üres mezőre kommunikáviós diagram	82
5.27. Torony fejlesztése minden gemből egygyel kommunikáviós diagram	83
5.28. Torony hatókörében nincs ellenség tüzeléskor kommunikáviós diagram	83
5.29. Torony lelő egy ellenséget fejlesztetlenül kommunikáviós diagram	84
5.30. Torony tüzel egy ellenségre fejlesztetlenül kommunikáviós diagram	85
7.1. Módosított osztálydiagram	90
7.2. Játék menete szekvenciadiagram	91
7.3. Torony kettévág egy ellenséget szekvenciadiagram	92
11.1. Grafikus interfész	121
11.2. Grafikus rendszer struktúra diagram	122
11.3. Osztály diagram	123
11.4. Akadály elhelyezése szekvenciadiagram	128
11.5. Akadály javítása szekvenciadiagram	129
11.6. Akadály törlődése szekvenciadiagram	129
11.7. Betöltés szekvenciadiagram	130
11.8. Ellenség beérkezik szekvenciadiagram	131
11.9. Ellenség halála szekvenciadiagram	131
11.10. Ellenség lépése szekvenciadiagram	132
11.11. Ellenség létrehozása szekvenciadiagram	133
11.12. Ellenség sebzése szekvenciadiagram	134
11.13. Torony elhelyezése szekvenciadiagram	135
11.14. Torony fejlesztése ellenség típusra szekvenciadiagram	135
11.15. Torony törlődése szekvenciadiagram	136
11.16. Varázserű változása szekvenciadiagram	136

2. Követelmény, projekt, funkcionalitás

2.1. Bevezetés

2.1.1. Cél

A dokumentum célja az elkészülendő szoftver bemutatása fejlesztői szemszögből, ami a szoftver fejlesztői dokumentációja, amely segítségével megismerhető a szoftver felépítése és fejlesztésének lépései.

2.1.2. Szakterület

A kialakítandó szoftver egy játék, amely kizártlag szórakoztatási célokat szolgál. A fejlesztők célja egy olyan szoftver létrehozása, amely megfelel a feladatkiírásban foglalt követelményeknek.

2.1.3. Definíciók, rövidítések

HSZK Hallgatói Számítógép Központ

szoftver a program amit írunk

2.1.4. Hivatkozások

- http://en.wikipedia.org/wiki/Tower_defense
- [http://en.wikipedia.org/wiki/The_Lord_of_the_Rings_\(film_series\)](http://en.wikipedia.org/wiki/The_Lord_of_the_Rings_(film_series))

2.1.5. Összefoglalás

A dokumentum további részei az analízis modell, ami a szoftver főbb osztályainak, alkotóelemeinek tárgyalása, a szkeleton, amiben a belső működést tárgyaljuk.

A prototípus részben a részletes terveket, use-case-eket lehet majd megtalálni.

A szoftver grafikus tulajdonságait, interfészét, a végső fázisban dokumentáljuk.

2.2. Áttekintés

2.2.1. Általános áttekintés

A szoftver kialakítandó architektúrája objektum-orientált, modell-nézet-vezérlő mintát követ.

A modell fontosabb alrendszeréi:

- pálya,
- tornyok,
- akadályok,
- ellenségek.

Az ellenségek, illetve a tornyok és akadályok közötti kapcsolat lényege, hogy a tornyok tudják, mikor ér a hatókörükbe egy ellenség. A tornyoknak tudniuk kell értesíteni az ellenségeket, amikor tüzet nyitnak, ami alapján az ellenségnak változik az életereje.

A pálya és az ellenségek közötti kapcsolat lényege, hogy az ellenségek ismerjék a pályán lévő utat, tudják azt követni.

2.2.2. Funkciók

Az eredeti feladatkiírás szövege

A gonosz emberek, tündék, törpök és hobbitok szövetséget kötnek, hogy elpusztítsák az Egy Gyűrűt a Végzet Hegyénél. Szerencsére csak Mordor földjén keresztül tudnak eljutni a hegyhez, így jóságos Szarumánnak lehetősége van védelmi tornyokat építeni, hogy segítsen megvédeni Szauron hatalmát. A játék célja annak megakadályozása, hogy a Gyűrű szövetségének tagjai közül bárki is eljusson a Végzet Hegyéhez. Egy ellenség akkor pusztul el, ha összességében megfelelő mértékű sebzést kap a tornyokból származó lövedékektől. A tornyok építéséhez Szarumánnak a varázserejét kell használnia. Szarumán akkor tud tornyot építeni, ha megfelelő mennyiségű varázsereje van hozzá. A varázsereje minden egyes elpusztított ember, tünde, törp vagy hobbit után bizonyos mértékben növekszik.

A Gyűrű szövetségének tagjai különböző utakon juthatnak el a Végzet Hegyéhez. Az utakról nem térhetnek le. Szarumán az utakra nem tud tornyot építeni, csak az utak mellé. Az utakra azonban lehet akadályokat, amik az akadály területén lassítják az ellenség haladását. A tornyoknak van egy adott hatótávolsága és tüzelési gyakorisága. Szarumán a varázserejét arra is használhatja, hogy a tornyokat és akadályokat különböző varázskövekkel ruházza fel. A varázsköveknek több fajtája is létezik, és különböző hatásúak lehetnek. Egyes kövek növelhetik a tornyok hatótávolságát vagy tüzelési gyakoriságát, más kövek egy-egy típusú ellenfél esetén megnövelik a lövedékek sebzési erejét.

A játék során az ellenségek folyamatosan jönnek. A játék elején ritkábban, később gyakrabban és nagyobb csoportokban, azonban számuk véges, előbb-utóbb elfogynak. A játék akkor ér véget, ha egy ellenség eljut a Végzet Hegyéhez, vagy ha már sikerült az összes ellenséget kiirtani. Az első esetben Szauron és Szarumán megsemmisül, utóbbi esetben fényes győzelmet aratnak, és örökké uralni fogják a világot.

Feladat specifikációja

A feladat egy grafikus felületű logikai játékszoftver készítése Java nyelven, ami megfelel a fenti leírás szövegének, továbbá működik a HSzK számítógépein.

A szoftver elindítása után a kezdőképernyő fogadja a felhasználót, ahol új játékot indíthat a megfelelő gombbal. Az új játék indítása esetén a felhasználónak ki kell választania egy pályát. Ha ez megtörtént, elindul a játék.

A pályák téglalap alakúak és különböző méretűek lehetnek. A pályák egy $n*m$ méretű négyzetrácsból állnak, ahol m a sorok n pedig az oszlopok számát jelenti.

A pálya négyzetrácsos szerkezetének a celláira a játékos védelmi egységeket helyezhet varázserőért cserébe. A cellák két típusúak lehetnek: út, és háttér. A hátteret alkotó cellákra tornyokat lehet helyezni, az út celláira pedig akadályokat.

A tornyoknak három jellemzője van: tüzelési sebesség, sebzés és hatótáv. Amint egy ellenség egy torony hatókörébe ér, az tüzet nyit rá, azonban egy torony egyszerre csak egy célpontra tüzelhet. Lehetséges a tornyokat fejleszteni is: két hely van minden toronyban, amibe a torony további képességeit, azaz a hatótávolságot és a tüzelési gyorsaságot fokozó kristályokat helyezhetünk. Végül pedig minden torony esetén lehetőség van egy darab, az egyes fajok elleni sebzést növelő kristályt helyezni.

Az út celláira elhelyezhető akadályok lassítják a felettük elhaladó ellenségeket, viszont elhelyezésük után használódnak minden ellenség lassításakor, még végül meg nem szűnnek. Az akadályok jellemzői, hogy mennyi ellenség haladhat át rajtuk és a lassítás mértéke. Az akadályokat is lehetséges fejleszteni: lehet növelni a lassítás mértékét, illetve meg lehet javítani az akadályt.

A tornyok és akadályok elhelyezése úgy történik, hogy a játékos a kívánt cellára kattint, majd a felkínált lehetőségek közül kiválasztja az elhelyezendő tornyot, illetve akadályt.

A védelmi egységeket el is adhatja a játékos, ekkor az arra fordított varázserőnek, tehát a vásárlás és az esetleges fejlesztések árának a fele visszatérül. Az eladásához először rá kell kattintani az eladni kívánt védelmi egység képére, majd a megjelenő lehetőségek közül ki kell választani az eladást. Az akadályok esetén a visszajáró varázserő az elhasználtság mértékétől is függ: a ráfordított költségek felét, tehát ami egy torony

esetén visszajárna, még meg kell szorozni az ellenségek számának, amik még áthaladhatnak az akadályon, és az ellenségek számának, amik összesen áthaladhatnak az akadályon, hánnyadosával.

A tornyok és akadályok fejlesztéséhez a felhasználó kiválasztja a fejleszteni kívánt egységet a rajta való kattintással, és megjelennek az azzal kapcsolatos műveletek, amelyek között szerepel az egység különböző módú fejlesztése is. minden fejlesztés varázserőbe kerül.

Egy ellenségnak két jellemzője van: a sebessége és az életereje. A játékban négy különböző ellenfél van: emberek, tündék, török és hobbitok. minden fajnak különböző jellemzőik vannak: az emberek sebessége és életereje közepe, a török sebessége lassú, életerejük magas, a tündék sebessége gyors, életerejük magas, a hobbitok sebessége lassú, életerejük pedig alacsony. Egy ellenség megölése után növekedik egy adott mennyiséggel a játékos varázsereje. Ez a megölt ellenség életerejétől függ.

Az ellenségek a belépési pontnál, vagy pontoknál érkeznek a játék kezdetétől a végéig, és a Végzet hegye felé tartanak. Az útvonalon lehetnek elágazások is, amikhez érve az ellenségek véletlenszerűen haladnak tovább a Végzet Hegye felé. A hegyet Szarumán varázslata védi, ami azonnal végez az odaérkezőkkel, azonban minden megölt ellenség után gyengül, míg végül meg nem szűnik. A hegyet védő varázslat ereje pályánként változik. A varázslat megszűnése után az első bejutó ellenséggel a játék véget ér.

A játékos akkor nyer, ha megöl minden ellenséget, és egy sem jut be a hegybe.

2.2.3. Felhasználók

A felhasználókról (játékosokról) a továbbiakban, az általanosság megszorítása nélkül, néhány tulajdonságot feltételezünk. A szoftver (játék) interaktív mivolta miatt a felhasználóról feltételezhető, hogy képes elsajátítani a meglévő leírások, útmutatók, illetve a menük és fülek alapján a szoftver használatát. Korbéli megkötés nincs, azonban részéről elvártak bizonyos alapszintű ismeretek a számítógép, illetve az ahhoz kapcsolódó inputok (billentyűzet, egér) kezeléséről. Továbbá feltételezett részéről a szoftver rendeltetésszerű használata.

2.2.4. Korlátozások

Az elkészült szoftvernek bizonyos, nem funkcionális, korlátoknak eleget kell tennie.

Legfontosabb ezek közül, hogy tudnia kell futni a HSZK-s gépeken. Ebből kifolyólag a különböző hardveres és szoftveres megkötések szempontjából ezen gépek paramétereit mérvadóak. A szoftvert Java 1.6 nyelven kell implementálni. A szoftvernek parancssorból futtathatónak és hordozhatóak kell lennie. A grafikai megjelenést tekintve egy adott, a képernyőn lévő, elemről a felhasználó számára egyértelműen előnthatónek kell lennie, hogy mi a szerepe, célja a szoftverben.

2.2.5. Feltételezések, kapcsolatok

A dokumentáció olvasójáról feltételezzük, hogy ismeri az alap koncepcióját a torony védés játékoknak, illetve ismeri A Gyűrűk Ura trilógia világát.

2.3. Követelmények

2.3.1. Funkcionális követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Use-case	Komment
F2.3.1.1	A szoftver elindítása után megjelenő főmenüből lehessen új játékot indítani	Funkcionális tesztelés	alapvető	feladat-leírás	Új játék	

F2.3.1.2	Lehessen pályát választani az új játék kezdése után	Funkcionális tesztelés	fontos	csapat	Pálya kiválasztása	
F2.3.1.3	A pályán lehessen tornyokat és akadályokat elhelyezni	Funkcionális tesztelés	alapvető	feladat-leírás	Torony/Akadály elhelyezése	
F2.3.1.4	A pályán elhelyezett tornyokat és akadályokat lehessen fejleszteni	Funkcionális tesztelés	alapvető	feladat-leírás	Torony/Akadály fejlesztése	
F2.3.1.5	A pályán elhelyezett tornyokat és akadályokat el lehessen adni	Funkcionális tesztelés	fontos	csapat	Torony/Akadály eladása	
F2.3.1.6	A szoftver ne fagyjon le játék közben, működjen helyesen	Funkcionális tesztelés	alapvető	csapat		

2.3.2. Erőforrásokkal kapcsolatos követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
E2.3.2.1	Java 1.6 futtató környezet	Futtatás előtt feltelepítettség ellenőrzése	Fontos	HSZK gépkövetelmény	
E2.3.2.2	Operációs rendszer: Windows XP vagy újabb vagy Linux vagy Mac OS X 10.7.3 vagy újabb	Felhasználó ellenőrzi	Fontos	Java 1.6 követelmény	

2.3.3. Átadással kapcsolatos követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
A2.3.3	A szoftvert 3 fázisban adjuk át: szkeleton, prototípus és grafikus változat. Mindhárom esetben forráskódot adunk át, amihez könnyen érhető telepítési útmutatót csatolunk.	Budai Péter labорvezető ellenőrzi.	Fontos		A mellékelt útmutató segítségével egy általános számítógépes ismeretekkel rendelkező felhasználó telepíteni tudja a szoftvert. Szükség esetén a telepítést vállaljuk.

2.3.4. Egyéb nem funkcionális követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
N2.3.4.1	Bárki számára könnyen kezelhető legyen a szoftver.		Fontos	csapat	A szoftverrel mellé tartozó README fájl utasításai szerint, JAVA JDK 1.6 alatt fordíthatónak és futtathatónak kell lennie.
N2.3.4.2	Bemenetekre egy-ből kell regálni a rendszernek.	Tesztelés a minimum gépigénynek a környékén.	Fontos	általános	
N2.3.4.3	Billentyűzet vagy egér segítségével kezelhető legyen		Alapvető	általános	
N2.3.4.4	Hordozhatóság	A szoftvert teszteljük több operációs rendszeren.	Fontos	feladat-leírás	Windows, Linux, OS X
N2.3.4.5	Terhelhetőség	Olyan tesztesetekkel amik sok objektumot használnak.	opcionális	csapat	
N2.3.4.6	Megfelelő dokumentáltság, többek közt karbantarthatóság szempontjából	Projektvezető (Budai Péter)	Alapvető	Projektvezető	

2.4. Lényeges use-case-ek

2.4.1. Use-case leírások

Use-case neve	Új játék
Rövid leírás	Új játék kezdése
Aktorok	Játékos
Forgatókönyv	A játékos a főmenüben kiválasztja az „Új játék kezdése” gombot. Ekkor megjelenik a pálya kiválasztása.

Use-case neve	Pálya kiválasztása
Rövid leírás	Pálya kiválasztása pályák közül
Aktorok	Játékos
Forgatókönyv	A játékos rákattint a kiválasztandó pálya névére. Ezután megjelenik a kiválasztott pálya és elkezdődik a játék.

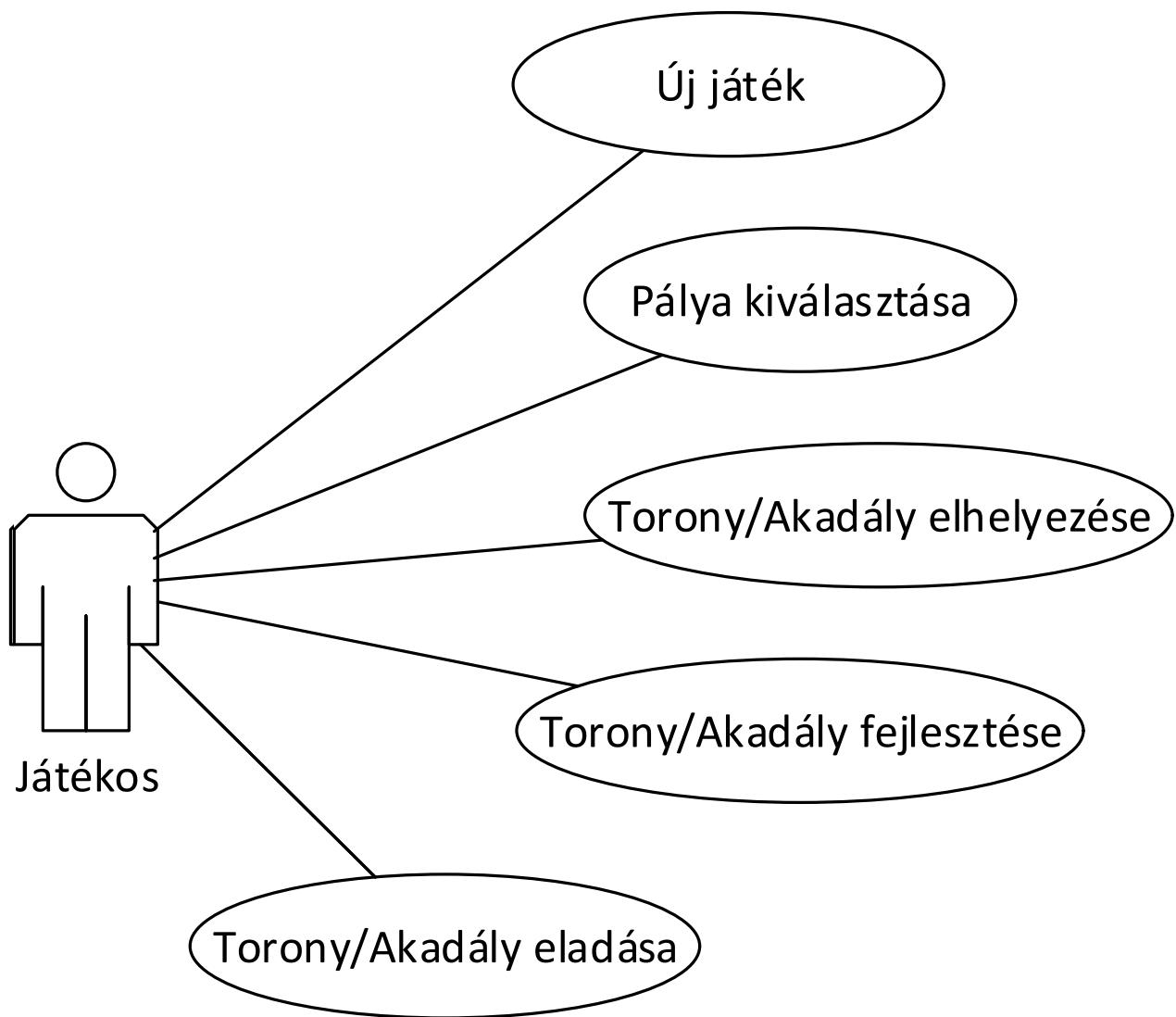
Use-case neve	Torony/Akadály elhelyezése
Rövid leírás	Torony vagy Akadály elhelyezése az arra kijelölt helyeken
Aktorok	Játékos

Forgatókönyv	A játékos az arra kijelölt helyre kattintva választhatja ki az elhelyezni kívánt tornyot vagy akadályt amennyiben van rá elég varázsereje.
--------------	--

Use-case neve	Torony/Akadály fejlesztése
Rövid leírás	Korábban elhelyezett torony/akadály hatékonyságának növelése
Aktorok	Játékos
Forgatókönyv	A játékos a már elhelyezett tornyot vagy akadályt fejlesztheti, hatékonyságát növelheti bizonyos mértékben, amennyiben van rá elég varázsereje.

Use-case neve	Torony/Akadály eladása
Rövid leírás	Korábban elhelyezett torony/akadály eladása
Aktorok	Játékos
Forgatókönyv	A játékos a már elhelyezett tornyot vagy akadályt eladhatja, és így visszakapja a ráfordított költségek egy részét.

2.4.2. Use-case diagram



2.1. ábra. Grafikus interfész

2.5. Szótár

akadály (obstacle) az út celláira helyezett védelmi egység, a felette haladó ellenségek lassabban haladnak

belépési pont (entry point) az a hely, ahol az ellenségek belépnek a pályára

cella (cell) a pálya négyzetrácsos szerkezetét alkotó elemek

eladás (sell) egy védelmi egység megszűntetése, ami az arra költött varázserő egy részének visszatérülésével jár

elhasználódás (attrition) a csapdák amortizációja, amit a rajtuk áthaladó ellenségek okoznak

elhelyezés (placing) egy védelmi egység a pálya egy bizonyos cellájára való telepítése varázserőért cserébe

ellenség (enemy) azok az egységek, amik a pálya belépési pontjainál jönnek be, és a Végzet Hegye felé haladnak. Fajtái: tündér, törp, hobbit, ember

ember(human) az ellenség egy fajtája, sebessége közepes, életereje közepes

életerő (health) ellenségeket jellemző érték, amit a tornykból kilött lövedékekkel lehet csökkenteni. Ha eléri a 0-t az ellenség meghal.

faj: ellenség típusok tündér, törp, hobbit, ember

fejlesztés (upgrade) egy védelmi egység hatékonyabbá tétele kristállyal

gépigény olyan számítógépes hardverek és nem hardveres erőforrások, amelyek minimálisan szükségesek a szoftver telepíthetőségéhez, rendes működéséhez

háttér (background) a pálya úton kívüli része, ide helyezhetjük a tornyokat

hobbit(hobbit) az ellenség egy fajtája, sebessége lassú, életereje alacsony

játékos (player) a szoftver felhasználója

kristály (gem) tornyok/akadályok fejlesztésére alkalmas varázseszköz

kristály tartó(gem slot) a tornyok kristályokat tároló helyei

nyerni (win) az akkor fennálló helyzet, mikor elfogynak az ellenségek, és egy sem jutott be a Végzet Hegyébe

pálya (map) játéktér, legalább 1 út van rajta, minden más része háttér

readme olyan fájl, amely információkat tartalmaz a szoftver telepítéséről és használatba vételéről

sebez (damage) csökkenti az ellenség életterejét

szótár ez a dokumentum

torony (tower) a háttér celláira helyezett védelmi egység, automatikusan tüzel az ellenségre

törp(dwarf) az ellenség egy fajtája, sebessége lassú, életereje magas

tündér(elf) az ellenség egy fajtája, sebessége gyors, életereje magas

tüzel (fire) lövedéket kibocsát, mellyel ellenséget sebez meg

újratöltési idő(reload time) a torony két tüzelése között eltelt idő

út (way, road) a pálya azon része, ahol az ellenséges egységek haladnak, és az akadályokat helyezhetjük

varázserő (witchcraft) a játékban használt fizetőeszköz, cserébe lehet a védelmi egységeket elhelyezni, fejleszteni

védelmi egység torony vagy akadály

Végzet Hegye az ellenségek célpontja, az ide való eljutásuk megakadályozása a játékos célja

veszíteni (lose) az akkor fennálló helyzet, mikor egy ellenség bejut a Végzet Hegyébe

2.6. Projekt terv

2.6.1. Határidők

	Dátum	Feladat
1	febr. 14.	14 h - csapatok regisztrációja
2	febr. 24.	Követelmény, projekt, funkcionalitás - beadás
3	márc. 3.	Analízis modell kidolgozása 1. - beadás
4	márc. 10.	Analízis modell kidolgozása 2. - beadás
5	márc. 17.	Szkeleton tervezése - beadás
6	márc. 24.	Szkeleton - beadás
7	márc. 31.	Prototípus koncepciója - beadás
8	ápr. 7.	Részletes tervezés - beadás
9	ápr. 22.	Prototípus - beadás
10	ápr. 28.	Grafikus felület specifikációja - beadás
11	máj. 12.	Grafikus változat - beadás
12	máj. 16.	Összefoglalás - beadás

2.6.2. Átadás

A szoftvert három fő fázisban adjuk át:

1. Szkeleton. A szoftver váza, az átadott részletes specifikáció, és analízis modell alapján. Ez a verzió csak a modell működését szemlélteti, csak karakteres ki- és bemenetekkel.

Az objektumoknak csak az interfésze definiált.

Az egyes metódusok, kiírják a képernyőre nevüket és meghívják azon metódusokat, amelyeket a szolgáltatás végrehajtása érdekében meg kell hívniuk. Elágazás esetén feltesz egy kérdést, aminek a válasza alapján fut tovább a modell.

2. Prototípus. Elkészült szoftver minden tulajdonságával bír a prototípus, csak a grafikus megjelenítés hiányzik róla. A pillanatnyi állapotot alfanumerikus képernyőn lehet követni, illetve log fájlokba menti a futásának részleteit.

3. Grafikus változat. A prototípustól csak a megjelenésben különbözik, illetve annak belső megvalósításával bővített változata.

2.6.3. Szerepkörök

- Csapatvezető: Elekes Tamás
 - Értekezlet előkészületei, lebonyolítása
 - Project felügyelet
 - Kód tervezés
 - Implementálás
 - Audio programozás
- Fuksz Domonkos
 - Dokumentálás
 - Dokumentumok formai minősége

- Tesztelés
- Kódolás
- Nagy András
 - Értekezletek logolása
 - Implementálás
 - Tesztelés
- Seres Márk Dániel
 - Feladat beadás
 - Implementálás
 - Tesztelés
 - UML diagram készítés
- Rédey Bálint
 - Dokumentálás
 - Implementálás
 - Grafikus programozás
 - UML diagram készítés
 - Feladatok feltöltése

Ezen felül minden csapattagnak a konzultációkat látogatnia kell, illetve heti 1 teljes létszámú értekezleten részt kell vennie.

2.6.4. Felhasznált fejlesztőeszközök

Verziókezelés A csoportunk szempontjából nagyon fontos, hogy mindenki naprakészen hozzáférjen a kód és a dokumentumok aktuális verzióhoz. Ehhez a Git verziókezelő rendszert választottuk, ami felhő alapú, könnyen kezelhető rendszer.

Java JDK A szoftvert Java nyelven kerül implementálásra, amihez az ingyenes Eclipse szoftvert használjuk. Az Eclipse nagyon jó környezetet ad Java JUnit tesztelés végzésére is.

UML Fontos a jó minőségű dokumentáció elkészítéséhez az ábrák megfelelő és szép elkészítése. Erre a célról több UML rajzoló szoftvert is használunk: StarUML és Microsoft Office Visio 2013.

Kommunikáció A csapatunk során elengedhetetlen a megfelelő kommunikáció a tagok között. Egyszerű szöveges kommunikációra Facebook csoportot használunk, míg fontosabb dolgok tárgyalásakor a Skype konferenciabeszélgetést használjuk.

2.7. Napló

Kezdet	Időtartam	Részvevők	Leírás
2014.02.09. 16:00	1 óra	Seres Rédey Fuksz Nagy Elekes	Alakuló értekezlet. Ismerkedés, adatok egyeztetése.

Kezdet	Időtartam	Résznevők	Leírás
2014.02.19. 08:15	1 óra	Seres Rédey Fuksz Nagy Elekes	Konzultáció
2014.02.19. 09:20	2,5 óra	Seres Rédey Fuksz Nagy Elekes	Értekezlet. Döntés: Szerepkörök tisztázása. Fuksz dokumentálja a 2.1.1, 2.1.2, 2.3.2, 2.4.1 pontokat. Seres dokumentálja a 2.2.1, 2.2.2, 2.4.1, 2.3.1 pontokat. Rédey dokumentálja a 2.2.3, 2.2.4, pontokat. Nagy dokumentálja a 2.3.3 pontot. Elekes dokumentálja a 2.3.4, 2.6 pontokat. 2.1.3-5, 2.2.5, 2.5 pontokat az előző pontokkal végezve dokumentáljuk.
2014.02.19. 13:00	1 óra	Elekes	2.3.4, 2.6 pontok dokumentálása
2014.02.19. 14:15	45 perc	Rédey	2.2.3, 2.2.4 pontok dokumentálása
2014.02.19. 18:00	2 óra	Nagy	A dokumentum miért readonly? hibakeresés
2014.02.21. 16:00	1 óra	Seres	2.2.1, 2.2.2 pont dokumentálása
2014.02.21. 16:00	20 perc	Nagy	A 2.3.3 dokumentálása
2014.02.22. 18:30	15 perc	Fuksz	2.4.1 pont kitöltése
2014.02.22. 19:30	10 perc	Seres	2.3.1 pont kitöltése
2014.02.23. 15:45	1 óra	Fuksz	2.1.1, 2.1.2, 2.3.2, 2.4.2 pont kitöltése
2014.02.23. 17:00	40 perc	Elekes	2.1.5, 2.2.5 pont, szótár
2014.02.23. 19:00	10 perc	Nagy	2.1.3, 2.1.4 pont kitöltése

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) lelassítja. 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 (tündér, 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 műlásának 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 függvények. Ennek a lövedéknek a feladata, hogy az ellenségnak 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 referenciaját. 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.

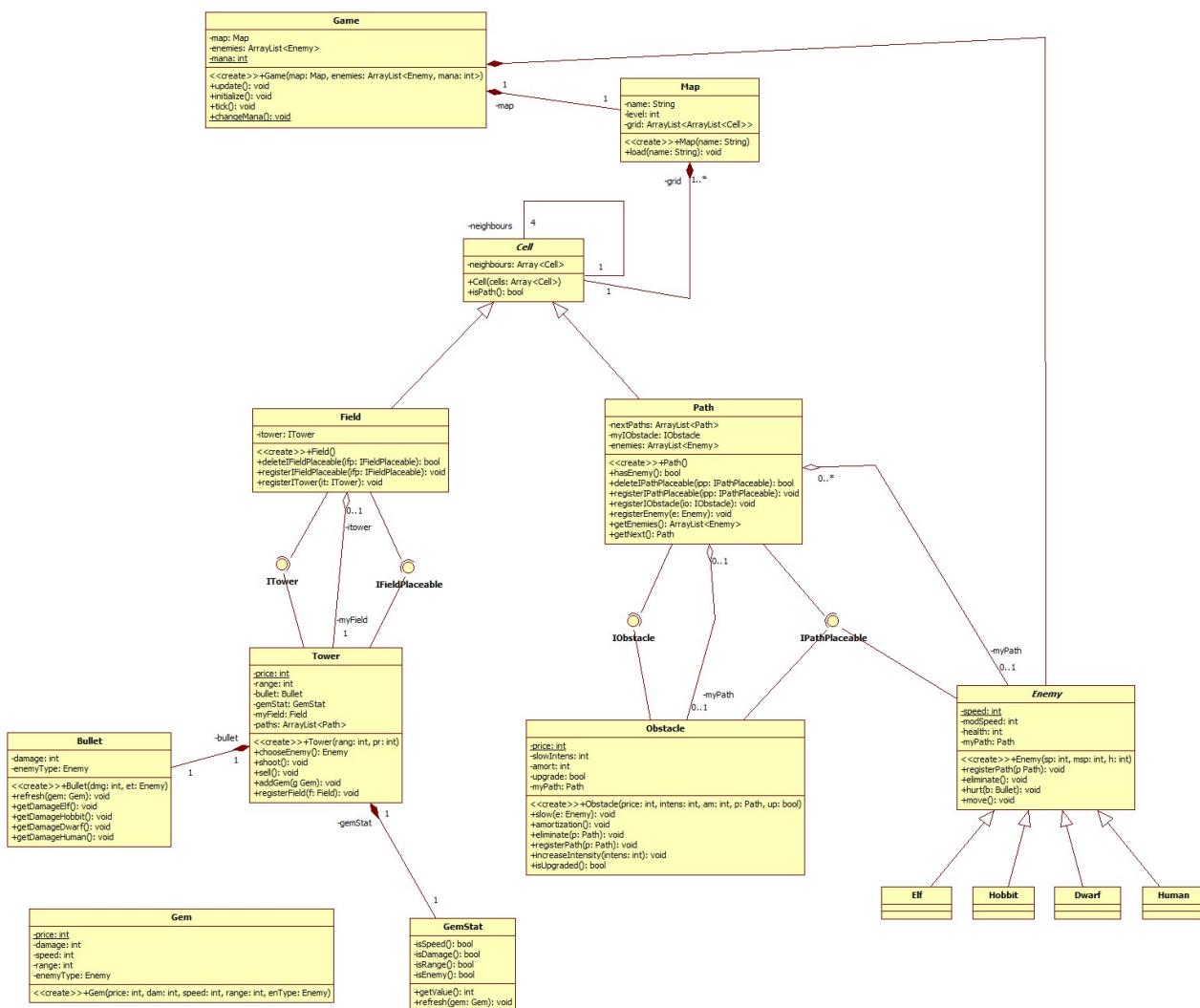
3.1.9. Torony (Tower)

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 űsosztály hurt metódusát.

- Űsosztályok

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 beregiszt- rálja 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.

- Űsosztályok

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): konstruktur

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 hatósugárövelő 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 között 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 intensity, 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
- Interfészök
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 intensity, Path p): lelassítja a p-n lévő ellenséget intensity-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): fejlesztéskor megnöveli a lassítás 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): konstruktur

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>): konstruktur, 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 → Cell

- Interfészek

Nincs

- Attribútumok

- ITower itower: a mezőn álló torony interfészű elem tárolása

- Metódusok

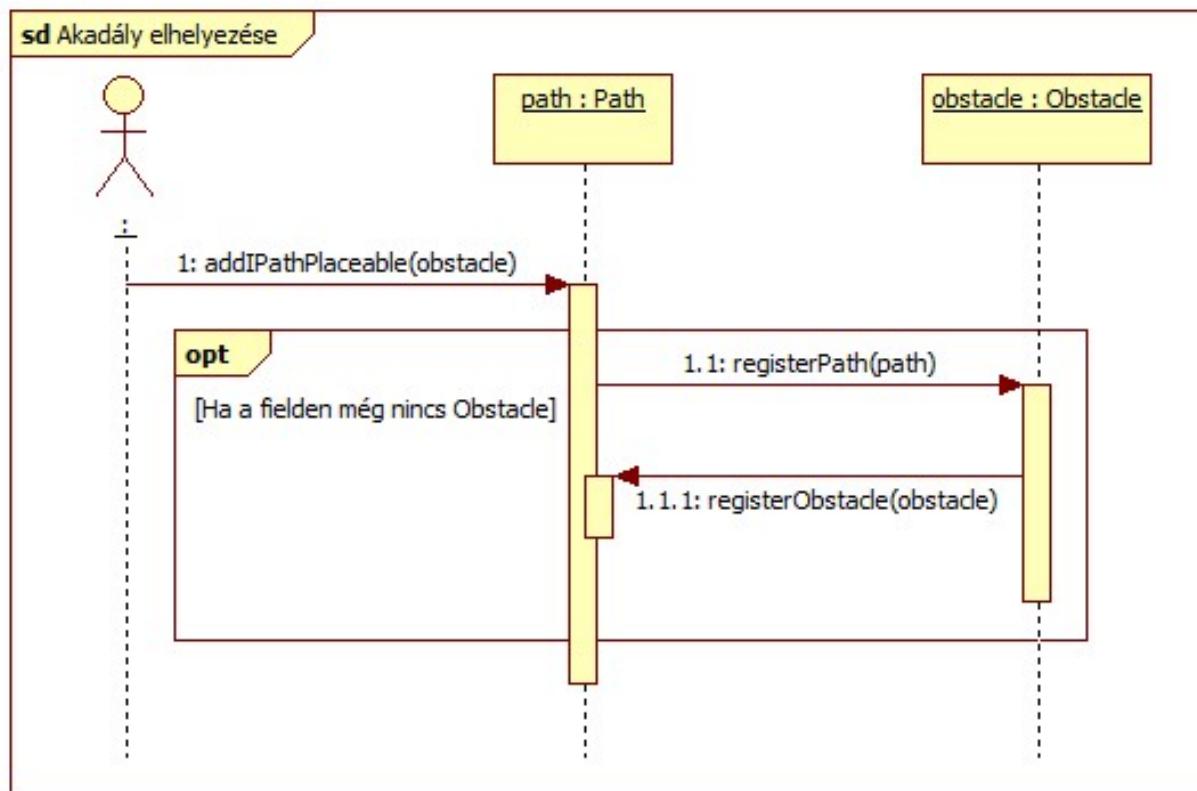
- bool isPath(): hamis értékkel tér vissza
- void addIFieldPlaceable(): egy új tornyot ad hozzá a mezőhöz
- void deleteIFieldPlaceable(IFieldPlaceable ifield): eltávolítja a tornyot a mezőről
- void registerITower(ITower itower): beteszi ifieldbe a kapott tornyot
- Field(): konstruktur

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 registerObstacle(Obstacle o): a o kapott akadály lesz az obstacle
 - ArrayList<Enemy> getEnemies(): visszatér az enemies-el
 - Path getNext(): paths-ből ad vissza egy elemet

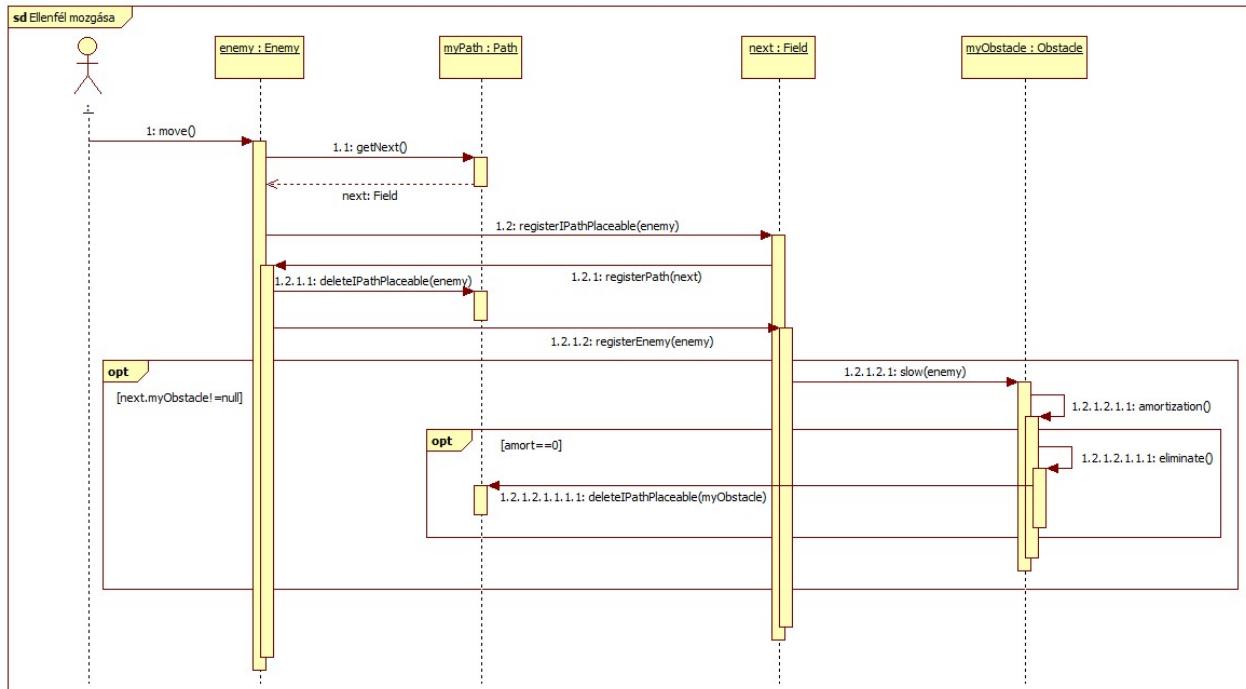
3.4. Szekvencia diagramok

3.4.1. Akadály elhelyezése



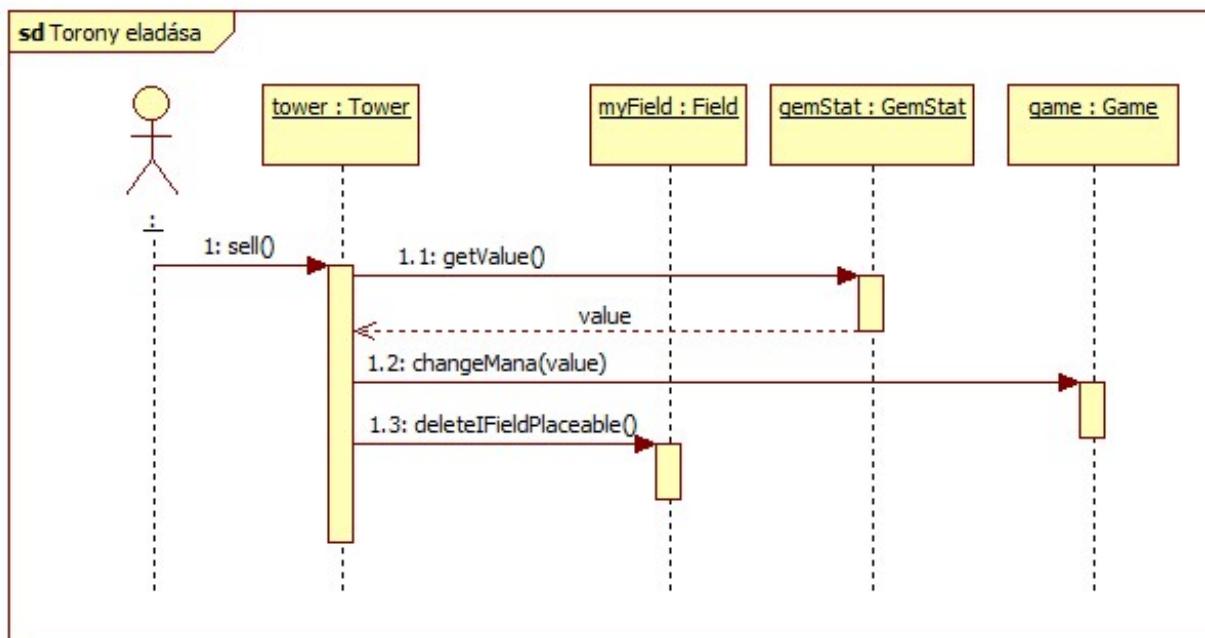
3.2. ábra. Akadály elhelyezése szekvenciadiagram

3.4.2. Ellenfél mozgása



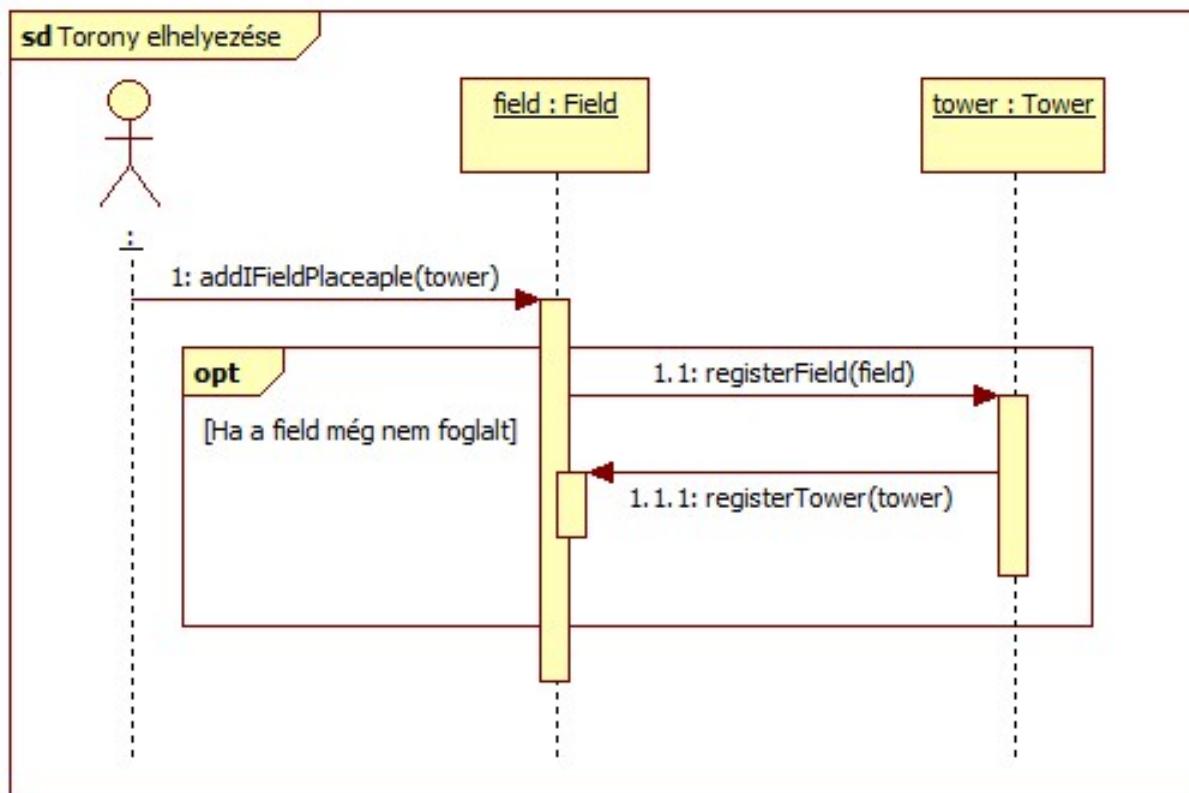
3.3. ábra. Ellenfél mozgása szekvenciadiagram

3.4.3. Torony eladása



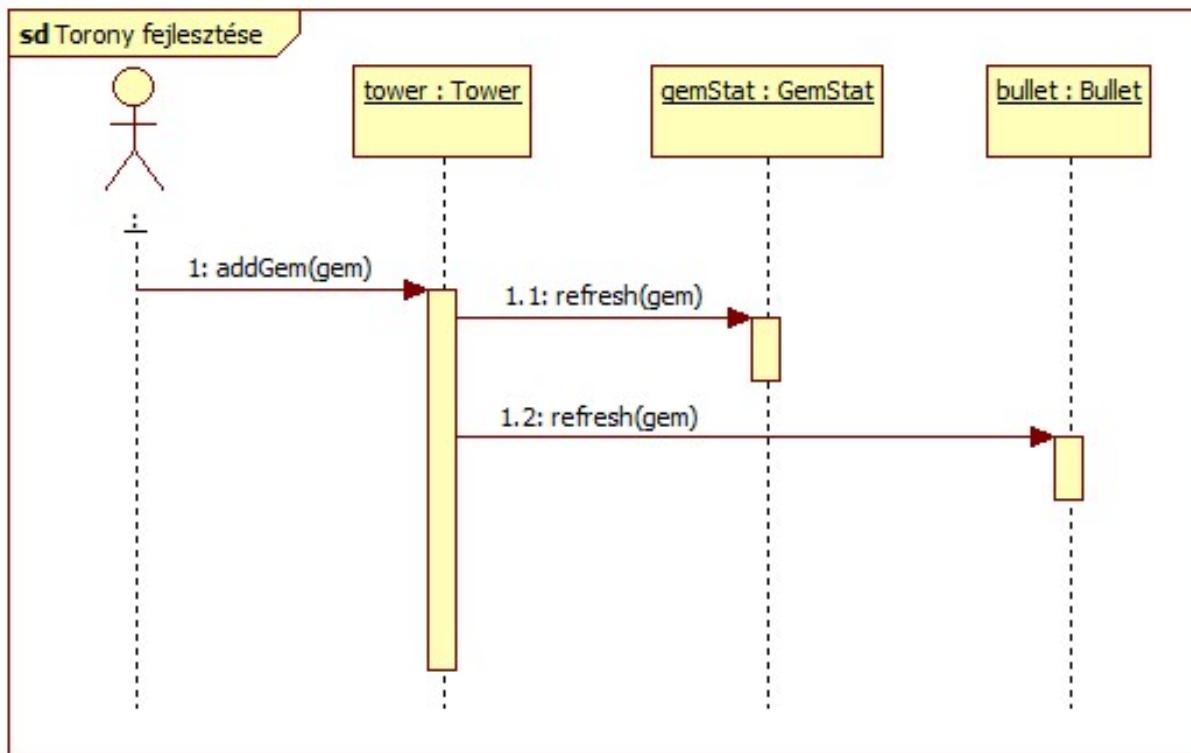
3.4. ábra. Torony eladása szekvenciadiagram

3.4.4. Torony elhelyezése



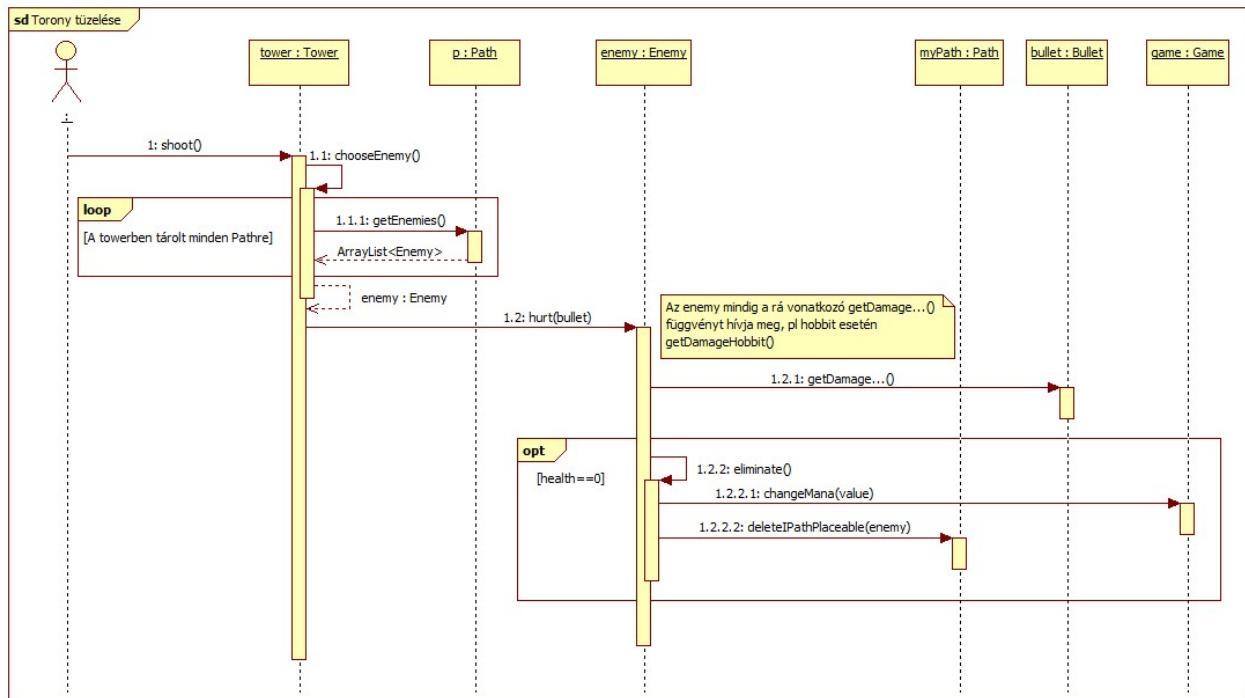
3.5. ábra. Torony elhelyezése szekvenciadiagram

3.4.5. Torony fejlesztése



3.6. ábra. Torony fejlesztése szekvenciadiagram

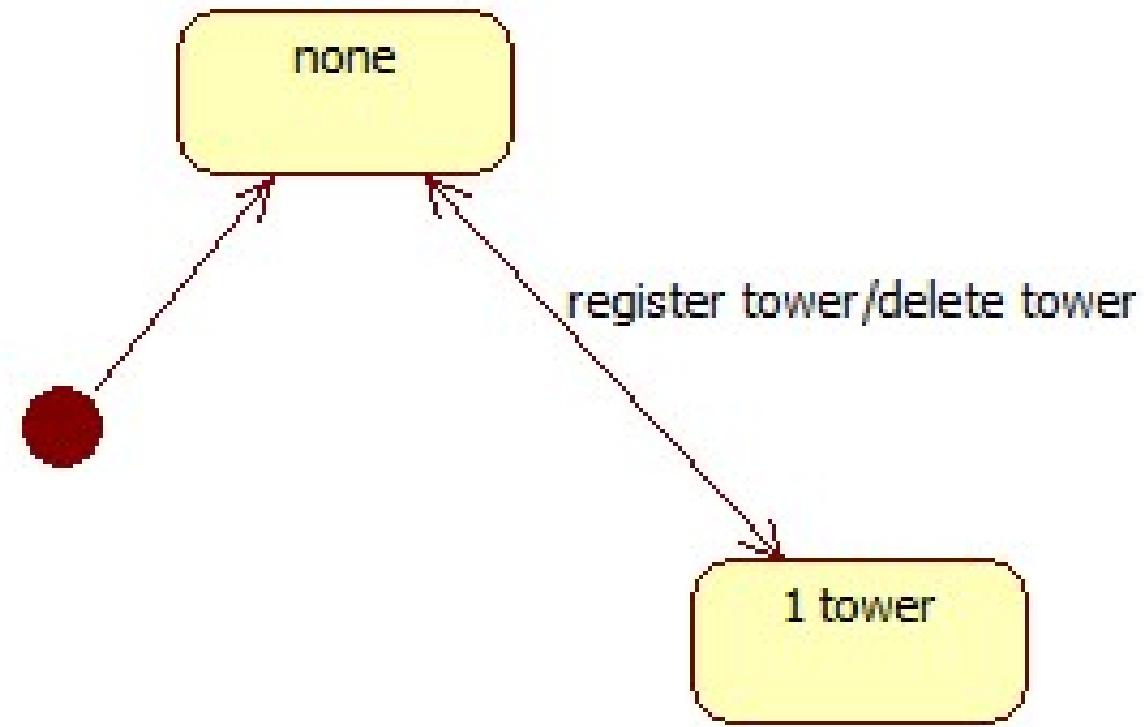
3.4.6. Torony tüzelése



3.7. ábra. Torony tüzelése szekvenciadiagram

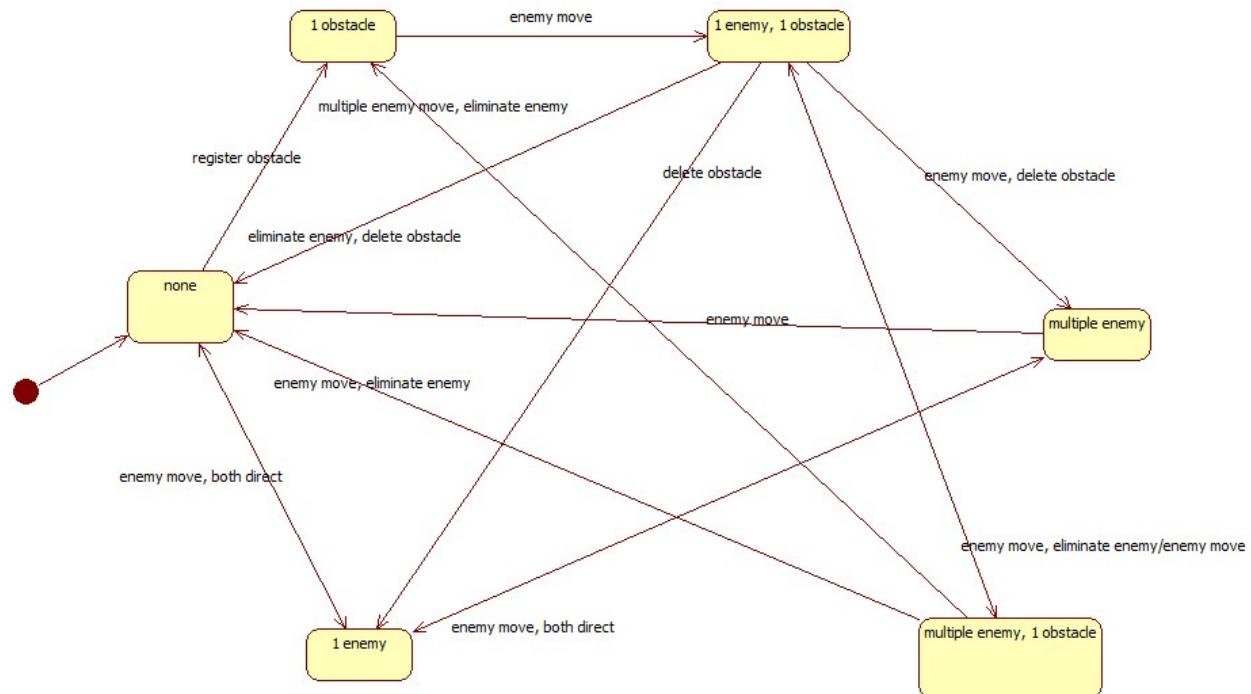
3.5. State-chartok

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	Résznevők	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

Kezdet	Időtartam	Résznevők	Leírás
2014.03.01. 9:20	2,5 óra	Rédey	Obstacle, IObstacle, Game osztályok elkészítése
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 casek 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

4. Analízis modell kidolgozása 2

4.1. Objektum katalógus

4.1.1. Akadály (Obstacle)

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

4.1.2. Ellenség (Enemy)

Egy ellenséget (tündér, hobbit, törp vagy ember) megvalósító objektum. Az ő felelőssége, hogy egy adott lövedék (Bullet) hatására sebződjön, vagy ha már sokat sebződött, akkor haljon meg, valamint az, hogy celláról cellára mozgassa magát, és ha eléri a végzet hegyét értesítse a játék (Game) osztályt, hogy intézkedjen.

4.1.3. Játék (Game)

A játék (Game) objektum felelőssége többek közt a játék ütemezése, az idő műlásának 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. A game osztály hozza létre az ellenségeket és indítja el őket az úton.

4.1.4. 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.

4.1.5. Lövedék (Bullet)

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

4.1.6. Mező (Field)

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

4.1.7. Pálya (Map)

A Map osztály a játéktér elemeit, mint cellák tárolja, egy két dimenziós tömbben. Megadja minden egyes cellához, a szomszédjai referenciáját.

4.1.8. 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 tornyokat, illetve el is adhatja őket. Ez a legfontosabb eszköz amivel a játékos meg tudja akadályozni az ellenségek célbajutását.

4.1.9. Út (Path)

A Path a Cell osztály leszármazottja. Az út típusú cellákat reprezentálja. Tartalmazza a rajta lévő ellenségeket és esetleg akadályt. minden út tudja azt is, hogy hova lehet lépni róla egy lépésben.

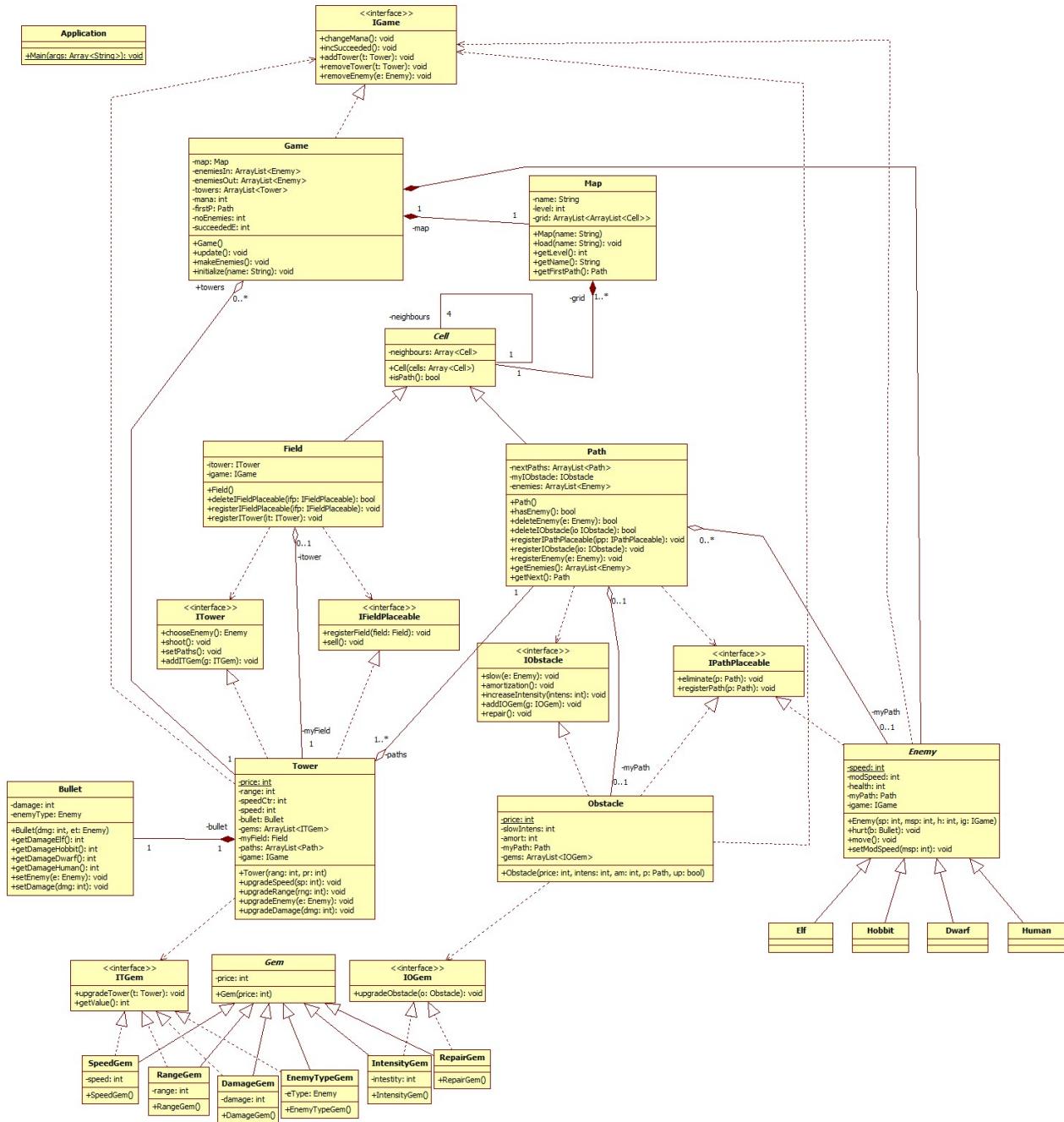
4.1.10. Obstaclehez tartozó kristályok: Intensity, Repair

Akadályra helyezhető kristályok, ami növeli a lassítás értékét vagy megjavítja az akadályt.

4.1.11. Towerhez tartozó krsitályok: Range, Speed, Damage, EnemyType

A torony lövésének hatósugarát, gyorsítását, sebzését, ellenféltípusra sebzés, növelő kristályok osztályai.

4.2. Statikus struktúra diagramok



4.1. ábra. Osztálydiagram

4.3. Osztályok leírása

4.3.1. Bullet

- Felelősség
Ellenség kapja meg, és ebből tudja meg mennyire sebződik.
- Ősosztályok

Object

- Interfészek
Nincs
- Attribútumok

int damage alapsebzés

String enemyType a torony itt tárolja, hogy melyik ellenség típusra erősebb a sebzése

- Metódusok

Bullet(int dmg, String et) Konstruktor

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

void setEnemy(Enemy e) beállítja az ellenséget akire specializált

void setDamage(int damage) beállítja a lövedék sebzését

4.3.2. 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

static final int speed A két lépés között eltelt idő.

static final int maxHP A maximális életerő.

int modSpeed Az ellenség belső idő mérője. A setModSpeed változtathatja – jellemzően negatív irányba, akadályokon.

Path nextPath A következő path címe, ahova lép.

protected int health Életerejét tárolja ebben. Hurt függvényben csökkenti.

protected Path myPath az a mező, ahol tartózkodik

protected IGame igame ezen keresztül tudja módosítani a manát, amikor meghal, illetve ha elér a végzet hegyére módosítani a számlálót (Game.succeededE), hogy nőjön egyel

- Metódusok

hurt(Bullet b) sebződik (abstract method)

move() mozog, a következő path-ra lép, cellát vált

Enemy(ig: IGame) konstruktor

void setModSpeed(int msp) modSpeed változót változtatja. Lassítani lehet vele.

void setHealth(int h) health setter.

4.3.3. 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 b) sebződik, a kapott Bullet alapján bizonyos mértékű összeget levon az életpontjuktól.

4.3.4. Game

- Felelősség

Lásd objektum katalógus.

- Ősosztályok

Object

- Interfészek

IGame

- Attribútumok

Map map játék térképe

List<Enemy> enemiesOut pályára még be nem lépett ellenségek

List<Enemy> enemiesIn pályára már belépett ellenségek

List<Tower> towers a tornyok, amik a pályán vannak

int mana maradék varázserő

Path firstP az út kezdő cellája

int noEnemies kezdeti hullámérték, amely folyamatosan nő, azt mutatja meg, hogy következő körben hány ellenséget kell létrehozni és beküldeni a pályára

int succeededE végzet hegyét elérte enemy-k száma

- Metódusok

void update() frissíti a modellt, grafikát

void initialize(String name) kiinduló állapot felvétele

Game() konstruktor

void makeEnemies() létrehoz néhány ellenséget, ezeket beteszi az enemiesOut-ba

4.3.5. Controller

- Felelősség

A felhasználó és a játék közötti kommunikációnak véghezvitele a felelőssége. Kristályok, tornyok, akadályok vásárlását lehet rajta keresztül megcsinálni. Ellenőriznie kell, hogy van-e a játékosnak elég várázsereje a tranzakcióhoz.

- Ősosztályok

Object

- Interfészek

Nincs

- Attribútumok

IGame Szükséges függvények elérésére szolgál, hogy tudja módosítani a Game-et.

Field chosenField Ha Field-et választ ki, ebben a változóban tárolja.

Path chosenPath Ha Path-t választ ki, ebben a változóban tárolja.

String chosenEnemy A kiválasztott EnemyType.

- Metódusok

void buyTower() Torony vételére szolgál. Létrehoz egy tornyot, beregisztrálja a chosenField-re.

void buyObstacle() Lásd buyTower Obstacle-el és chosenPath-el.

void buySpeed/Range/Damage/EnemyType/Intensiyty/RepairGem() Kristályok vásárlása. A Towert/Obstacle-t amire vettük a chosenField/Path-en érjük el.

void setField(Field f) A chosenField-et állítja.

void setPath(Path p) A chosenPath-t állítja.

void setEnemy(String e) A chosenEnemy-t állítja.

Field/Path/Enemy getChosenField/Path/Enemy() Getter a chosen attribútumokhoz.

4.3.6. IGame

- Felelősség

Az IGame interfész szolgáltatást nyújt az akadályoknak, tornyoknak, ellenségeknek, hogy rajta keresztül manát írjanak jóvá/csökkentsenek, illetve ellenségek esetén a végzet hegyét elért ellenségek számát módosítsák. Speciális interfész a Game osztályhoz.

- Metódusok

void changeMana() manát megváltoztató metódus

void incSucceeded() succeededE értékét megváltoztató metódus

void addTower(Tower t) hozzáad egy tornyot a listához

void removeEnemyIn(Enemy e) ellenség törlése enemiesIn-ból

void removeEnemyOut(Enemy e) ellenség törlése enemiesOut-ból

void removeTower(Tower t) torony törlése towers-ból

void addEnemyIn(Enemy e) ellenség törlése enemiesIn-ból

int getMana() mana lekérdezése

4.3.7. Gem

- Felelősség
A kristály osztály felel a torony fejlesztéséért.
- Ősosztályok
Object
- Interfészek
Nincs
- Attribútumok

public static final int price az ár, amennyi varázserőbe kerül.

- Metódusok

Gem() konstruktor

4.3.8. 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

void increaseIntesity(int intens) megnöveli az intesity-t intens-el

void addIOGem(IGem: iog) iog kristályt hozzáadja az akadályhoz

repair() megjavítja az akadályt

4.3.9. 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

public static final int price az ára

ArrayList<IOGem> gems a megvett kristályok listája

- Metódusok

Obstacle() konstruktur

4.3.10. ITower

- Felelősség

A torony funkciói vannak benne.

- Metódusok

void setPaths() a saját cellájából kiindulva a hatósugarával lefedett területen felkeresi, és beregiszt- rálja 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 addITGem(ITGem gem) paraméterként megkapja a kiválasztott kristályt, bulletet, torony attribútumait frissíti.

Enemy chooseEnemy() A torony tárolja a hatókörbe eső path cellákat. minden tick-ben véig megy rajtuk, és kiválaszt egyet, amelyiken van ellenség, és oda fog lőni. Az ellenséggel tér vissza.

void sell() torony eladása és törlése myFieldről

String getEnemyType() az ellenségre fejlesztettség lekérése.

4.3.11. Tower

- Felelősség

Lásd objektumkatalógus

- Ősosztályok

Nincs

- Interfészek

ITower, IFieldPlaceable

- Attribútumok

public static final int price az ára varázserőben.

int range lőtáv, hatókör.

int speedCtr A torony belső idő mérője. Ezt vizsgálja minden lövés előtt, hogy eltelt-e elég idő.

int speed két lövés között eltelt minimális idő.

Bullet bullet A torony tárol egy lövedéket, mindig ezt lövi ki.

ArrayList<ITGem> gems A megvásárolt kristályokat tárolja.

ArrayList<Path> paths Hatósugárba eső út cellák.

Field myField mező, amin áll.

IGame igame Egy interfész a játék logikára, amivel a bejutott ellenségek számát, és a varázserőt is lehet állítani.

- Metódusok

Tower(IGame igame) konstruktur

void upgradeSpeed(int sp) fejleszti a lövési sebességét.

void upgradeRange(int rng) fejleszti a lőtávot.

void upgradeEnemy(Enemy e) egy ellenségtípusra növeli a sebzést.

void upgradeDamage(int dmg) növeli a sebzést.

4.3.12. 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(IGame igame) az osztály konstruktora, a paraméterként megadott névvel rendelkező fájlból betölti a pálya térképét

void load(String name) megnyitja a paraméterként kapott nevű fájlt, és abból betölti a pálya celláinak tulajdonságait, felépíti a pályát.

int getLevel() visszaadja a pálya szintjét.

String getName() visszaadja a pálya nevét

Path getFirstPath() visszaadja a pálya belépési pontjának referenciáját

Cell getCell(int i, int j) i-j koordinátájú cella legérése

Array<Array<Cell>> grid getGrid() játéktér lekérése.

bool isLoaded() megadja, hogy be van-e töltve a pálya

int getHeight/Width() magasság, szélesség lekérése

Dimension getSize() pálya mértetit adja meg dimenzióban

4.3.13. 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> cells, IGame igame) konstruktur

bool isPath() olyan értékkel tér vissza amilyen típusú a cella

Array<Cell> getNeighbours() neighbours getter

Array<Cell> setNeighbours(Array<Cell> neig) neighbours setter

4.3.14. Field

- Felelősség
Ld. objektum katalógus
- Ősosztályok
Object → Cell
- Interfészek
Nincs
- Attribútumok

ITower itower a mezőn álló torony interfészű elem tárolása

IGame igame : referencia Game interfészéhez

- Metódusok

bool isPath() hamis értékkel tér vissza

void registerIFieldPlaceable(IFieldPlaceable ifp) egy új tornyot ad hozzá a mezőhöz

void deleteIFieldPlaceable(IFieldPlaceable ifp) eltávolítja a tornyot a mezőről

void registerITower(itower ITower) beteszi ifieldbe a kapott tornyot

Field(IGame igame) konstruktur

ITower getITower() itower getter

bool hasTower() megadja, hogy van-e torony a mezőn

4.3.15. Path

- Felelősség
Ld objektum katalógus
- Ősosztályok
Object → Cell

- Interfészok
Nincs
- Attribútumok

IObstacle iobstacle az esetleg az úton levő akadályt tárolja

ArrayList<Enemy> enemies az éppen áthaladó ellenségek listája

ArrayList<Path> nextPaths következő path-ok címei

- Metódusok

bool isPath() igaz értékkel tér vissza

void deleteObstacle(IObstacle io) kitörli a tárolójából a paraméterként kapott referenciával meggyező 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 betesz az enemies-be

void registerObstacle(Obstacle o) a o kapott akadály lesz az obstacle

ArrayList<Enemy> getEnemies() visszatér az enemies-el

Path getNext() paths-ból ad vissza egy elemet

IObstacle getIObstacle() myIObstacle getter

bool hasObstacle() megadja, hogy van-e akadály az úton

void setNextPaths(ArrayList<Path> np) az következő utak címei

Path() konstruktor

4.3.16. Towerhez tartozó krsítályok: Range, Speed, Damage, EnemyType

- Felelősség
Ld objektum katalógus
- Ősosztályok
Object → Gem
- Interfészek
ITGem
- Attribútumok

int/ int/ int/ String range/ speed/ damage/ eType

- Metódusok

Konstruktörök

4.3.17. Obstaclehez tartozó kristályok: Intensity, Repair

- Felelősség
Ld objektum katalógus
- Ősosztályok
Object → Gem
- Interfészek
IOGem
- Attribútumok

int intensity az IntensityGem-hez tartozik, a Repair-nek nincs attribútuma

- Metódusok

Konstruktörök

4.3.18. IOGem

- Felelősség
Akadályra helyezhető kristályok interfésze.
- Metódusok

void upgradeObstacle(Obstacle o) a kapott akadályt fejleszti.

4.3.19. ITGem

- Felelősség
Toronyra illeszthető kristályok interfésze.
- Metódusok

void upgradeTower(Tower t) fejleszti a kapott t tornyot magával

int getValue() visszaad egy, a torony árával képzett értéket, a torony eladásakor jóváírandó mana érték kiszámításához

4.3.20. IFieldPlaceable

- Felelősség
Interfész a mezőre helyezhető osztályok számára. Azonosítja azokat az objektumokat, amelyeket csak a mező típusú pályaelem tartalmazhat.
- Metódusok

void registerField(Field field) a mezőre helyezhető objektumnak megadja paraméterben annak a mezőnek a referenciáját, amelyikre helyezve lesz.

4.3.21. IPathPlaceable

- Felelősség

Interfész az útra helyezhető osztályok számára. Azonosítja azokat az objektumokat, amelyeket csak az út típusú pályaelem tartalmazhat.

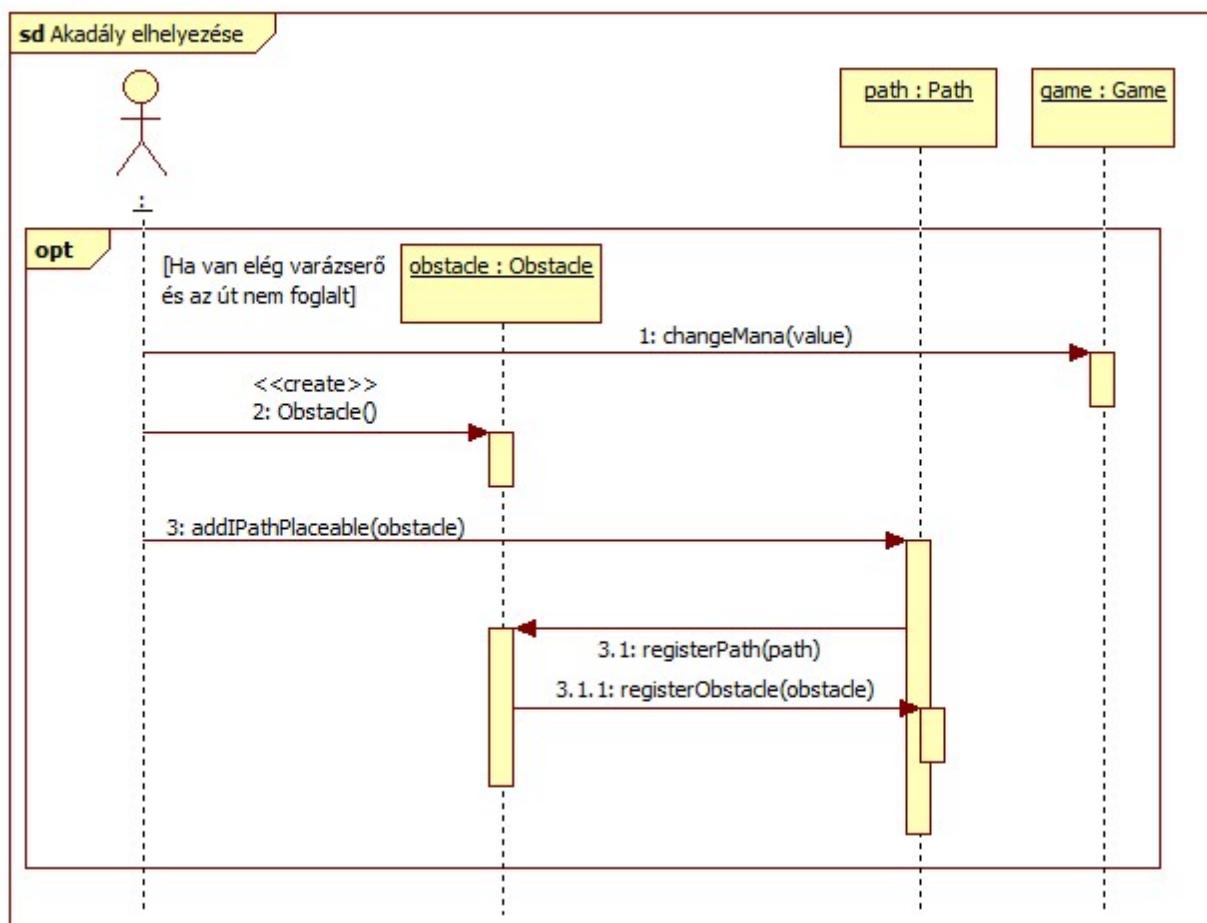
- Metódusok

void eliminate(Path p) az útra helyezhető objektum eltávolítása az útról.

void registerPath(Path p) az útra helyezhető objektumnak megadja paraméterben annak az útnak a referenciáját, amelyikre helyezve lesz.

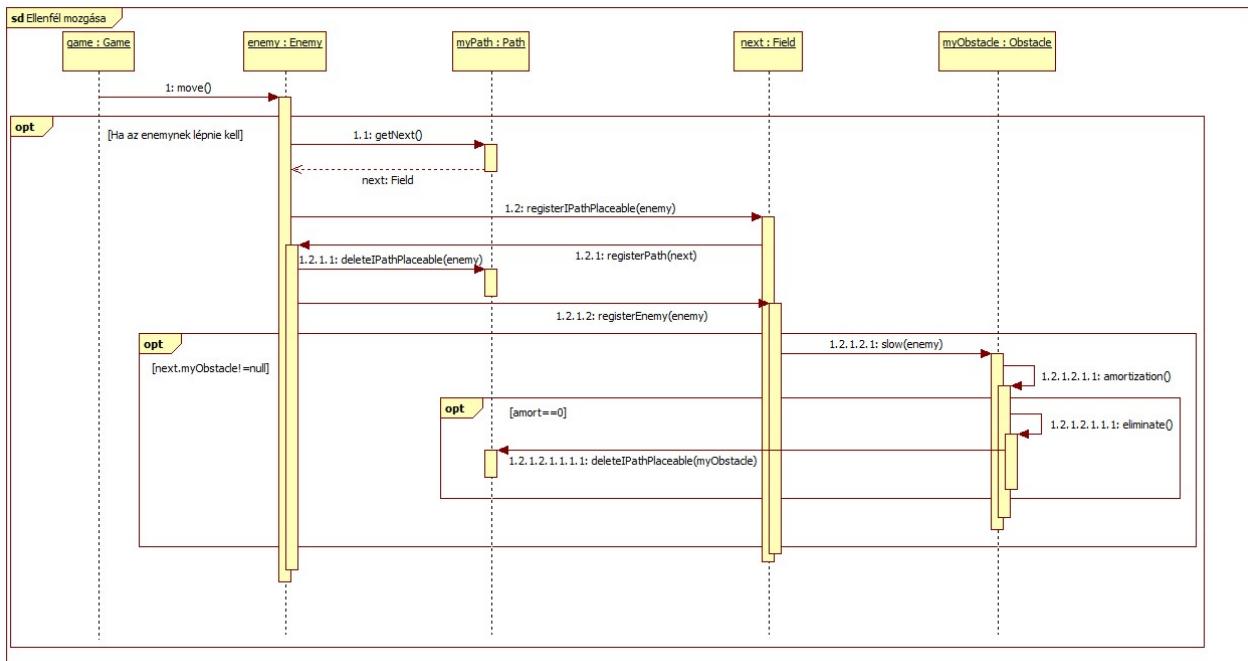
4.4. Szekvencia diagramok

4.4.1. Akadály elhelyezése



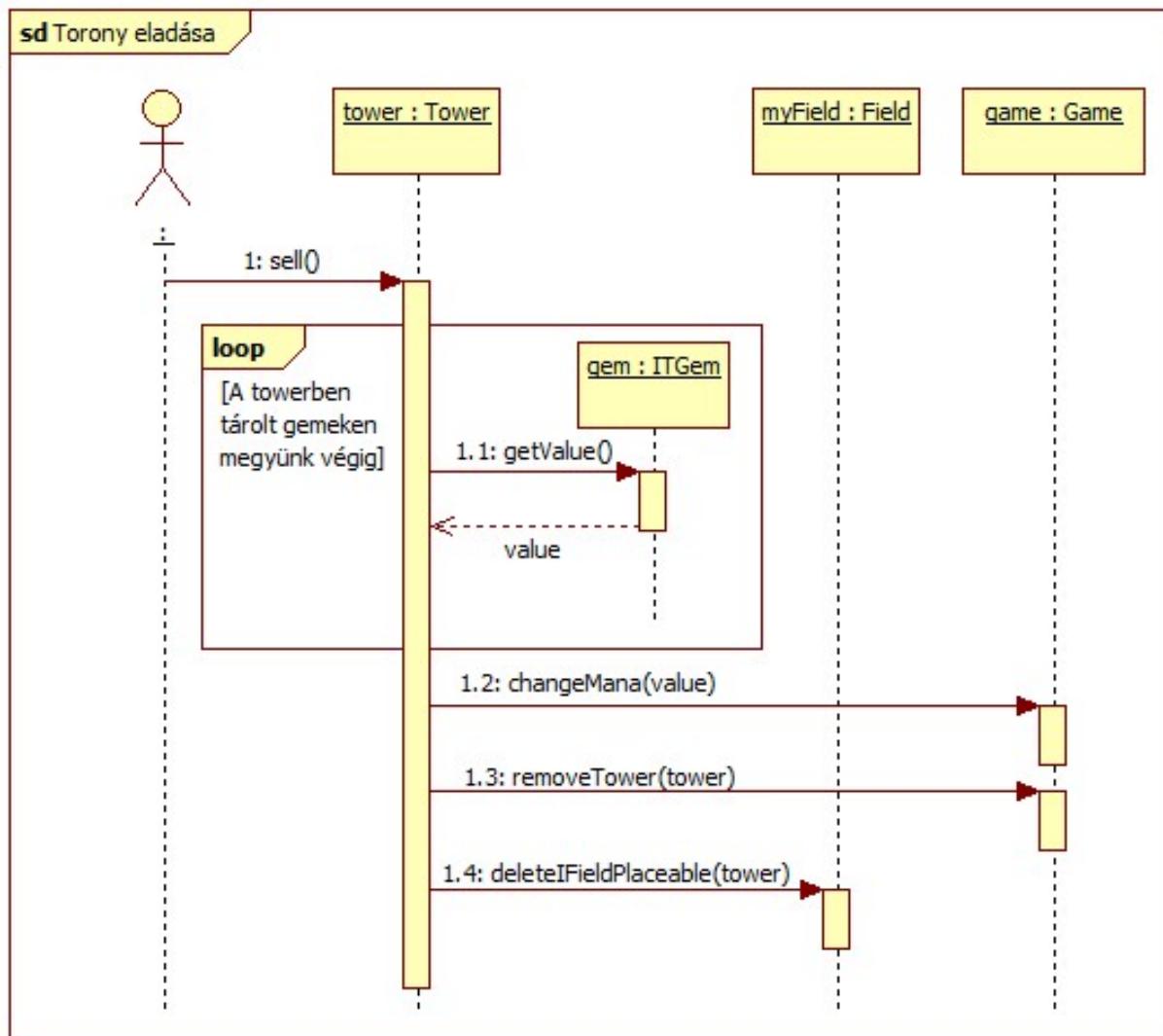
4.2. ábra. Akadály elhelyezése szekvenciadiagram

4.4.2. Ellenfél mozgása



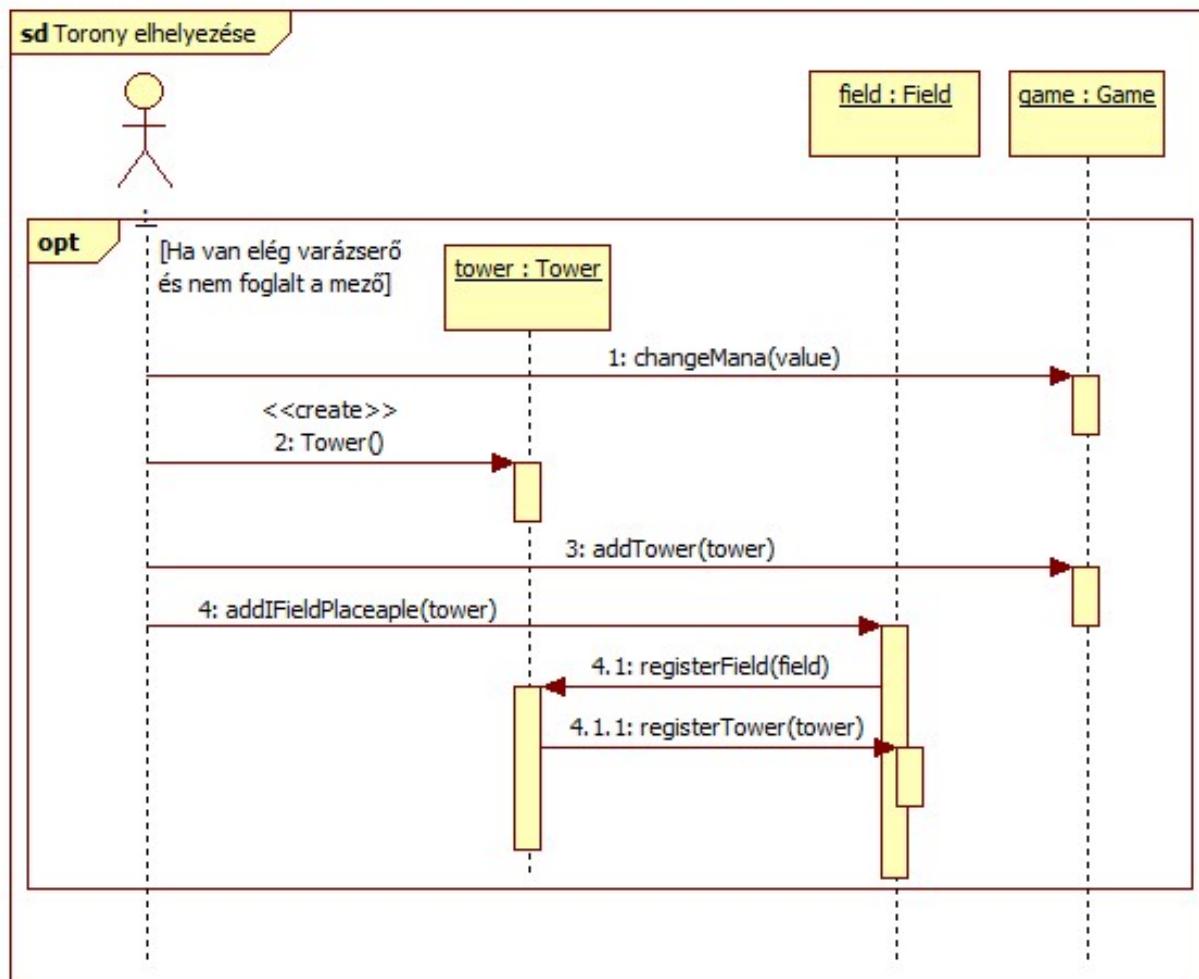
4.3. ábra. Ellenfél mozgása szekvenciadiagram

4.4.3. Torony eladása



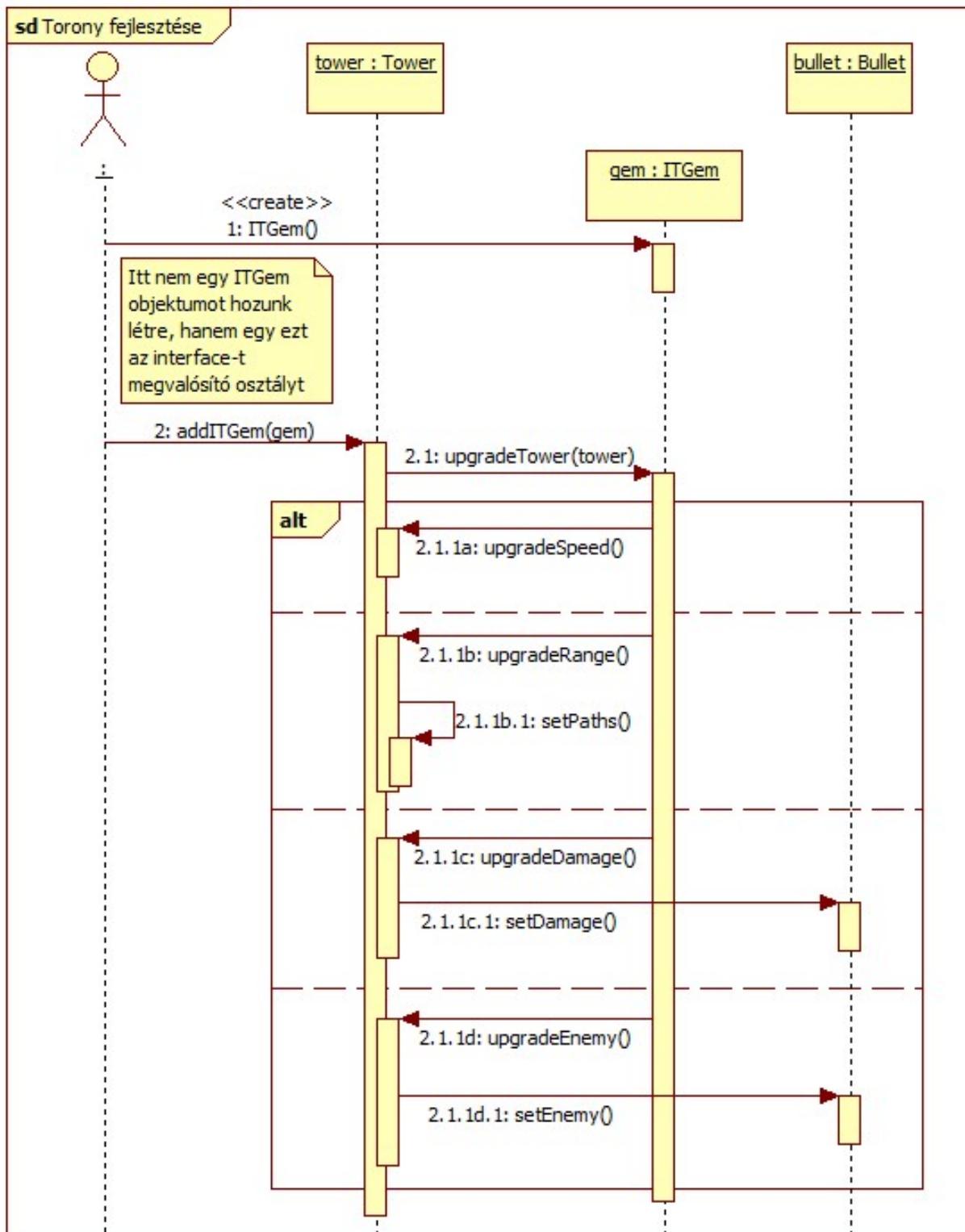
4.4. ábra. Torony eladása szekvenciadiagram

4.4.4. Torony elhelyezése



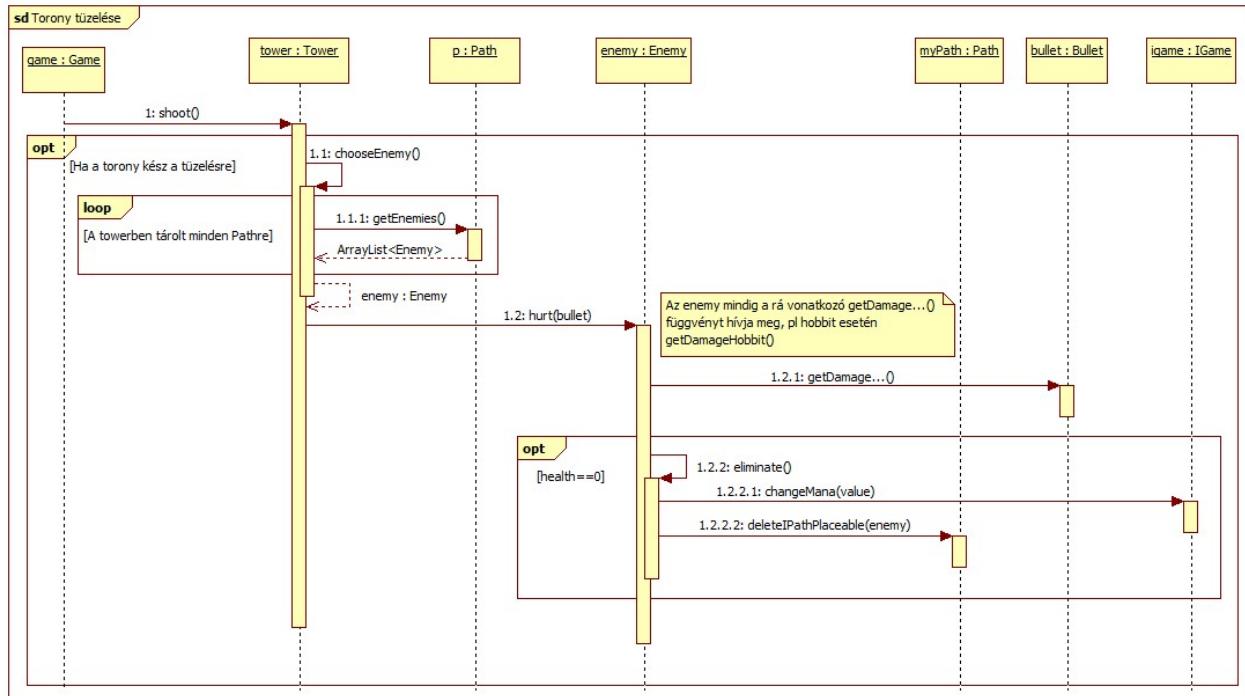
4.5. ábra. Torony elhelyezése szekvenciadiagram

4.4.5. Torony fejlesztése



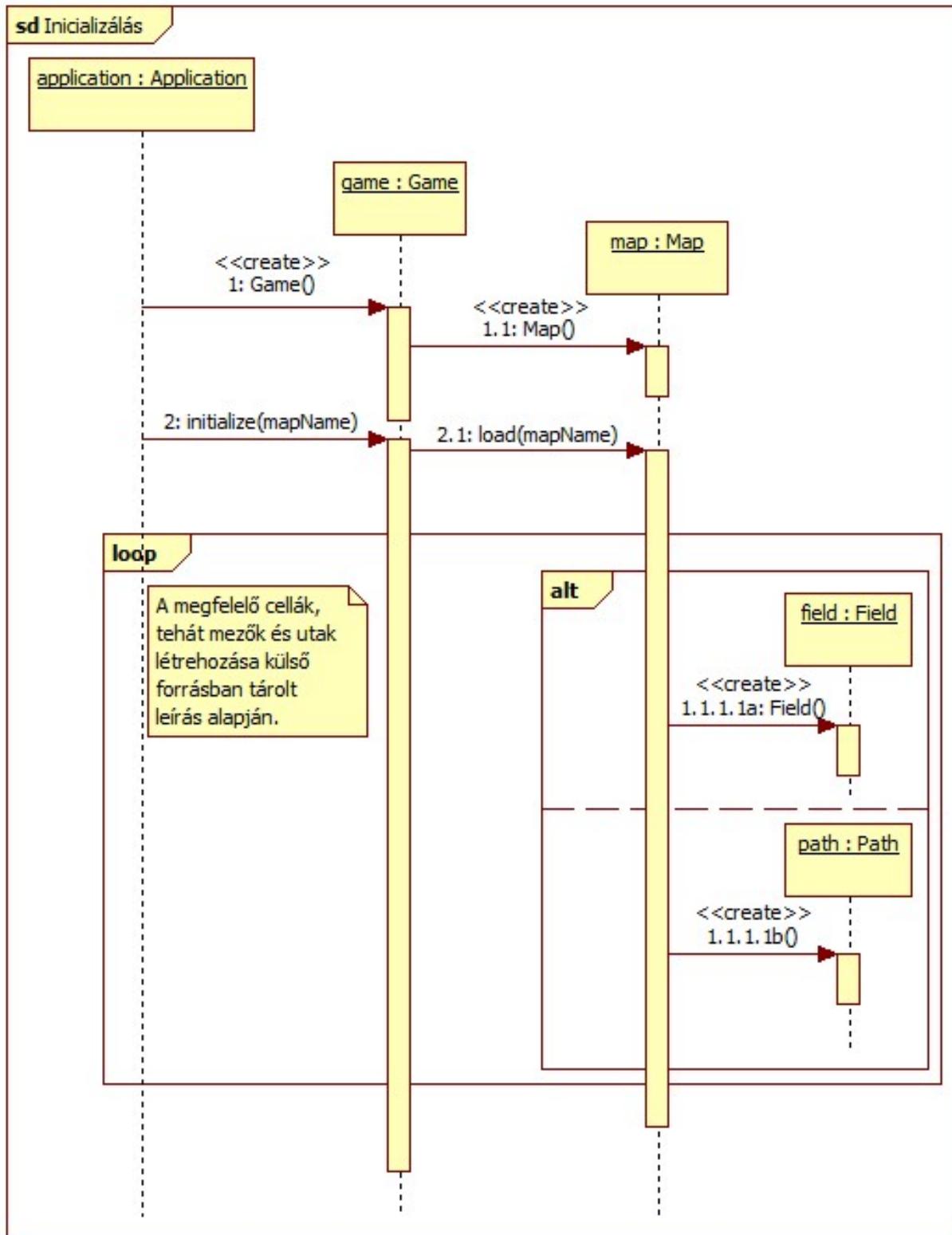
4.6. ábra. Torony fejlesztése szekvenciadiagram

4.4.6. Torony tüzelése



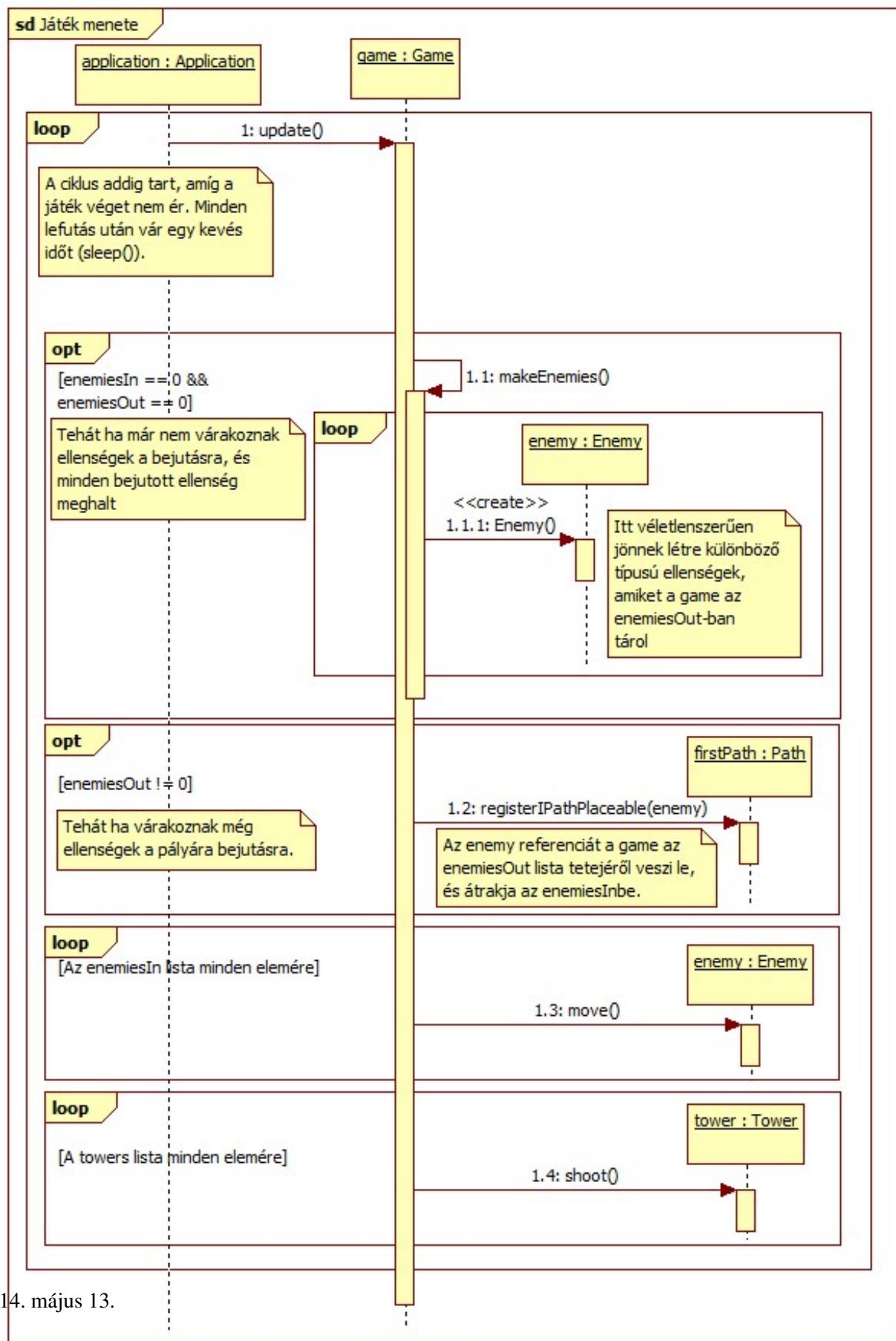
4.7. ábra. Torony tüzelése szekvenciadiagram

4.4.7. Inicializálás



4.8. ábra. Inicializálás szekvenciadiagram

4.4.8. Játék menete



4.5. State-chartok

4.6. Napló

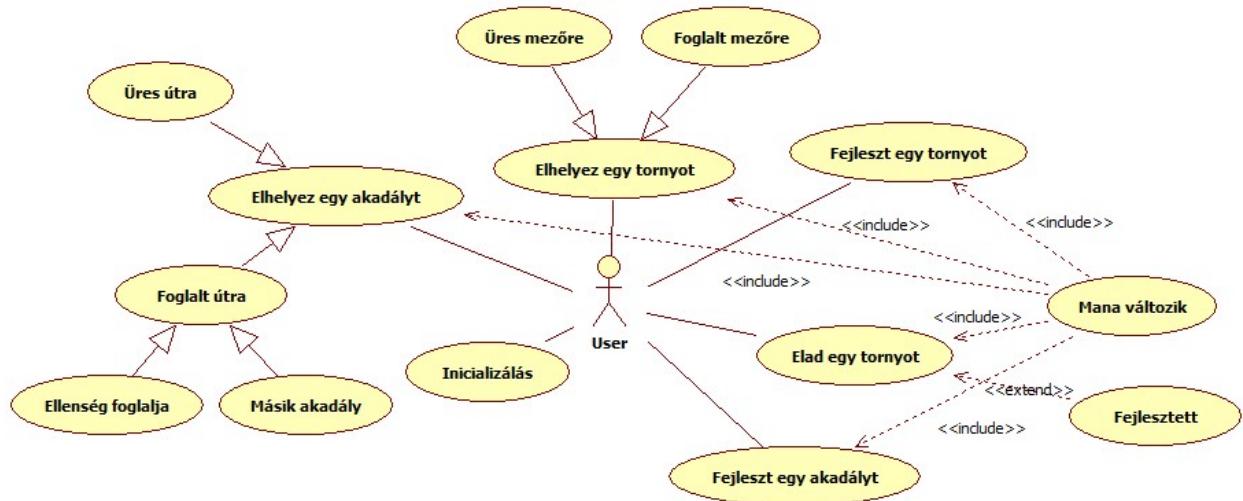
Kezdet	Időtartam	Résznevők	Leírás
2014.03.05. 08:00	1,5 óra	Elekes Seres Rédey Nagy Fuksz	Konzultáció
2014.03.05. 15:00	30 perc	Rédey	Gem, IOGem, ITGem és a leszármazottaik/implementálóik bevezetése az osztálydiagrammiba, IGame interfész felvétele
2014.03.06. 11:45	45 perc	Elekes	IOGem, ITGem, Speed, Range, Damage, EnemyType, Intensity, Repair osztályleírások
2014.03.07. 14:00	3 óra	Elekes Seres Rédey	A megírt gem-ek átbeszélése, inicializálás, ellenségek beküldése, ellenség elér a végzethegyére, egyéb szekvenciák kidolgozása
2014.03.08. 14:00	1 óra	Elekes	Bullet, Tower, Gem-ek. Változtatások átviteli az osztályleírásokba.
2014.03.08. 16:00	1 óra	Rédey	Game, IGame, Obstacle, IOObstacle, Enemy és leszármazottai osztályokleírások, osztálydiagramm frissítése, észlelt hibák javítása
2014.03.09. 21:00	3 óra	Seres	Szekvencia diagramok elkészítése, módosítása
2014.03.10. 01:00	3 óra	Fuksz	Dokumentáció véglegesítése

5. Szkeleton tervezése

5.1. A szkeleton modell valóságos use-case-ei

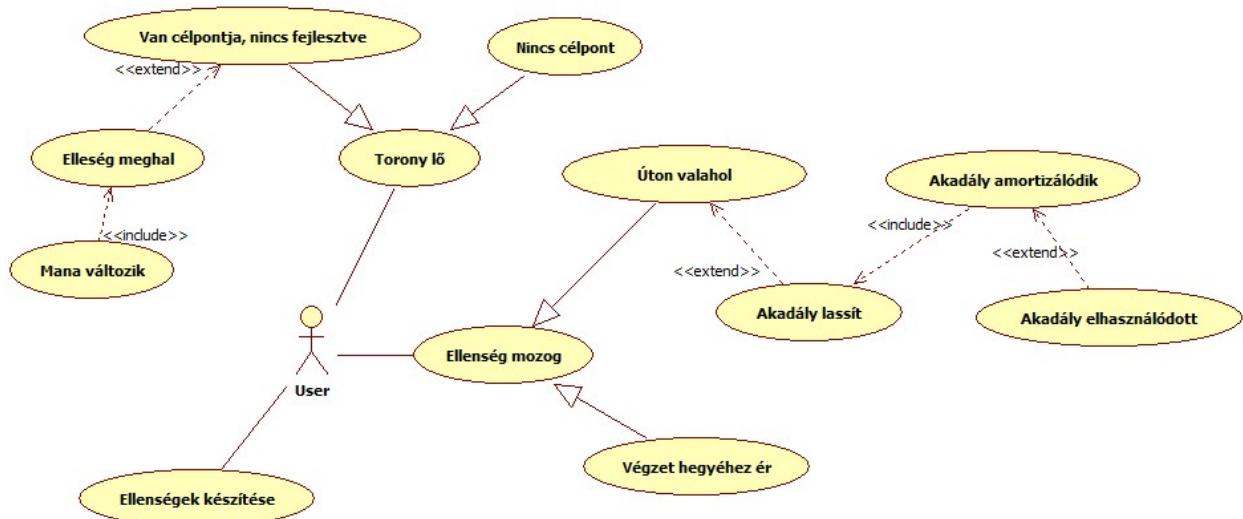
5.1.1. Use-case diagram

1. use-case diagram



5.1. ábra. Játékos use-case diagram 1.

2. use-case diagram



5.2. ábra. Játékos use-case diagram 2.

5.1.2. Use-case leírások

Use-case neve	1. Akadály elhelyezése ellenség által foglalt útra
Rövid leírás	A játékos egy akadályt akar helyezni egy útra, amin már van egy ellenség
Aktorok	User
Forgatókönyv	Lekérdezés, hogy az úton van-e ellenség vagy akadály, van ellen-ség, nem történik semmi.

Use-case neve	2. Akadály elhelyezése másik akadály által foglalt útra
Rövid leírás	A játékos egy akadályt akar helyezni egy útra, amin már van egy akadály
Aktorok	User
Forgatókönyv	Lekérdezés, hogy az úton van-e ellenség vagy akadály, van aka-dály, nem történik semmi.

Use-case neve	3. Akadály elhelyezése üres útra
Rövid leírás	A játékos egy akadályt akar helyezni egy útra, amin nincs sem ellenség sem akadály
Aktorok	User
Forgatókönyv	Lekérdezés, hogy az úton van-e ellenség vagy akadály, nincs egyik sem, varázserő levonása, akadály létrehozása, akadály el-helyezése az úton.

Use-case neve	4. Akadály fejlesztése
Rövid leírás	Akadály fejlesztése minden gem-el
Aktorok	User
Forgatókönyv	Minden, az akadályhoz tartozó gem típusból létrehozunk egyet, hozzáadjuk az akadályhoz és érvényre juttatjuk a fejlesztést

Use-case neve	5. Ellenség mozog
Rövid leírás	Egy ellenség egy celláról egy másikra jut
Aktorok	User
Forgatókönyv	Az ellenség elérhet a végzet hegyéhez, egyébként a következő cellára lép

Use-case neve	6. Fejlesztetlen torony eladása
Rövid leírás	A játékos elad egy tornyot
Aktorok	User
Forgatókönyv	A tornyot töröljük a mezőjéről és a Game osztály adott példányából, manát növeljük

Use-case neve	7. Fejlesztett torony eladása
Rövid leírás	A Játékos elad egy fejlesztett tornyot

Aktorok	User
Forgatókönyv	A tornyot töröljük a mezőjéről és a Game osztály adott példányából. A toronyban az összes típusú gemből van egy. Az ezek által adott adatokból és a torony árából képzett értékkel módosítjuk a manát.

Use-case neve	8. Inicializálás
Rövid leírás	A játék egy kezdeti állapotának felvétele
Aktorok	User
Forgatókönyv	A cellák létrehozása, pálya betöltése, kezdeti értékek beállítása

Use-case neve	9. Torony elhelyezése foglalt mezőre
Rövid leírás	A játékos egy tornyot akar helyezni egy mezőre, amin már van másik torony
Aktorok	User
Forgatókönyv	Foglaltság ellenőrzése, mező foglalt, nem történik semmi.

Use-case neve	10. Torony elhelyezése üres mezőre
Rövid leírás	A játékos egy tornyot akar helyezni egy mezőre, amin még nincs másik torony
Aktorok	User
Forgatókönyv	Foglaltság ellenőrzése, varázserő mennyiségének ellenőrzése, minden rendben, varázserő levonása, torony létrehozása, torony regisztrálása a game-ben, torony elhelyezése a mezőn.

Use-case neve	11. Torony fejlesztése
Rövid leírás	Torony fejlesztése minden gem típusból eggyel
Aktorok	User
Forgatókönyv	A toronyhoz hozzáadjuk a gem példányokat, minden fejlesztését érvényre juttatjuk

Use-case neve	12. Torony hatókörében nincs ellenség tüzeléskor
Rövid leírás	Torony hatókörében nincs ellenség tüzeléskor
Aktorok	User
Forgatókönyv	A torony elkéri az ellenséget és nem kap ellenséget. Nem csinál semmit

Use-case neve	13. Torony lelő egy ellenséget, fejlesztetlenül
Rövid leírás	Torony lelő egy ellenséget, fejlesztetlenül
Aktorok	User
Forgatókönyv	A torony elkéri a célpontot, sebez rajta az alapértékekkel, mivel nincs fejlesztve, az ellenség meghal, ezért töröljük az útról és a Game példányból. Ugyanígy működne fejlesztett esetben.

Use-case neve	14. Torony tüzel egy ellenségre, fejlesztetlenül
Rövid leírás	Torony tüzel egy ellenségre, fejlesztetlenül
Aktorok	User
Forgatókönyv	A torony elkéri a célpontot, sebez rajta az alapértékekkel, mivel nincs fejlesztve, az ellenség nem hal meg, csak sebződik. Ugyanígy működne fejlesztett esetben.

5.2. A szkeleton kezelői felületének terve, dialógusok

5.2.1. Kezelőfelület

A szkeleton arra alkalmas, hogy egyszerűen tudjuk tesztelni, hogy a szekvencia diagramoknak megfelelően hívódnak-e meg a metódusok.

Tényleges számítások, algoritmusok nem szerepelnek a programban, ez még csak az osztályok közötti kommunikációt teszteli.

A program előre megírt teszteseteket tud majd futtatni, amik tervezünk szerint lefedik a program működésének lényegi részét. Konzolon, karakteres bevitellel lehet az egyes teszteseteket kiválasztani.

Minden objektum függvénye hívásakor kiír egy nyilat (\rightarrow), kiírja a nevét, és a függvény nevét.

Ha újabb függvényt hív, egy tabulátorral beljebb írja ki.

Visszatéréskor egy visszafelé mutató nyilat (\leftarrow) és, hogy melyik függvényből tér vissza.

5.2.2. Tesztesetek

Teszt-eset neve	1. ObstacleBuyOnEnemy
Rövid leírás	Akadály elhelyezése olyan mezőre amin ellenség van
Használt interfések	
Teszttelt use-case	Akadály elhelyezése ellenség által foglalt útra

Teszt-eset neve	2. ObstacleBuyOnObstacle
Rövid leírás	Akadály elhelyezése másik akadály által foglalt útra
Használt interfések	
Teszttelt use-case	Akadály elhelyezése másik akadály által foglalt útra

Teszt-eset neve	3. ObstacleBuy
Rövid leírás	Akadály elhelyezése üres útra
Használt interfések	
Teszttelt use-case	Akadály elhelyezése üres útra

Teszt-eset neve	4. ObstacleBuyGems
Rövid leírás	Akadály feljesztése
Használt interfések	
Teszttelt use-case	Akadály feljesztése

Teszt-eset neve	5. EnemyMove
Rövid leírás	Ellenség léptetése

Használt interfések	IPathPlaceable
Tesztelt use-case	Ellenség mozog 3 cellát, akadály nélkül, esetleg bejut a végzet hegyéhez

Teszt-eset neve	6. TowerSellNonUpgraded
Rövid leírás	Torony eladása
Használt interfések	ITower, IGame
Tesztelt use-case	Fejlesztetlen torony eladása

Teszt-eset neve	7. TowerSellUpgraded
Rövid leírás	Torony eladása
Használt interfések	ITower, IGame
Tesztelt use-case	Fejlesztett torony eladása

Teszt-eset neve	8. Initialize
Rövid leírás	A játék kezdeti betöltésének lefuttatása
Használt interfések	
Tesztelt use-case	Inicializálás

Teszt-eset neve	9. TowerBuyOnField
Rövid leírás	Egy torony elhelyezése a pályán
Használt interfések	IGame, IFIELDPlaceable
Tesztelt use-case	Torony elhelyezése foglalt mezőre

Teszt-eset neve	10. TowerBuy
Rövid leírás	Egy torony elhelyezése a pályán
Használt interfések	IGame, IFIELDPlaceable
Tesztelt use-case	Torony elhelyezése üres mezőre

Teszt-eset neve	11. TowerBuyGems
Rövid leírás	Torony feljesztése minden gemből eggyel
Használt interfések	ITower
Tesztelt use-case	Torony feljesztése minden gemből eggyel

Teszt-eset neve	12. TowerShootNoEnemy
Rövid leírás	Torony hatókörében nincs ellenség tüzeléskor
Használt interfések	ITower
Tesztelt use-case	Torony hatókörében nincs ellenség tüzeléskor

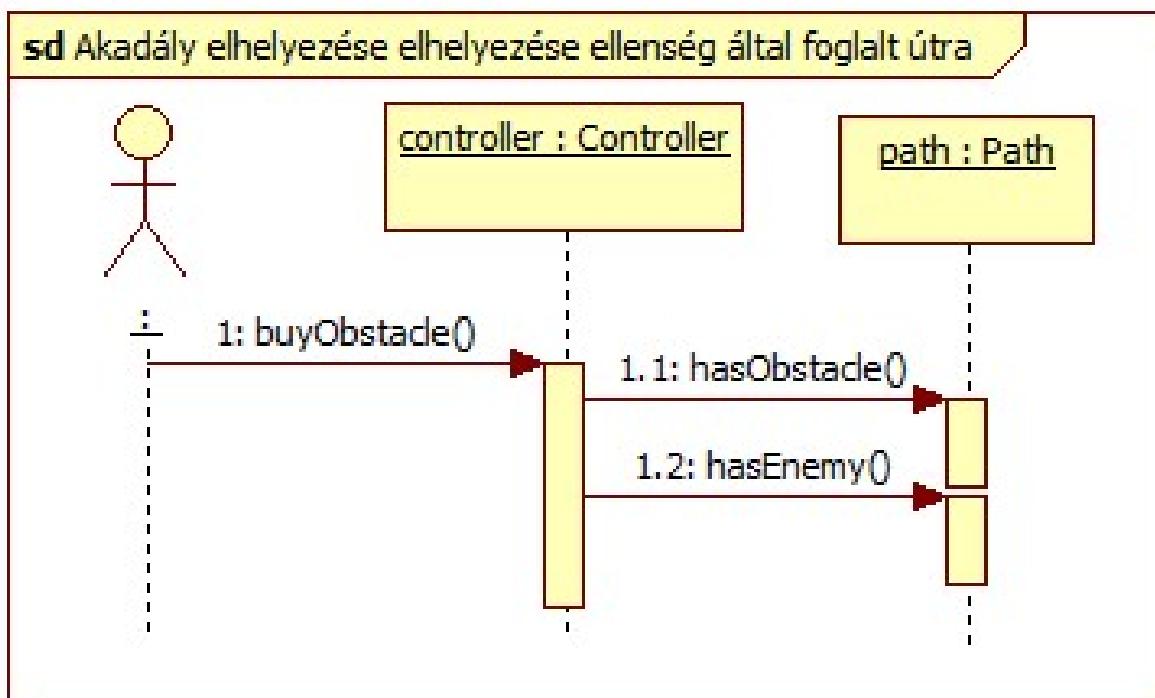
Teszt-eset neve	13. TowerKillEnemy
Rövid leírás	Torony megöl egy ellenséget

Használt interfések	ITower
Tesztelt use-case	Torony lelő egy ellenséget, fejlesztetlenül

Teszt-eset neve	14. TowerShootEnemy
Rövid leírás	Fejlesztetlen torony, hatósugarában egy ellenség van
Használt interfések	ITower
Tesztelt use-case	Torony tüzel egy ellenségre, fejlesztetlenül

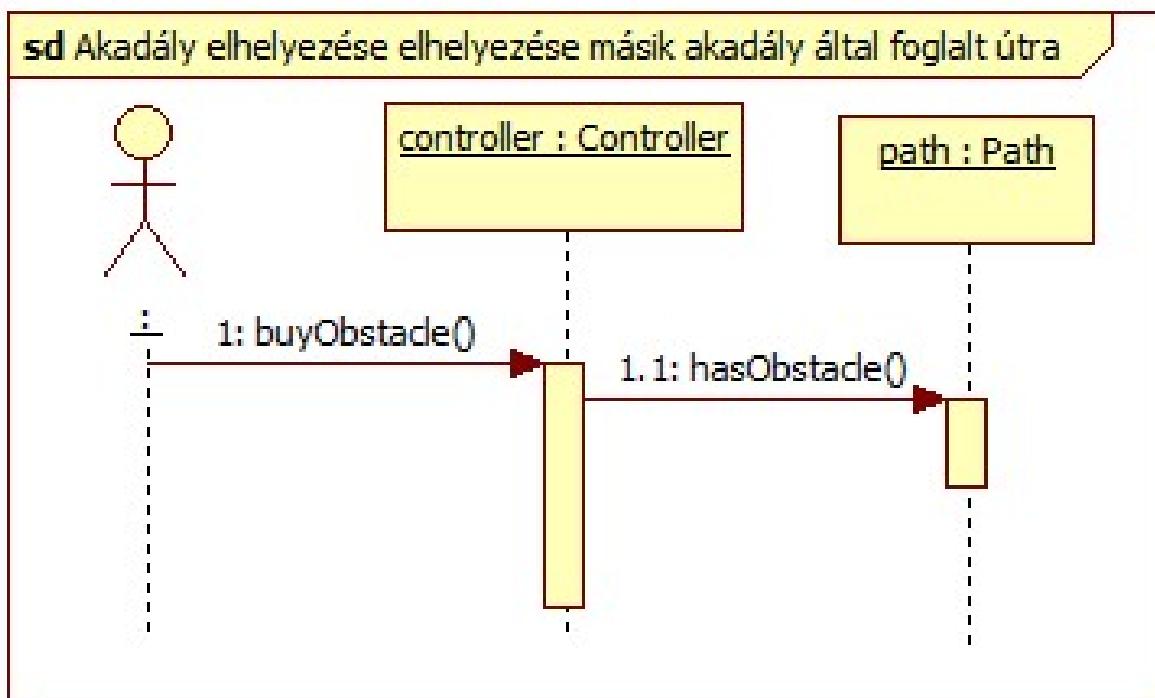
5.3. Szekvencia diagramok a belső működésre

5.3.1. 1. Akadály elhelyezése ellenség által foglalt útra



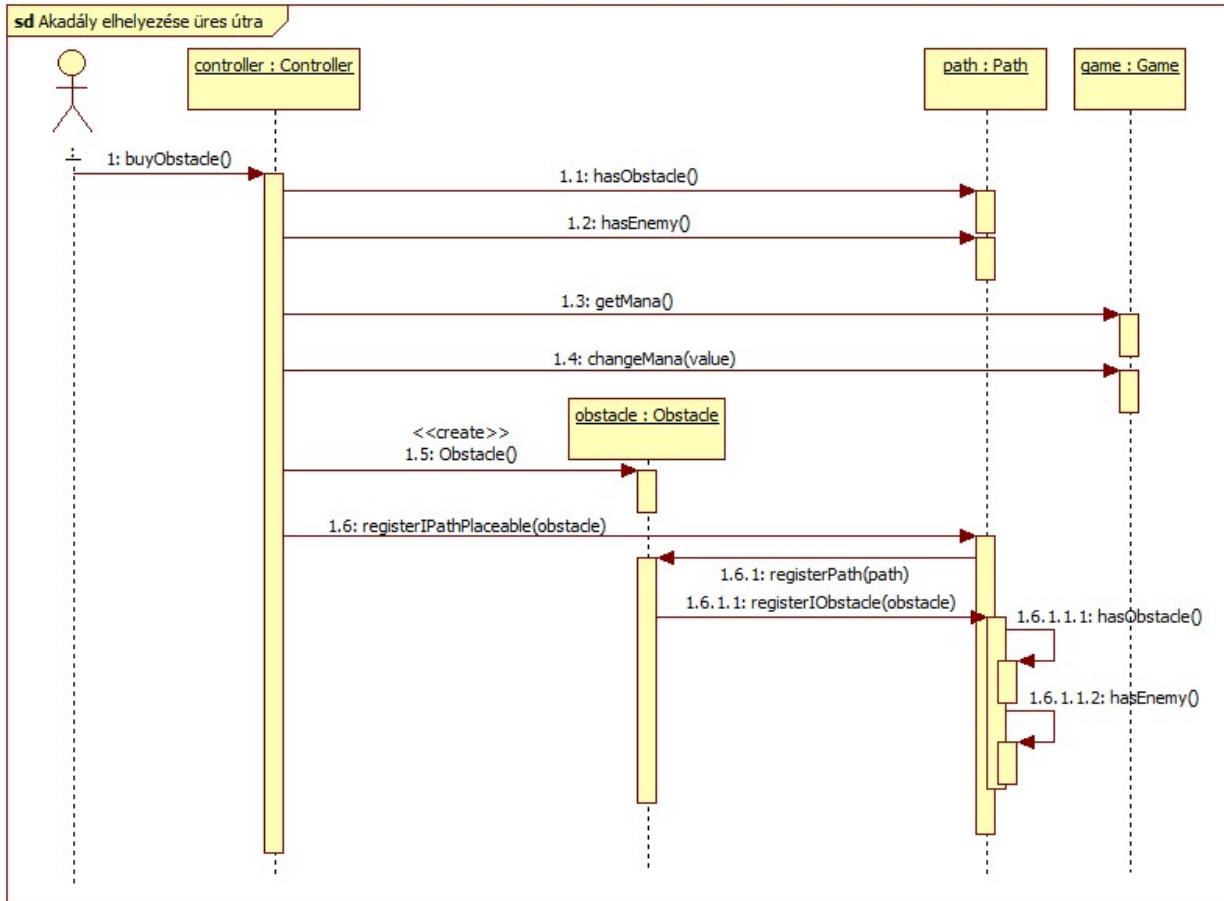
5.3. ábra. Akadály elhelyezése ellenség által foglalt útra szekvenciadiagram

5.3.2. 2. Akadály elhelyezése másik akadály által foglalt útra



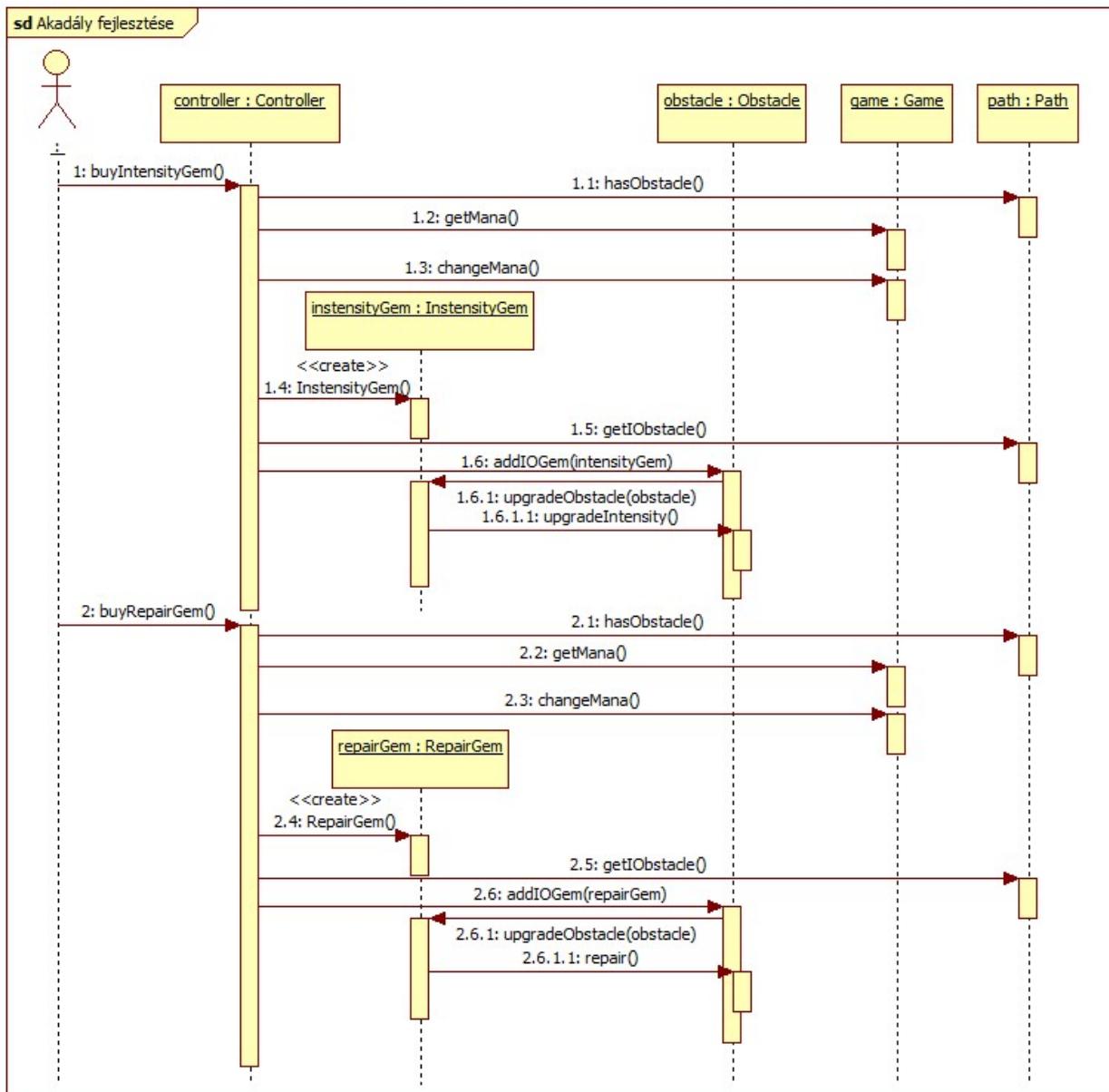
5.4. ábra. Akadály elhelyezése másik akadály által foglalt útra szekvenciadiagram

5.3.3. 3. Akadály elhelyezése üres útra



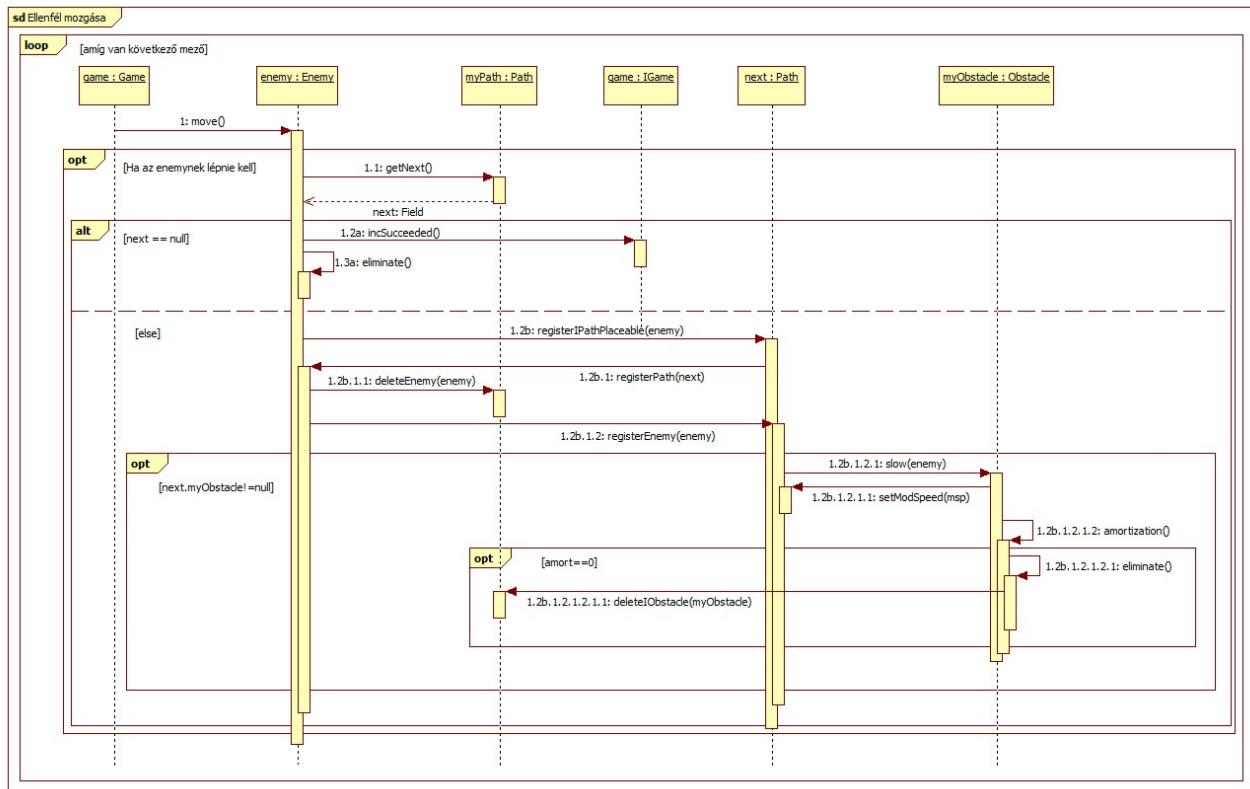
5.5. ábra. Akadály elhelyezése üres útra szekvenciadiagram

5.3.4. 4. Akadály fejlesztése



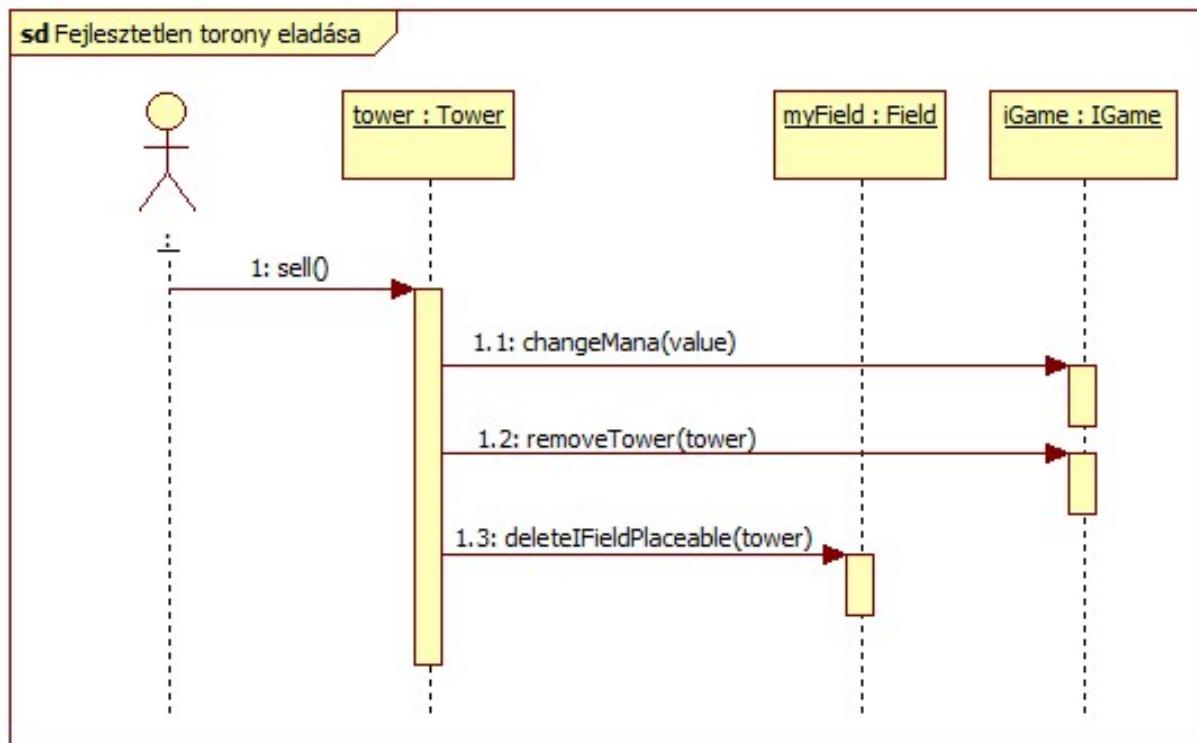
5.6. ábra. Akadály fejlesztése szekvenciadiagram

5.3.5. 5. Ellenfél mozgása



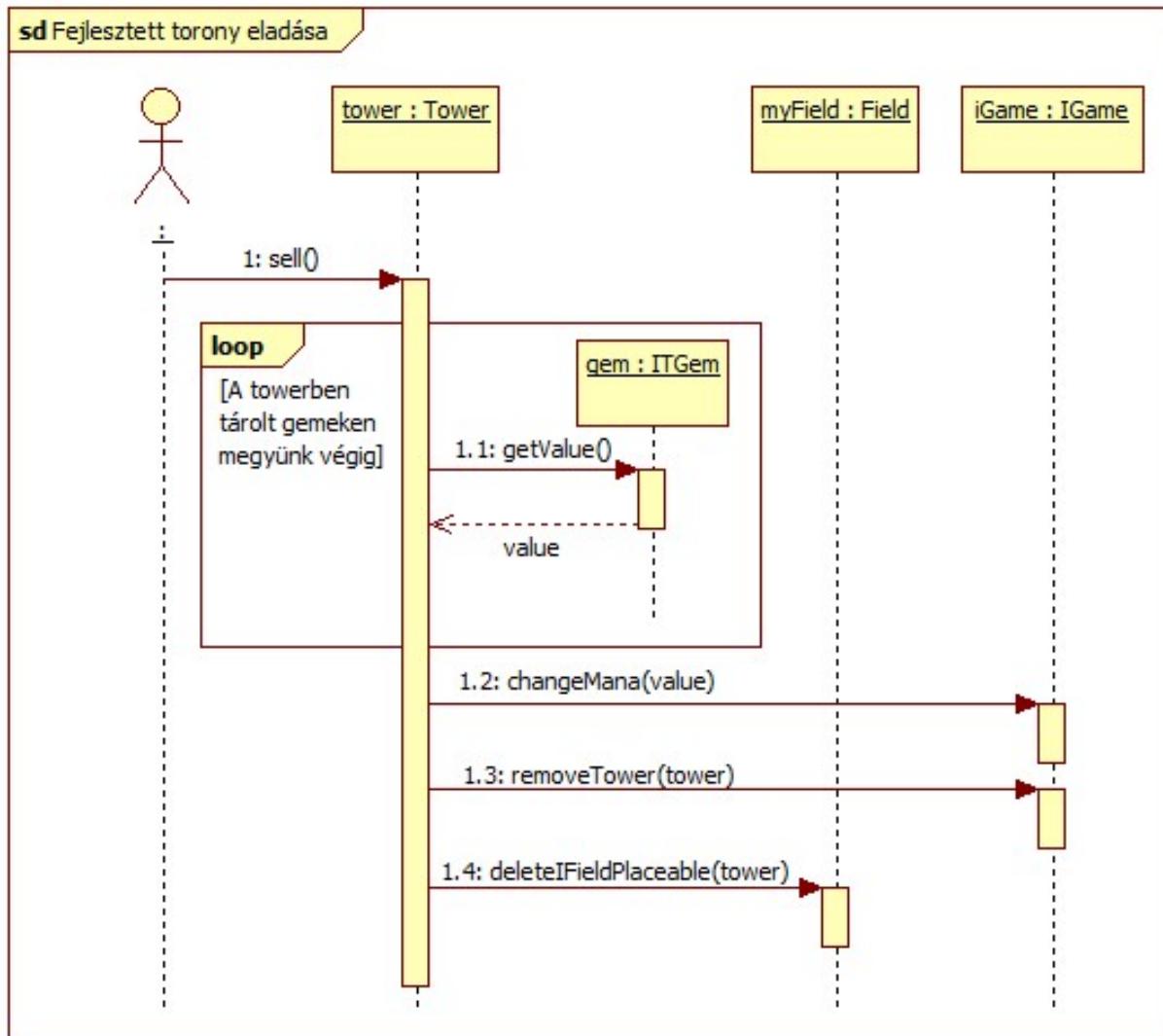
5.7. ábra. Ellenfél mozgása szekvenciadiagram

5.3.6. 6. Fejlesztetlen torony eladása



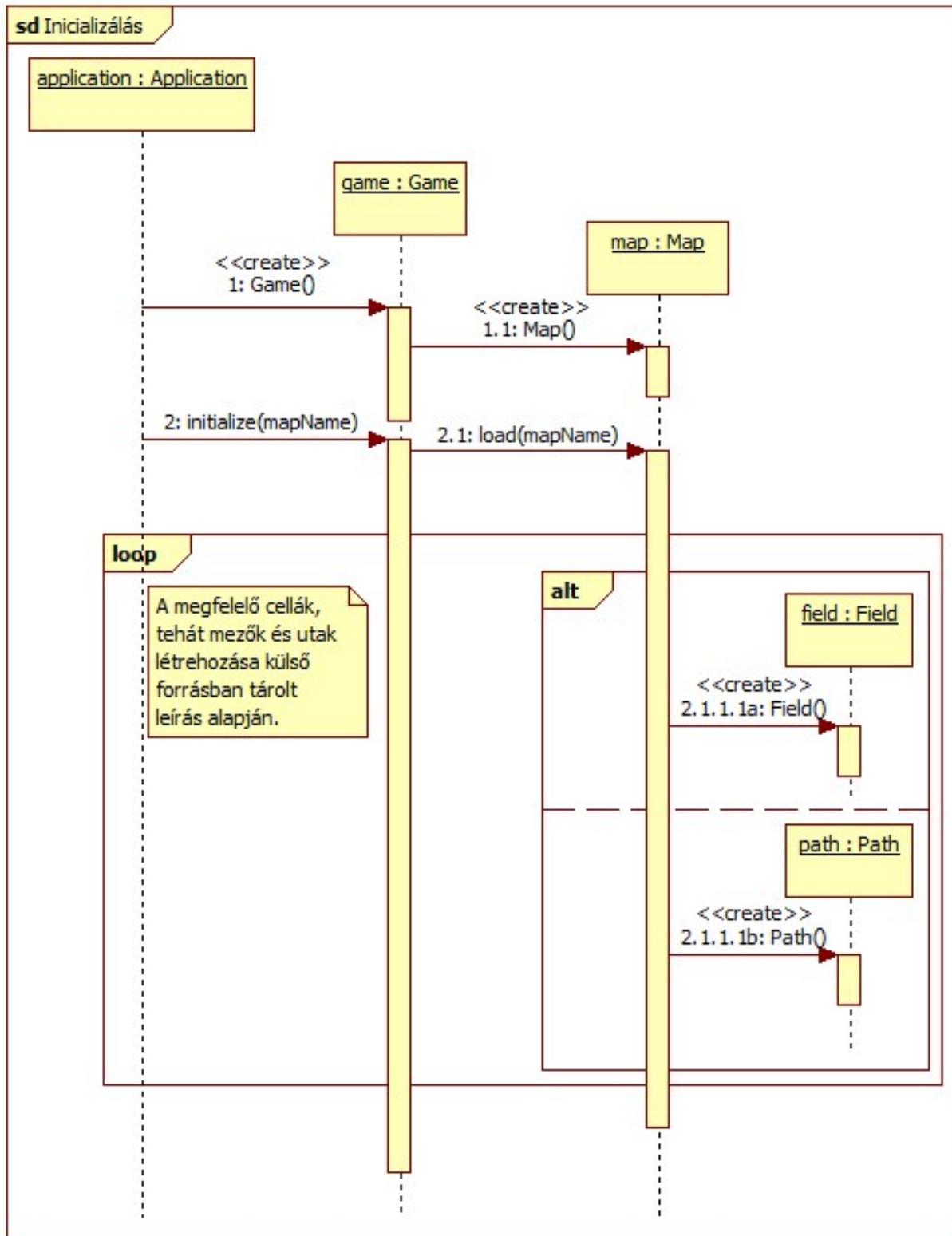
5.8. ábra. Fejlesztetlen torony eladása szekvenciadiagram

5.3.7. 7. Fejlesztett torony eladása



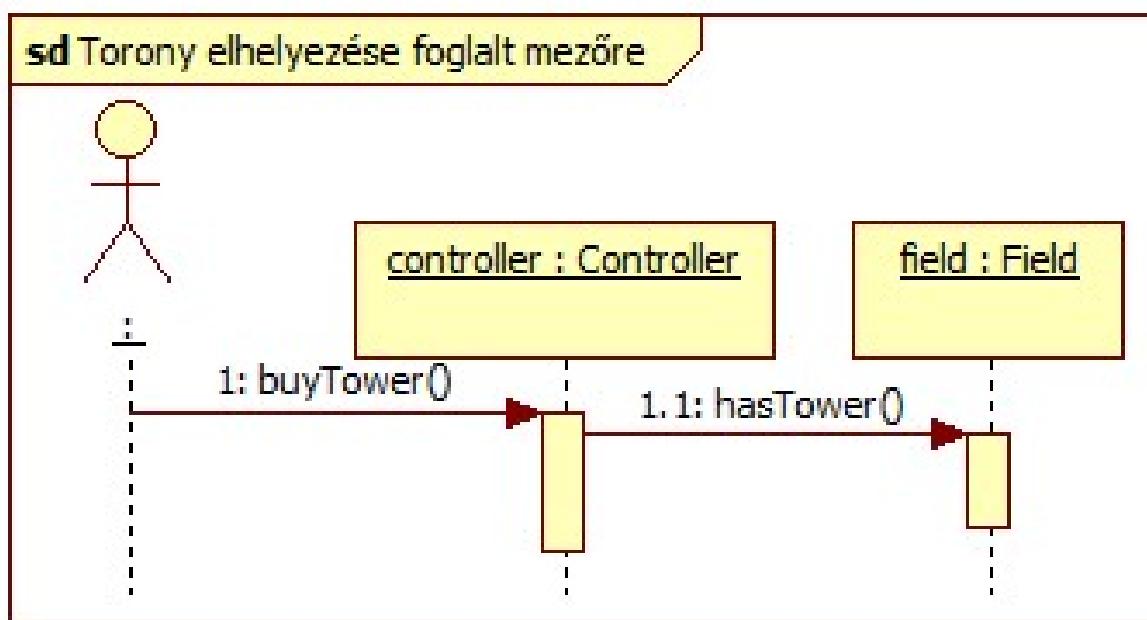
5.9. ábra. Fejlesztett torony eladása szekvenciadiagram

5.3.8. 8. Inicializálás



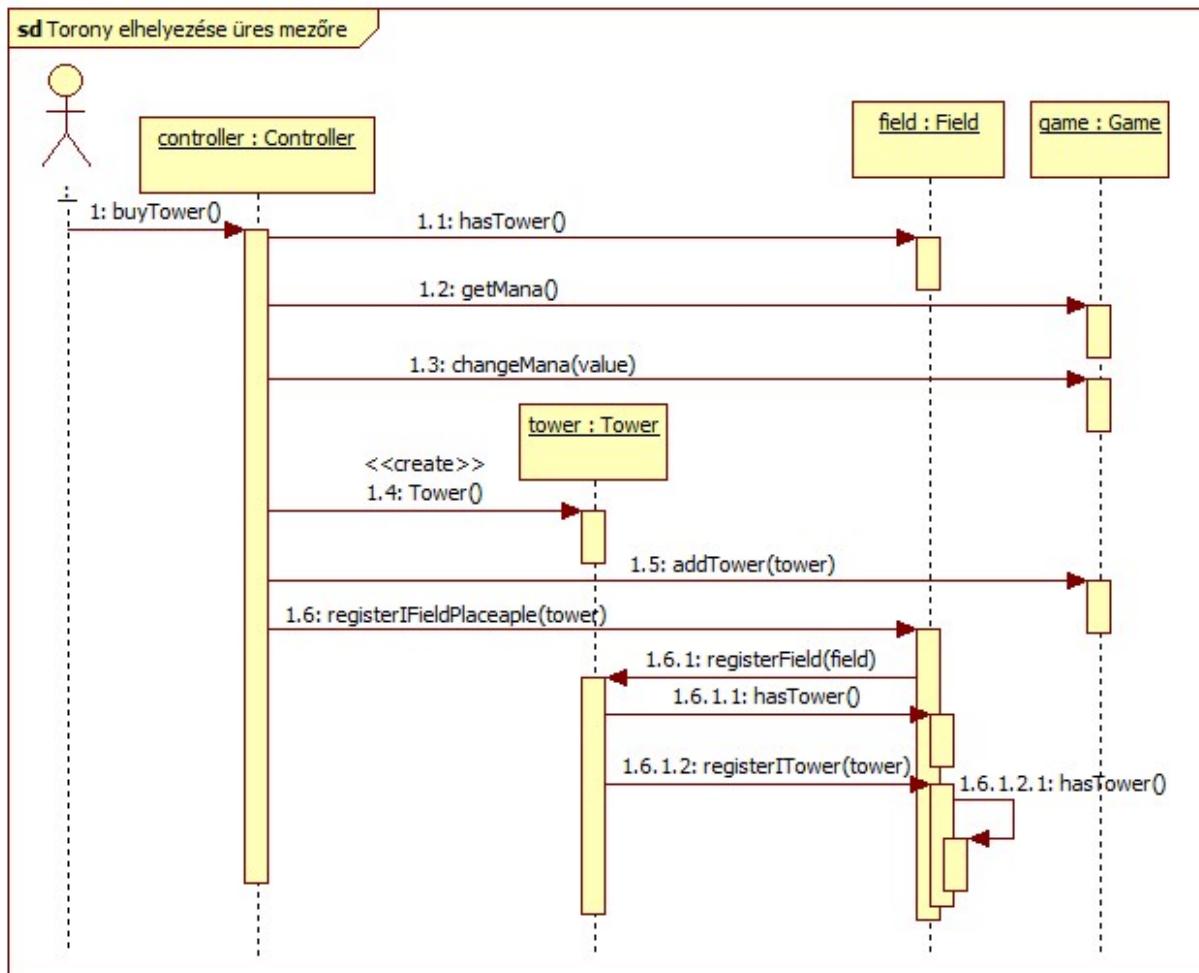
5.10. ábra. Inicializálás szekvenciadiagram

5.3.9. 9. Torony elhelyezése foglalt mezőre



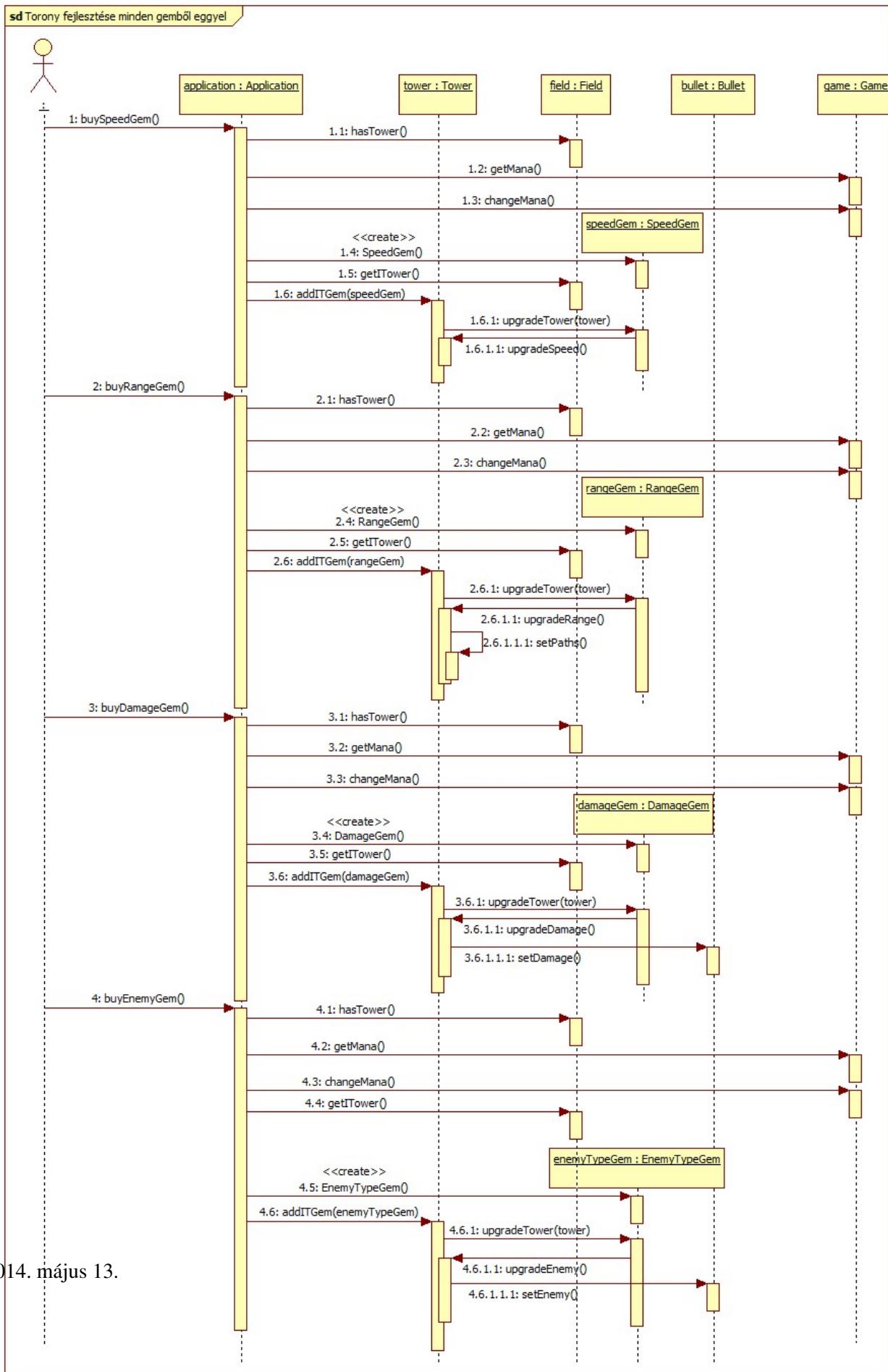
5.11. ábra. Torony elhelyezése foglalt mezőre szekvenciadiagram

5.3.10. 10. Torony elhelyezése üres mezőre

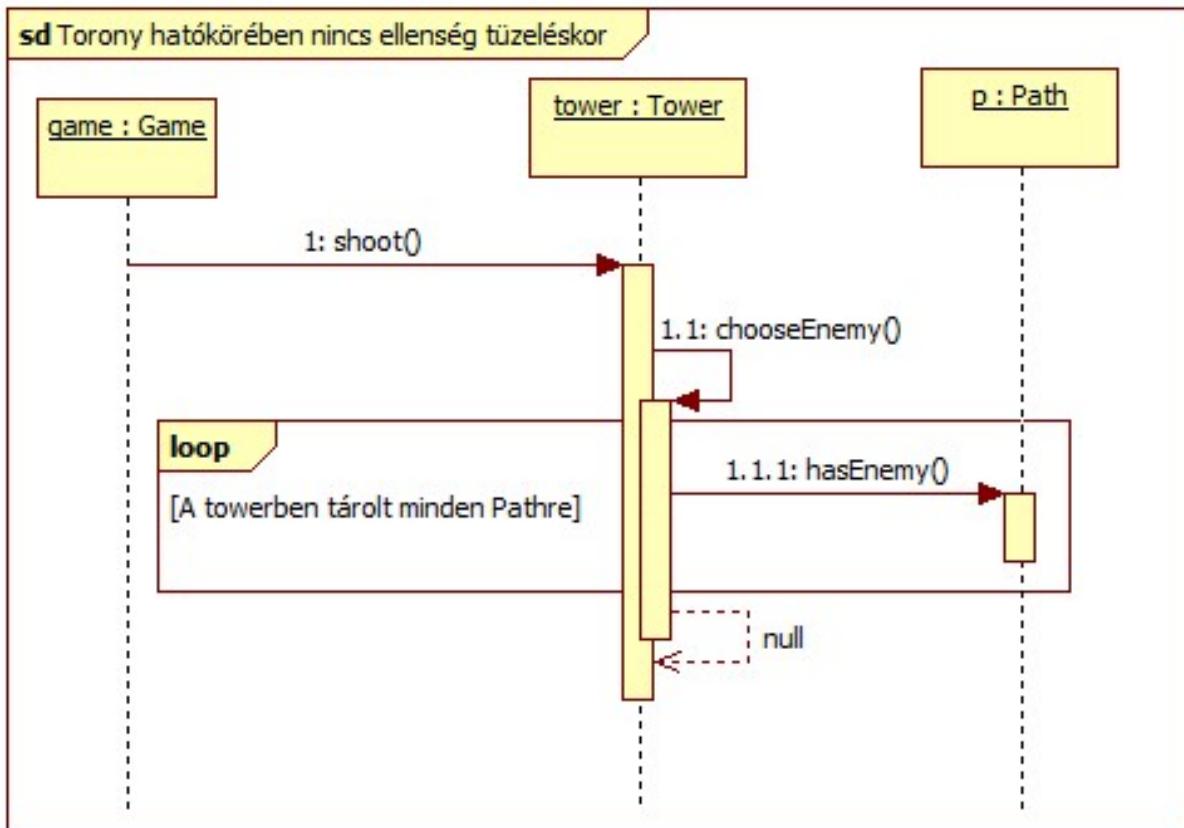


5.12. ábra. Torony elhelyezése üres mezőre szekvenciadiagram

5.3.11. 11. Torony fejlesztése minden gemből eggel

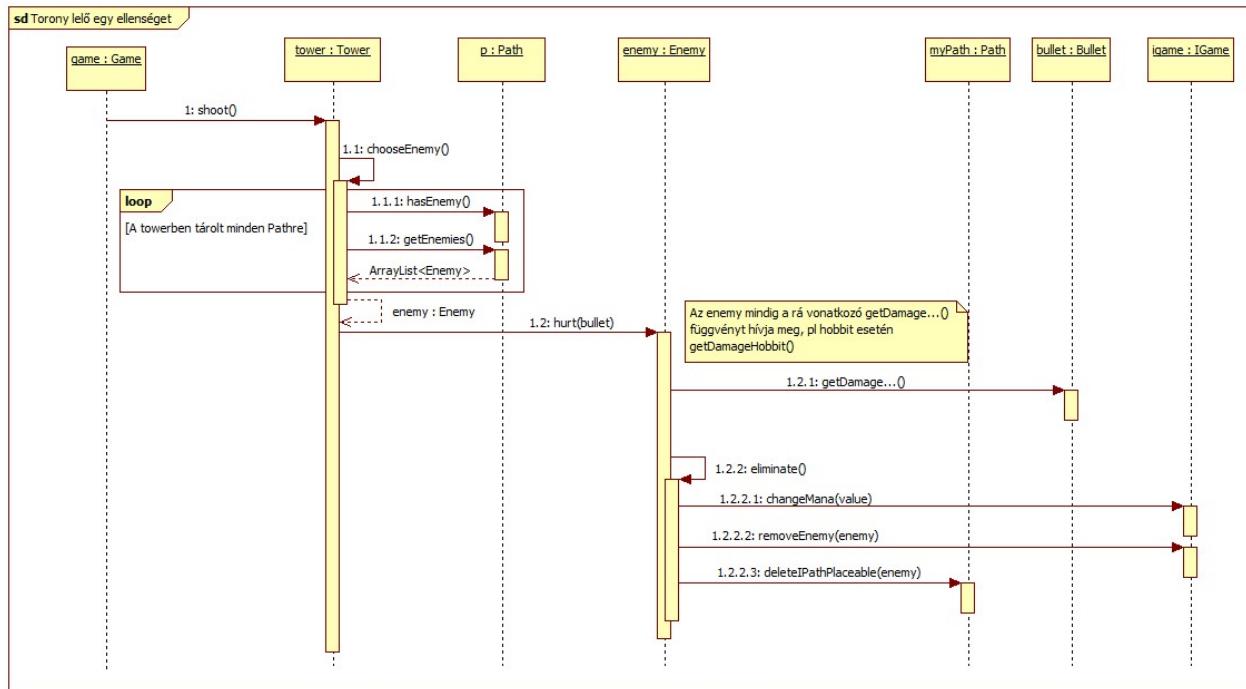


5.3.12. 12. Torony hatókörében nincs ellenség tüzeléskor



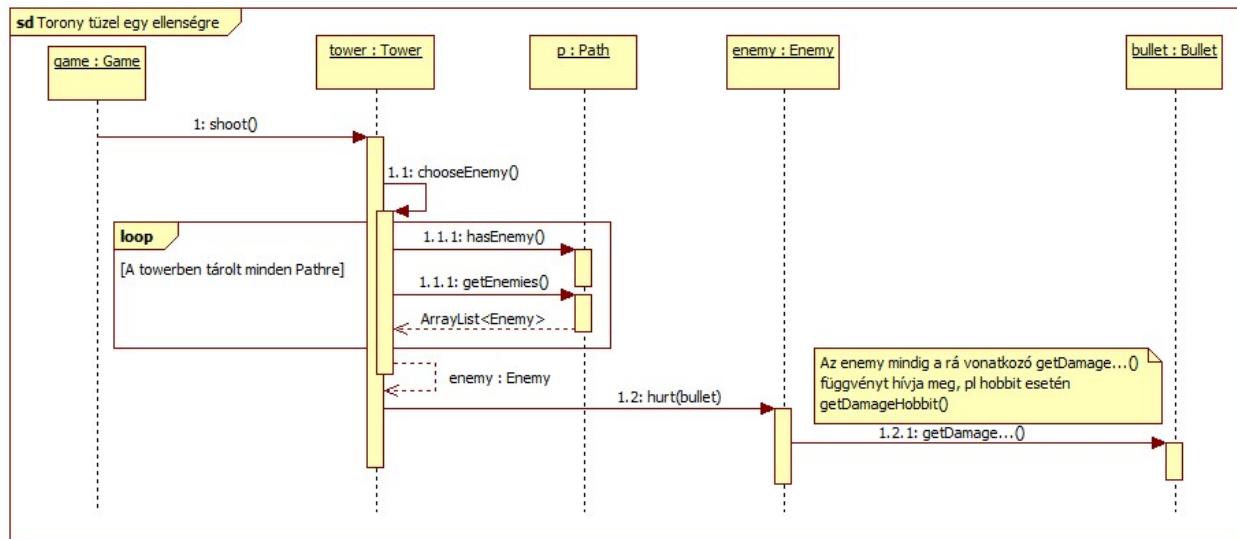
5.14. ábra. Torony hatókörében nincs ellenség tüzeléskor szekvenciadiagram

5.3.13. 13. Torony lelő egy ellenséget fejlesztetlenül



5.15. ábra. Torony lelő egy ellenséget fejlesztetlenül szekvenciadiagram

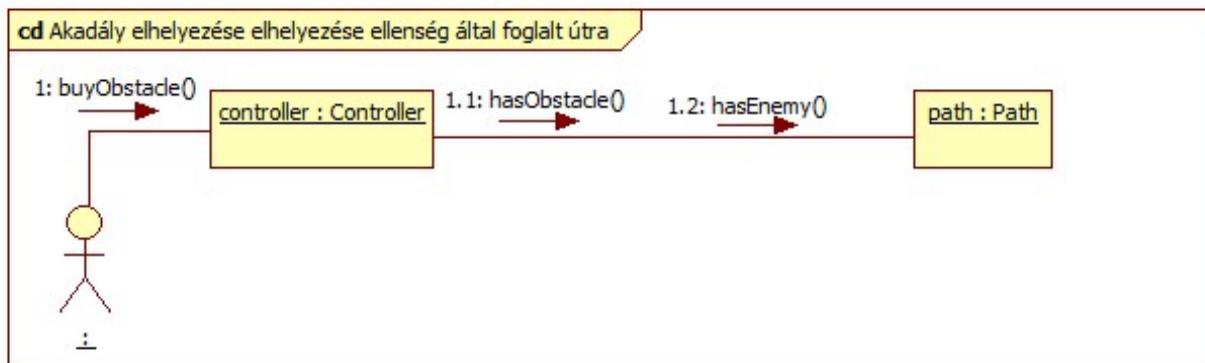
5.3.14. 14. Torony tüzel egy ellenségre fejlesztetlenül



5.16. ábra. Torony tüzel egy ellenségre fejlesztetlenül szekvenciadiagram

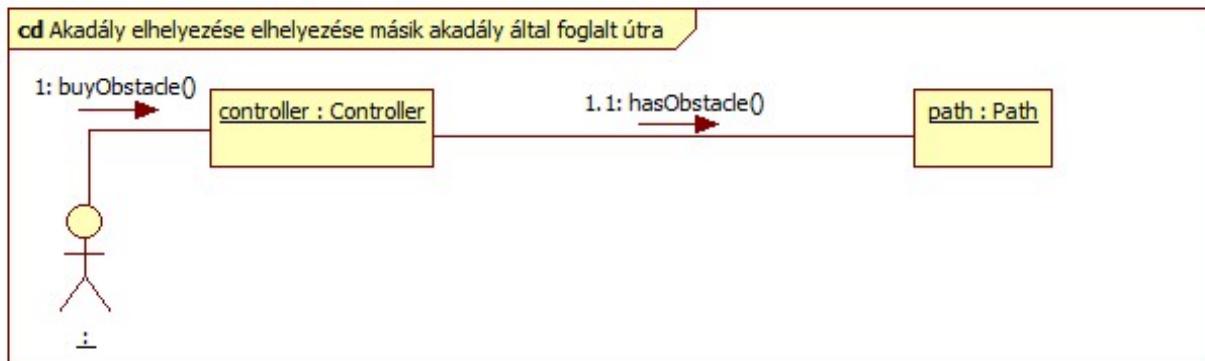
5.4. Kommunikációs diagramok

5.4.1. 1. Akadály elhelyezése ellenség által foglalt útra



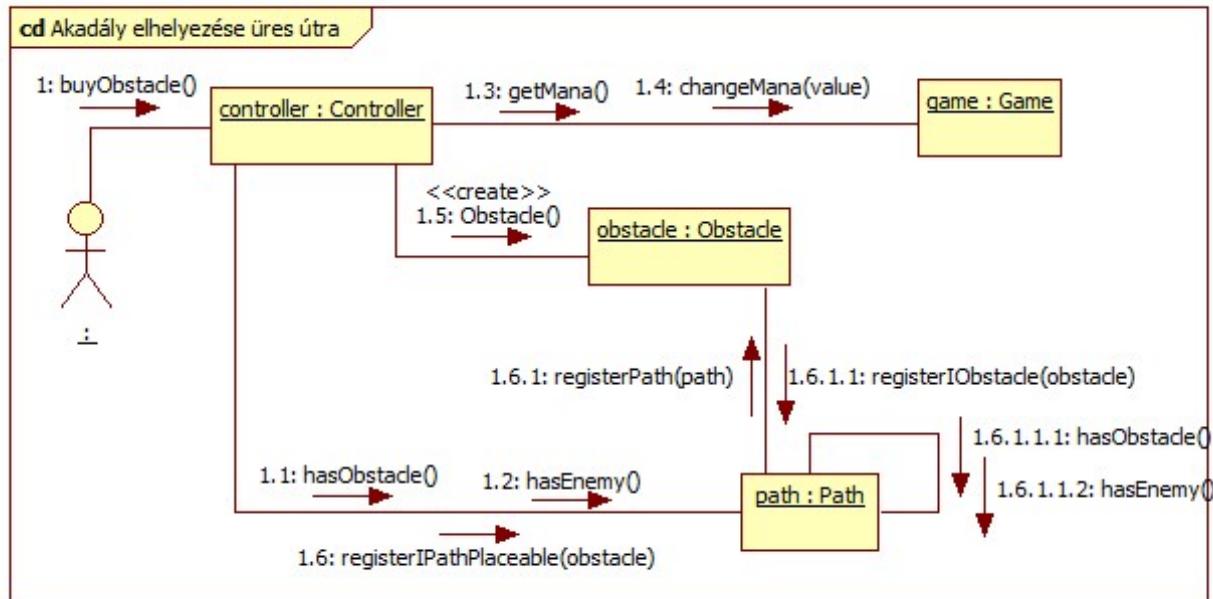
5.17. ábra. Akadály elhelyezése ellenség által foglalt útra kommunikáviós diagram

5.4.2. 2. Akadály elhelyezése másik akadály által foglalt útra



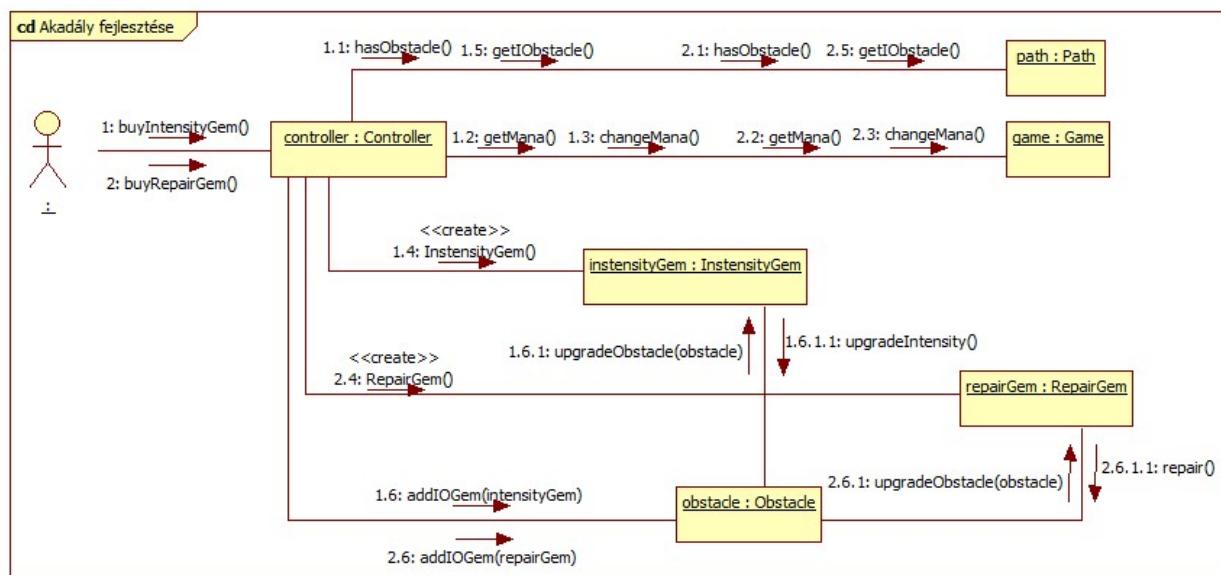
5.18. ábra. Akadály elhelyezése másik akadály által foglalt útra kommunikáviós diagram

5.4.3. 3. Akadály elhelyezése üres útra



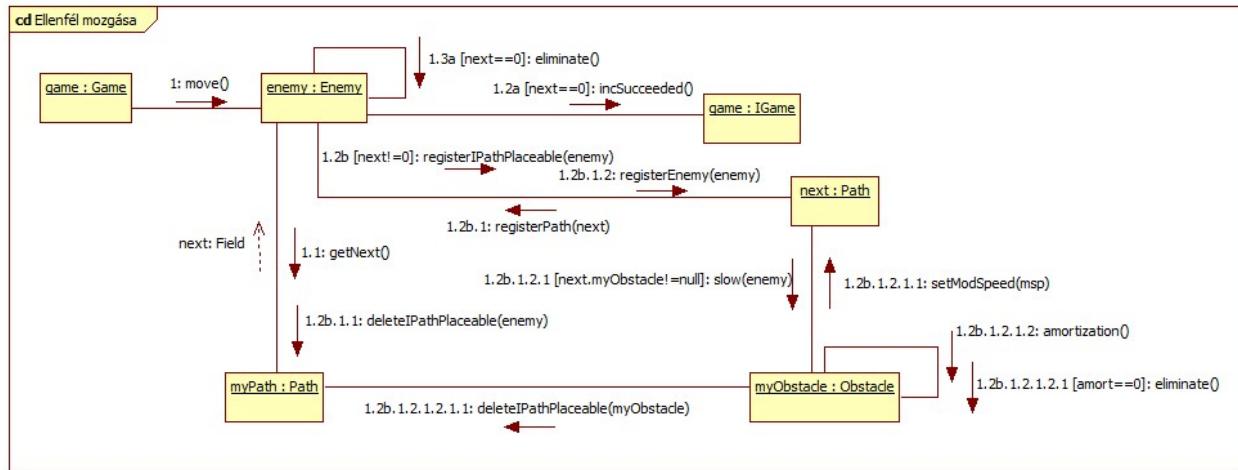
5.19. ábra. Akadály elhelyezése üres útra kommunikáviós diagram

5.4.4. 4. Akadály fejlesztése



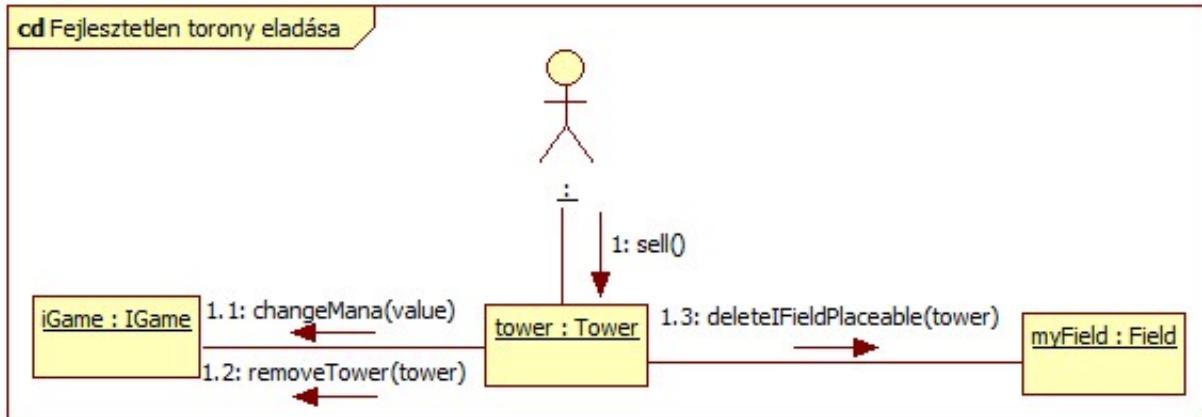
5.20. ábra. Akadály fejlesztése kommunikáviós diagram

5.4.5. 5. Ellenfél mozgása



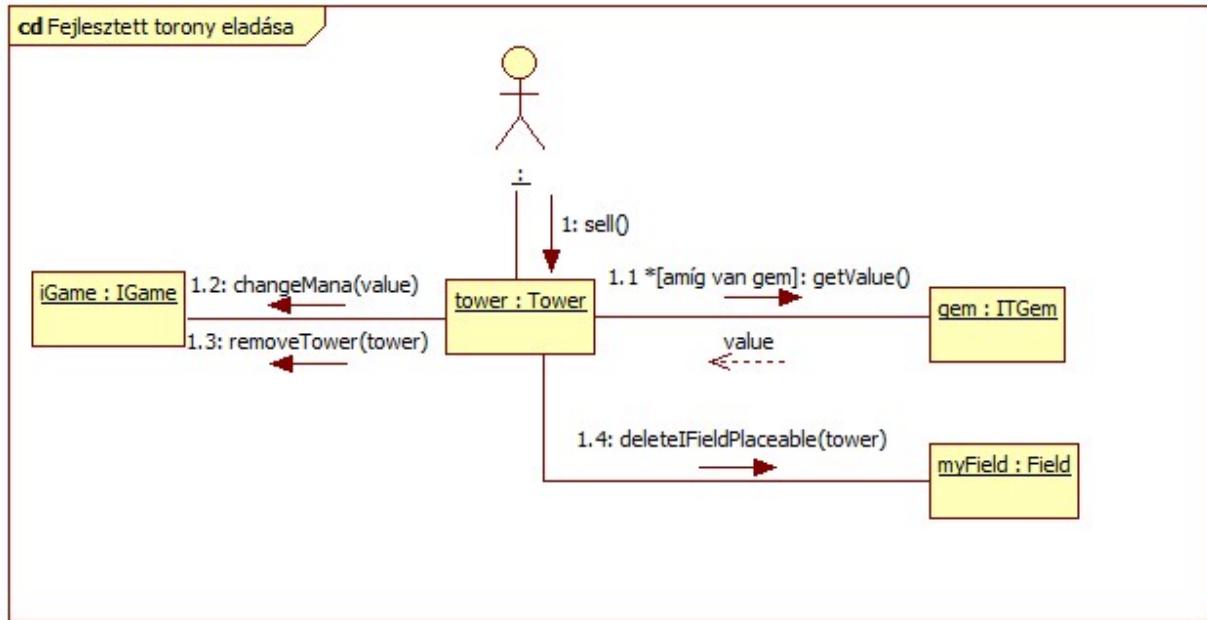
5.21. ábra. Ellenfél mozgása kommunikációs diagram

5.4.6. 6. Fejlesztetlen torony eladása



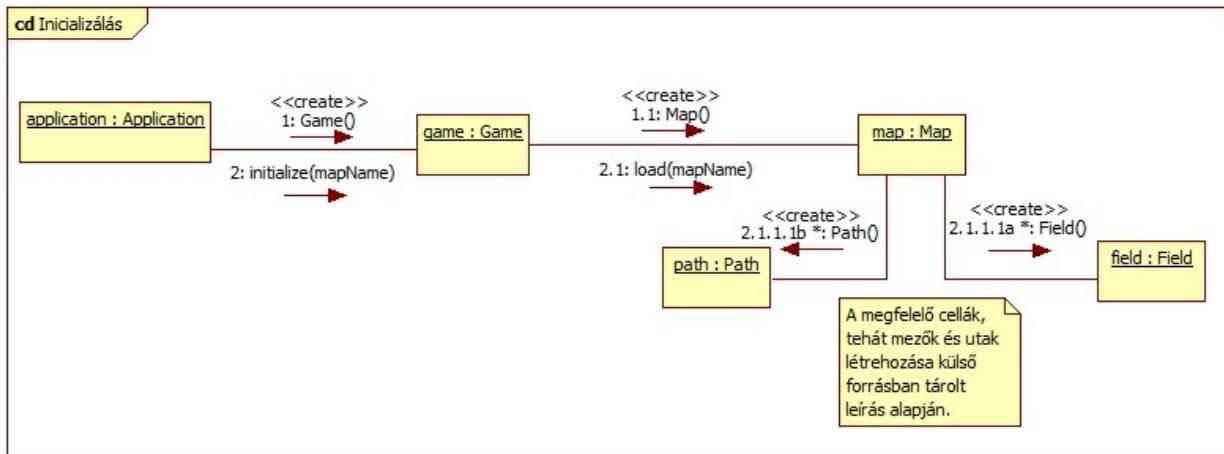
5.22. ábra. Fejlesztetlen torony eladása kommunikációs diagram

5.4.7. 7. Fejlesztett torony eladása



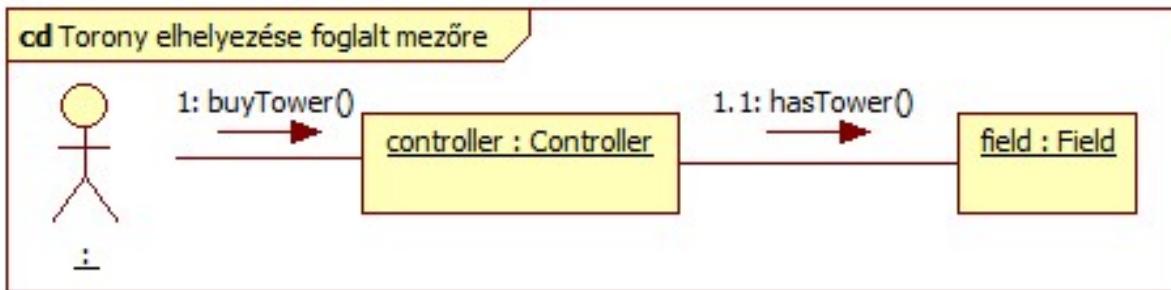
5.23. ábra. Fejlesztett torony eladása kommunikáviós diagram

5.4.8. 8. Inicializálás



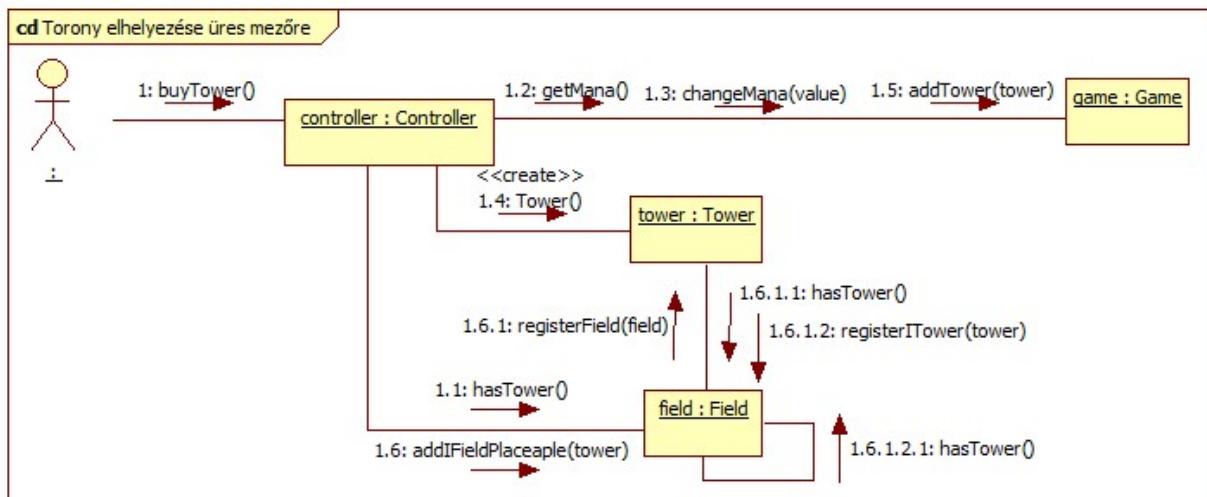
5.24. ábra. Inicializálás kommunikáviós diagram

5.4.9. 9. Torony elhelyezése foglalt mezőre



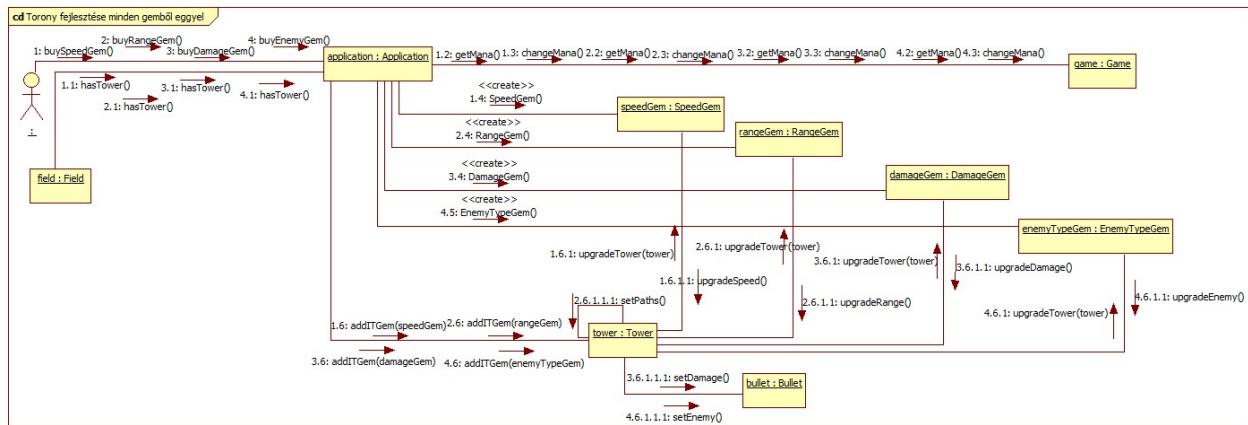
5.25. ábra. Torony elhelyezése foglalt mezőre kommunikáviós diagram

5.4.10. 10. Torony elhelyezése üres mezőre



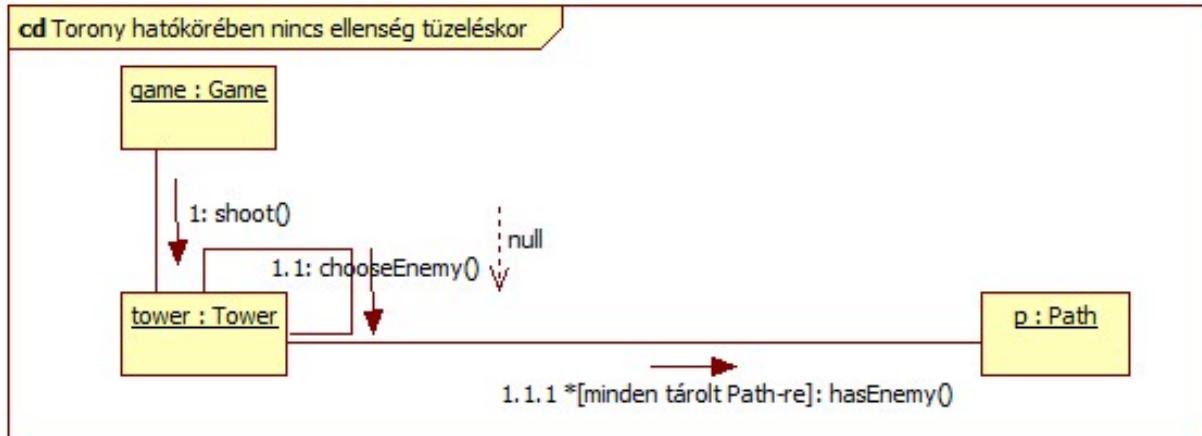
5.26. ábra. Torony elhelyezése üres mezőre kommunikáviós diagram

5.4.11. 11. Torony fejlesztése minden gemből eggel



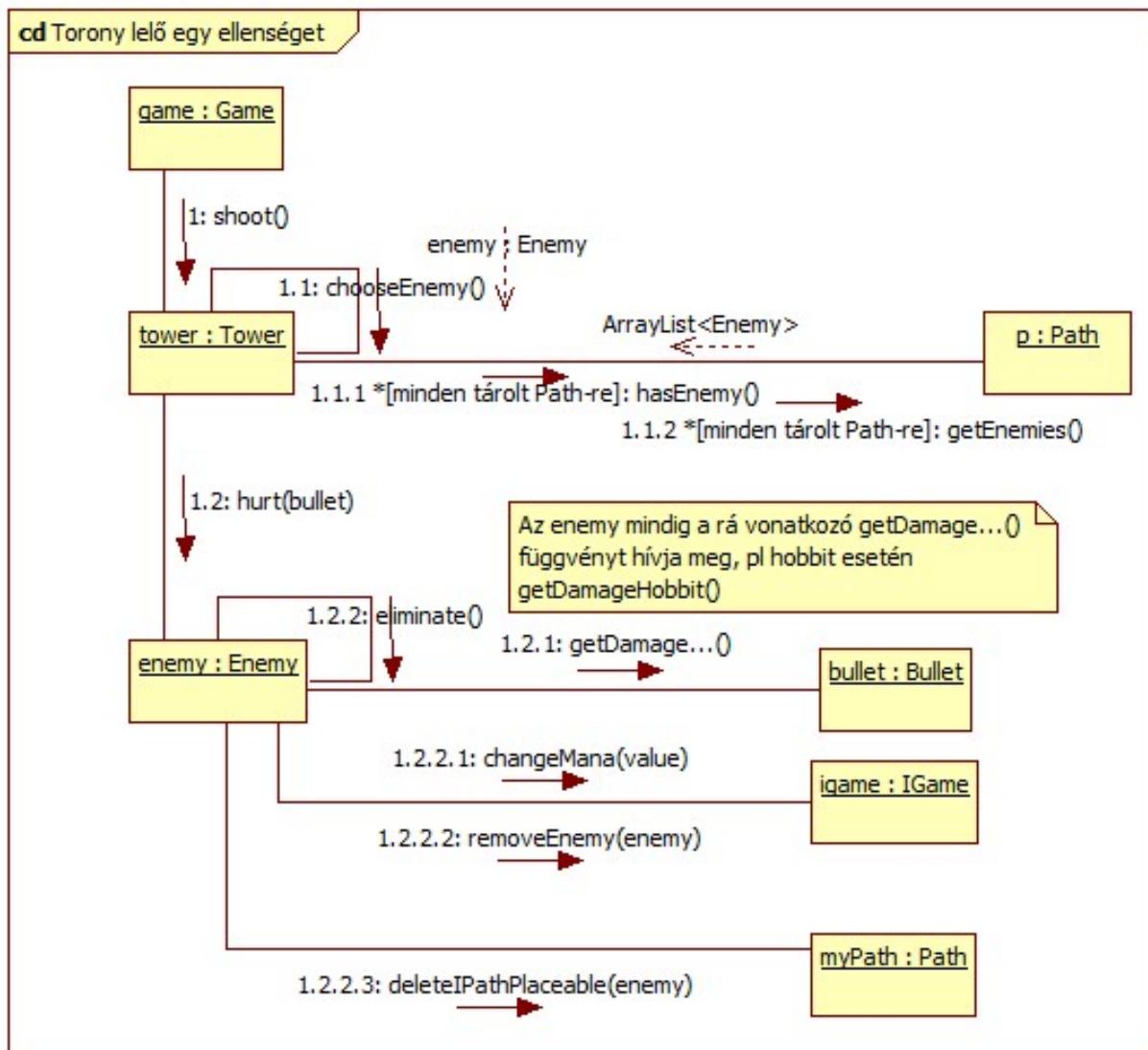
5.27. ábra. Torony fejlesztése minden gemből eggel kommunikációs diagram

5.4.12. 12. Torony hatókörében nincs ellenség tüzeléskor



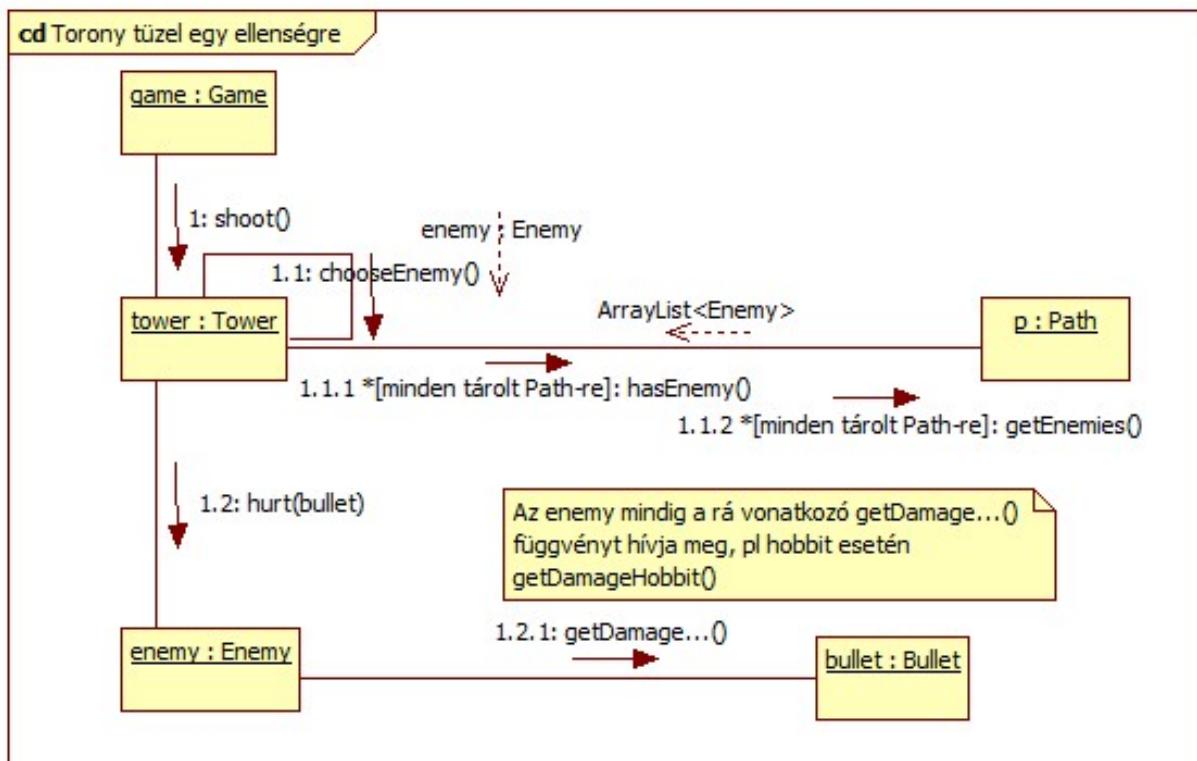
5.28. ábra. Torony hatókörében nincs ellenség tüzeléskor kommunikációs diagram

5.4.13. 13. Torony lelő egy ellenséget fejlesztetlenül



5.29. ábra. Torony lelő egy ellenséget fejlesztetlenül kommunikációs diagram

5.4.14. 14. Torony tüzel egy ellenségre fejlesztetlenül



5.30. ábra. Torony tüzel egy ellenségre fejlesztetlenül kommunikációs diagram

5.5. Napló

Kezdet	Időtartam	Résznevők	Leírás
2014.03.12 08:15	1,5 óra	Fuksz Rédey Elekes Seres Nagy	Konzultáció
2014.03.14 12:00	2 óra	Fuksz Rédey Elekes	Értekezlet. Lényeges use case-ek, szekvencia diagramok kiválasztása.
2014.03.15. 11:00	1 óra	Rédey	Use-case diagrammok
2014.03.15. 11:40	30 perc	Seres	Tornyok és akadályok elhelyezésének use-case leírásai
2014.03.15. 12:00	1,5 óra	Elekes	5.2-es pont. Tesztelő osztály implementálás
2014.03.16. 13:50	1,5 óra	Seres	Use-case-ekhez tartozó kommunikációs diagramok és szekvencia diagramok készítése
2014.03.16. 14:00	1,5 óra	Elekes	Tesztesetek, Controller osztály
2014.03.16. 19:30	1,5 óra	Rédey	Tesztesetek és use-casek megfeleltetése, use case diagram módosítása

Kezdet	Időtartam	Részttvevők	Leírás
2014.03.16. 22:00	3,5 óra	Seres	A fennmaradó use-case-ekhez tartozó kommunikációs diagramok és szekvencia diagramok készítése
2014.03.17. 01:30	2 óra	Fuksz	Dokumentáció véglegesítése

6. Szkeleton beadás

6.1. Fordítási és futtatási útmutató

6.1.1. Fájllista

Fájl neve	Méret	Keletkezés ideje	Tartalom
Application.java	454	2014.03.11.	Névvkel megegyező osztály.
Bullet.java	1956	2014.03.11.	Névvkel megegyező osztály.
Cell.java	642	2014.03.11.	Névvkel megegyező osztály.
Controller.java	4503	2014.03.11.	Névvkel megegyező osztály.
DamageGem.java	859	2014.03.11.	Névvkel megegyező osztály.
Dwarf.java	589	2014.03.11.	Névvkel megegyező osztály.
Elf.java	576	2014.03.11.	Névvkel megegyező osztály.
Enemy.java	2013	2014.03.11.	Névvkel megegyező osztály.
EnemyTypeGem.java	765	2014.03.11.	Névvkel megegyező osztály.
Field.java	2026	2014.03.11.	Névvkel megegyező osztály.
Game.java	2489	2014.03.11.	Névvkel megegyező osztály.
Gem.java	412	2014.03.11.	Névvkel megegyező osztály.
Hobbit.java	952	2014.03.11.	Névvkel megegyező osztály.
Human.java	587	2014.03.11.	Névvkel megegyező osztály.
IFieldPlaceable.java	300	2014.03.11.	Névvkel megegyező osztály.
IGame.java	486	2014.03.11.	Névvkel megegyező osztály.
IntensityGem.java	774	2014.03.11.	Névvkel megegyező osztály.
IObstacle.java	382	2014.03.11.	Névvkel megegyező osztály.
IOGem.java	260	2014.03.11.	Névvkel megegyező osztály.
IPathPlaceable.java	303	2014.03.11.	Névvkel megegyező osztály.
ITGem.java	278	2014.03.11.	Névvkel megegyező osztály.
ITower.java	328	2014.03.11.	Névvkel megegyező osztály.
Map.java	1108	2014.03.11.	Névvkel megegyező osztály.
Obstacle.java	2398	2014.03.11.	Névvkel megegyező osztály.
Path.java	4024	2014.03.11.	Névvkel megegyező osztály.
RangeGem.java	825	2014.03.11.	Névvkel megegyező osztály.
RepairGem.java	653	2014.03.11.	Névvkel megegyező osztály.
SkeletonTester.java	10032	2014.03.11.	A szkeleton programot tesztelhetővé tevő program.
SpeedGem.java	711	2014.03.11.	Névvkel megegyező osztály.
Tower.java	3984	2014.03.11.	Névvkel megegyező osztály.
SkeltonCompiler.bat	119	2014.03.23.	Parancssoros fordítást végzi.
SkeltonRun.bat	97	2014.03.23.	Futtatja a lefordított szkeletonet.

6.1.2. Fordítás

A szkeleton fordítását legegyszerűbben a parancssorból tudjuk elvégezni. A parancssorból közvetlenül elérhetőnek kell lennie a javac.exe fordítóprogramnak. A fordítást követően létrehozásra kerülnek a lefordított .class kiterjesztésű fájlok.

A fordítást egyszerűbb a program mellé mellékelt SkeletonComplier.bat fájl megnyitásával elvégezni.

6.1.3. Futtatás

A szkeleton futtatását a java.exe futtatóprogrammal lehet elvégezni. A parancssorból közvetlenül elérhetőnek kell lennie a java.exe fordítóprogramnak.

A futtatást egyszerűbb a program mellé mellékelt SkeletonRun.bat fájl megnyitásával elvégezni.

6.2. Értékelés

Tag	Munkaóra	Munka százalékban	Aláírás
Elekes	44,4 óra	22 %	
Fuksz	36,25 óra	22 %	
Nagy	20,5 óra	10 %	
Rédey	38,75 óra	23 %	
Seres	41,6 óra	23 %	

6.3. Napló

Kezdet	Időtartam	Résznevők	Leírás
2014.03.18.	8 óra	Elekes	Tesztelő program, tesztesetek implementálása. Controller osztály implementálása.
2014.03.20.	2,5 óra	Elekes	EnemyMove, EnemySucceeded, Inicializálás tesztesetek. Javítások a programban.
2014.03.22. 11:00	1 óra	Nagy	Konzolos menü a szekvenciadiagramok alapján.
2014.03.22. 20:00	7 óra	Seres	Szekvenciadiagramok és kommunikációs diagramok javítása
2014.03.23. 11:00	1 óra	Nagy	Az 5-ös doksi teszteseteinek aktualizálása.
2014.03.23. 20:00	30 perc	Elekes	Napló szummázása, 6.1.1 pont kitöltése.
2014.03.23. 18:00	1 óra	Fuksz	Batch fájlok a fordításhoz. 6.1.2 és 6.1.3 pontok kitöltése
2014.03.24. 01:00	2 óra	Fuksz	Dokumentáció véglegesítése

7. Prototípus koncepciója

7.0. Specifikáció változás

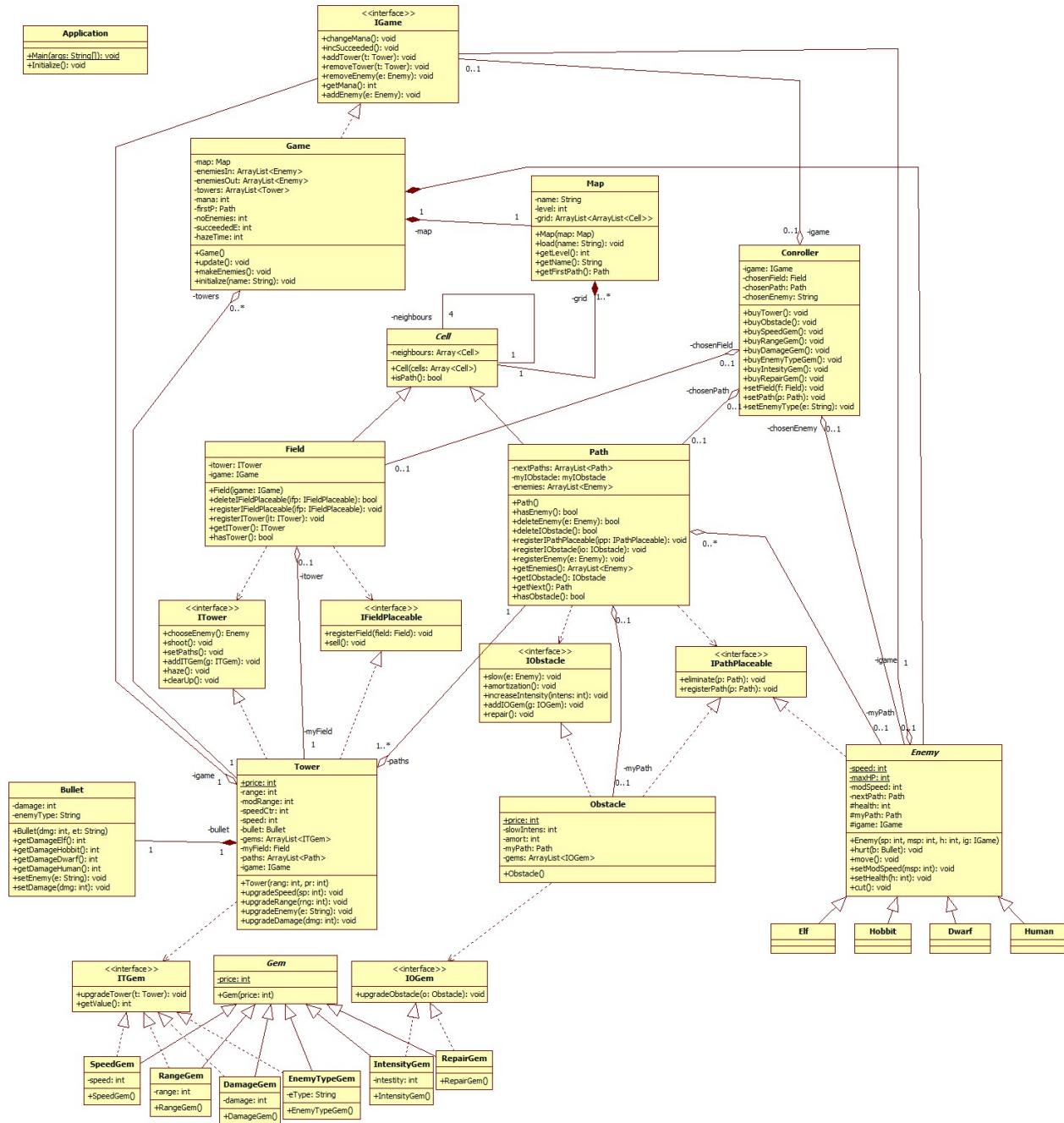
7.0.1. Módosítás

A tornyokra időnként köd ereszkedik, aminek következtében a látás erősen lecsökken. Ez hatással van a lövésre. A játékosok által járható útvonalon lehetnek elágazások és becsatlakozások. Az elágazásokon az egyes játékosok véletlenszerűen mennek a különböző irányokba. A tornyokban elvétve lehetnek olyan lövedékek, amelyek az eltalált játékost kettőbe vágják. A két játékos egymástól függetlenül él tovább, csökkentett életerővel.

A fentiek fényében a következő módosításokat hajtottuk végre:

- A köd torony hatósugarán változtat, így amíg a köd tart, addig a torony rövidebbre lát el, mint addig. Köd alatt is lehet fejleszteni a tornyot, ekkor ugyanúgy érvényre jut, de a köd által módosított hatókör fog nőni, majd a köd elmúltával olyan érték áll be, mintha nem lett volna köd és fejlesztették volna a tornyot. Köd véletlenszerűen keletkezik és minden tornyot érint. A köd időtartamát Game osztály hazeTime attribútuma szabja meg. Ez mindig ugyanannyi idő, az attribútum csak arról gondoskodik, hogy ha letelt, akkor visszaálljon minden (számol). A torony interfészébe bekerültek a haze() és clearUp() metódusok, előbbi ködbe borít, utóbbi kitisztít. A torony kapott egy modRange attribútumot, aminek segítségével kényelmesen kezelhető a ködbe borulás.
- Az elágazások és becsatlakozások tekintetében nem változtattunk, a modellen, mert az eddigiek alapján ezt már eddig is tudta. Vagyis amikor az Enemy továbblép, meghívódik a getNext() metódus, ami a következő Path címével tér vissza annak a Path-nak a nextPaths listájából, amelyikről lépni akar. Ha a listában több Path van, az pont az elágazásnak felel meg, ekkor véletlenszerűen adunk egy címet.
- Amikor egy Enemy-n meghívjuk a hurt metódust, átadunk egy Bulletet, amin az adott típusú Enemy meghívja a rá jellemző getDamage metódust, hogy megtudja a sebzést. Időnként 0-t kap majd, aminek hatására a meghívódik az Enemy-ben felvett cut() metódus, ami létrehoz egy ugyanilyen típusú Enemy-t (felüldefiniálás révén), aminek feleakkora életerőt ad, mint amennyi neki van, valamint a sajátját is felezi, továbbá minden egyéb attribútum értékét átmásolja az „ikertestvérébe”. Ezen felül, mivel az „ikertestvér”nek is be kell kerülnie a pályán lévő enemy-k közé, hogy vezérelhessük, az IGame interfészbe tettünk egy addEnemy fv-t, hogy az Enemyi betehesse az új Enemy-t.

7.0.2. Módosított osztálydiagram

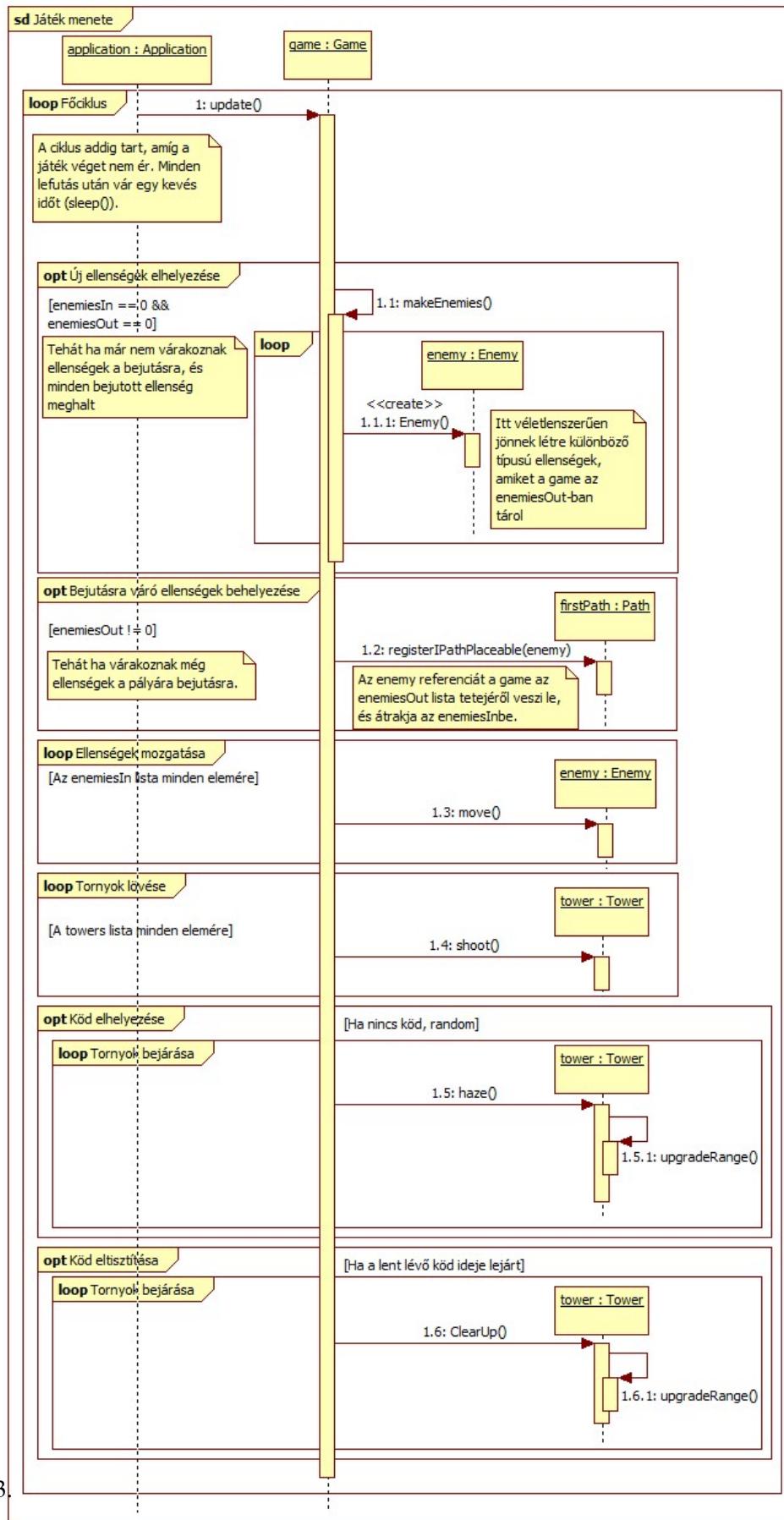


7.1. ábra. Módosított osztálydiagram

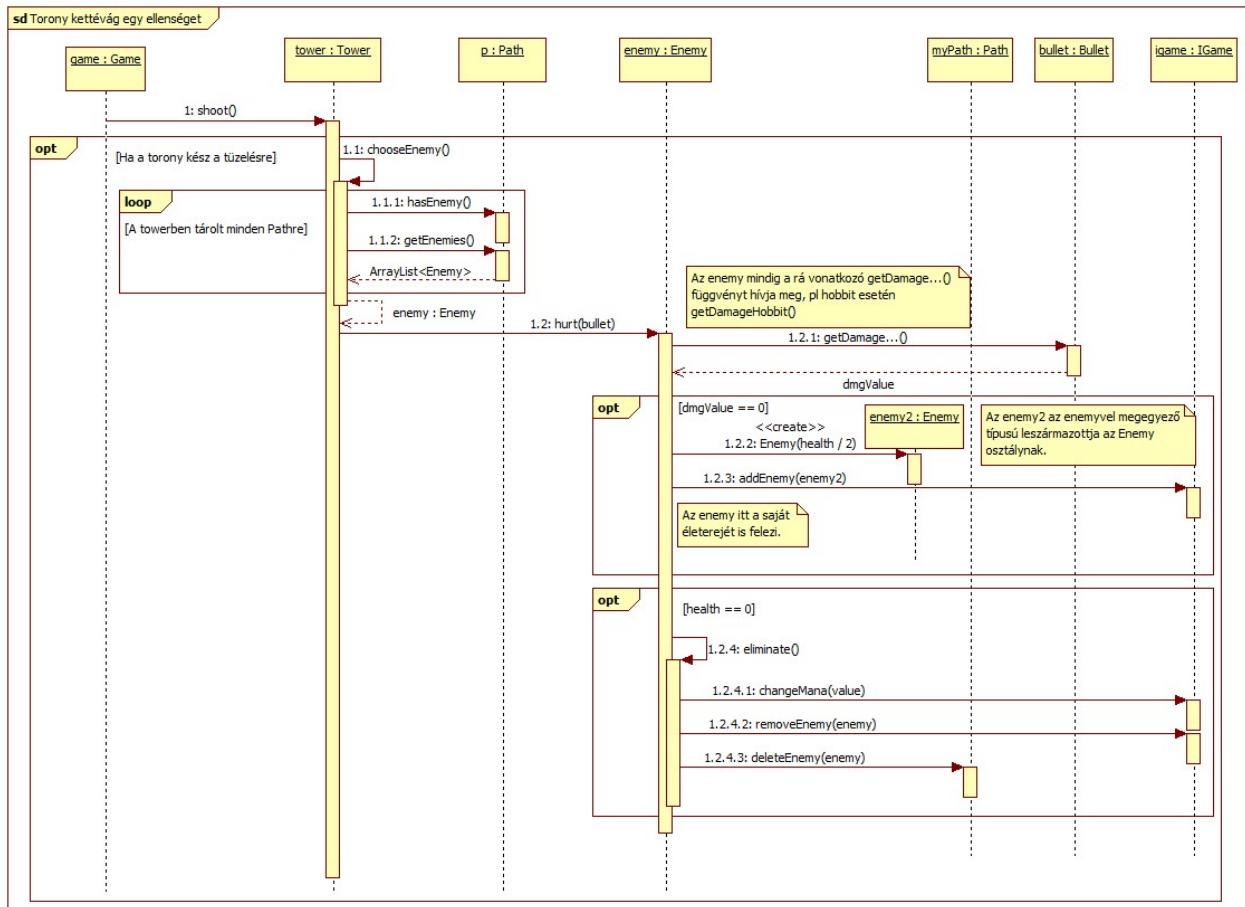
7.0.3. Módosított szekvenciadiagramok

A módosított diagramok alapjai az analízis modell szekvenciadiagramjai.

Játék menete



Torony kettévág egy ellenséget



7.3. ábra. Torony kettévág egy ellenséget szekvenciadiagram

7.1. Prototípus interface-definíciója

7.1.1. Az interfész általános leírása

Az interfész teljesen karakteres alapú, a szabványos ki- és bemenetet használja. Indítás után a 7.1.2 pontban leírt parancsokat lehet megadni, és a 7.1.3 pontban leírt módon kapjuk a kimenetet.

Lehetséges a ki- és bemenet átirányítása is, így például előre összeállított teszteseteket elmenthetünk fájlban, és átirányítással megadhatjuk a programnak, továbbá a kimenetét is fájlba irányíthatjuk a könnyebb elemzés érdekében.

Az átirányítás a hagyományos módon történik, például a

```
java "osztalynev" < bemenet.txt > kimenet.txt
```

parancs hatására a program megkapja soronként a bemenet.txt tartalmát, és a kimenet.txt fájlba írja a kimenetét.

7.1.2. Bemeneti nyelv

A bemeneti nyelv azoknak az utasításoknak a halmazát jelenti, amikkel a Prototípust vezérelni, és ezáltal tesztelni lehet.

Egy sorban 1 utasítás megadására van lehetőség. Az üres sorokat a parser figyelmen kívül hagyja. minden parancs esetén a paramétereket szóközzel elválasztva kell felsorolni. Az utasítás végét nem kell pontosvesszővel lezárni, mert minden utasítás új sorba kerül.

- loadmap

Leírás: Betölti a pályát.

Opciók: #1 → filename

Példa: loadmap tesztpalya.txt

- update

Leírás: Az idő műlását vezérli. Egy egész szám a paramétere, hogy hány tick fusson le.

Opciók: #1 → [0-9]+

Példa: update 10

- draw

Leírás: Kirajzolja a pályát, és a fontosabb adatokat:

- mennyi manája van a játékosnak
- egy torony mennyibe kerül
- egy akadály mennyibe kerül
- egy kristály mennyibe kerül

Opciók: –

Példa: draw

- buy

Leírás: Objektumok (torony, akadály) elhelyezése a pályán, és fejlesztésük Gemekkel.

Opciók: #1 → [tower | obstacle | gem] (mit veszünk)

#2 → x-y (hova vesszük)

#3 → [speed | range | damage | enemy | intensity | repair] (milyen gem)

#4 → [elf | hobbit | dwarf | human] (milyen ellenségre)

3. paraméter csak Gem esetén létezik, míg 4. paraméter csak Enemy-Gem esetén.

Példa: – buy tower 5-3

– buy obstacle 10-12

– buy gem 5-3 enemy hobbit

- random

Leírás: A véletlenszerűséget tudja ki és bekapcsolni.

Opciók: #1 → [true | false]

Példa: random true

- exit

Leírás: Kilép a programból.

Opciók: –

Példa: exit

A pálya formátuma

A pályát karakterekkel rajzoljuk ki. Szabályok: pálya széle mindenhol "field", kivéve "entry point" és "exit point"

x field

en entry point

r right

l left

u up

d down

ru right, up

rl right, left

rd right, down

ul up, left

ud up, down

dr down, right

rul right, up, left

rud right, up, down

rdl right, down, left

dul down, up, left

ex exit point

7.1.3. Kimeneti nyelv

Egy bemeneti parancs után ha a programnak lényeges függvénye hívódik meg, kiírja azt.

Ha helyénvaló, az objektum katalógus beli nevét is kiírja.

Ha több függvényhíváson keresztül ír ki, minden függvény mélyén egy tabbal beljebb ír ki.

Bemenet: loadmap

Kimenet: nincs

Bemenet: update <n>

Lehetséges kimenetek:

- update — minden frissítés elején kiírja
- makeEnemies — ha a pályán kívül várakozó és bent lévő ellenségekről tárolt tömb is üres.
- *katalógus név* created — egyes ellenségek konstruktorában kiírja a katalógus nevét
- *katalógus név* move — ha egy ellenség lép, ezt írja ki

- katalógus név crossroad — ha ellenségekkelághoz ért
- katalógus név shoot katalógus név target — ha egy torony ellenségre lő, ezt írja ki
- haze — amikor köd ereszkedik a tornyokra

Példa:

```
update 2
update
    makeEnemies
        Hobbit0 created
        Hobbit1 created
        Dwarf0 created
update
    Hobbit0 move
    Hobbit0 crossroad
```

Bemenet: draw

Kimenet:

- Fentebb definiált módon, táblázatosan kirajzolja a pályát. A pálya egy celláját egy számpár határozza meg, az oszlop és sor koordinátája.
- Your mana: n – a játékos manája
- Tower price: n – egy torony ára
- Obstacle price: n – egy akadály ára
- Gem price: n – egy kristály ára

Példa:

```
draw
    0   1   2   3   4   5
0   x   x   x   x   x   x
1   en   r   rd   r   d   x
2   x   x   d   x   d   x
3   x   x   d   x   d   x
4   x   x   r   r   r   ex
5   x   x   x   x   x   x
Your mana: 100
Tower price: 20
Obstacle price: 10
Gem price: 50
```

Bemenet: buy <ctype><coord><gem><enemy>

Lehetséges kimenetek:

- katalógus név created – a létrejövő objektum kiírja a katalógusban tárolt nevét
- cell occupied – ha a kapott koordinátán nem lehet elhelyezni az objektumot

Példa1:

```
buy tower 5-3
Tower0 created
```

Példa2:

```
buy gem 5-3 enemy hobbit
HobbitGem0 created
```

Bemenet: random
 Kimenet: nincs

Bemenet: exit
 Kimenet: nincs

7.2. Összes részletes use-case

Use-case neve	Játék inicializálása
Rövid leírás	A játék indulása után, inicializálja a változókat.
Aktorok	Application, Game
Forgatókönyv	

Use-case neve	Ellenség lépése
Rövid leírás	Egy ellenség az egyik celláról a másikra lép.
Aktorok	Game
Forgatókönyv	Az enemy elkéri a cellájától a következő cellát, törölte magát az aktuális celláról, és beregisztráltatja magát a következőre.

Use-case neve	Ellenség meghal
Rövid leírás	Egy ellenség meghal, tehát teljesen eltűnik.
Aktorok	Game
Forgatókönyv	Az enemy törölte magát az összes enemy listájából (Game), és az aktuális cellájából is.

Use-case neve	Ellenség bejut a végzet hegyéhez
Rövid leírás	Egy ellenség bejut a végzet hegyéhez. A játék vége.
Aktorok	Game
Forgatókönyv	Annyiban tér el a sima mozgástól, hogy a Game ellenőrzi, hogy Szarumán varázslata elfogyott-e már, és ha igen, akkor véget ér a játék.

Use-case neve	Ellenség elágazáshoz ér
Rövid leírás	Egy ellenség egy út-celláról legalább 2 másik cellára léphet tovább.
Aktorok	Game
Forgatókönyv	Az ellenség mozgásának egy speciális esete. Amikor az enemy elkéri a Path-tól a következő cellát, akkor beállítástól függően vagy determinisztikusan a listában szereplő első cellát kapja meg, vagy veletlenszerűen a listából egyet.

Use-case neve	Torony megvétele
Rövid leírás	Üres mezőre tornyot vesz a felhasználó.
Aktorok	Controller, Tower

Forgatókönyv	Először a Controller setField metódusával kiválaszt a felhasználó egy mezőt amire lerak egy tornyot.
--------------	--

Use-case neve	Toronyra kristály vétele
Rövid leírás	Torony fejlesztése kristállyal.
Aktorok	Controller, Tower
Forgatókönyv	A felhasználó kiválaszt egy mezőt, és ha van rajta torony fejleszti egy kiválasztott kristállal.

Use-case neve	Torony eladása
Rövid leírás	Torony eladása.
Aktorok	Controller, Tower
Forgatókönyv	A felhasználó kiválaszt egy mezőt, és a rajta lévő tornyot eladja.

Use-case neve	Toronyra köd ereszkedik
Rövid leírás	Toronyra köd ereszkedik, ami a hatósugarát lecsökkenti.
Aktorok	Tower, Game
Forgatókönyv	A Game kiválaszt egy tornyot, amire ködöt rak. Lecsökken a hatósugara a toronynak, ezért frissíti az út cellák listáját. (setPath)

Use-case neve	Torony lő
Rövid leírás	Egy, a torony a hatósugarában lévő ellenségre tüzel az ellenség.
Aktorok	Tower, Enemy
Forgatókönyv	A torony lekér a hatósugarában lévő mezők közül egy ellenséget, amire kilövi a bullet-jét.

Use-case neve	Akadály megvétele
Rövid leírás	Egy üres útra akadályt vesz a játékos.
Aktorok	Controller, Obstacle
Forgatókönyv	Először a Controller setPath metódusával kiválaszt a felhasználó egy mezőt amire lerak egy tornyot.

Use-case neve	Akadály lassít
Rövid leírás	A lerakott akadály lassítja a rajta áthaladó ellenséget.
Aktorok	Path
Forgatókönyv	Amikor egy ellenség egy akadállyal terhelt cellára lép, akkor a Path beállítja az enemy modSpeed-jét.

Use-case neve	Akadály elromlik
Rövid leírás	Akadály elromlik
Aktorok	Path
Forgatókönyv	A path törli az akadályt.

Use-case neve	Akadály eladás
Rövid leírás	A játékos elad egy akadályt.
Aktorok	Controller
Forgatókönyv	A felhasználó kiválaszt egy utat, és a rajta lévő akadályt eladja.

7.3. Tesztelési terv

Teszt-eset neve	Ellenség hullám indítása
Rövid leírás	A Game létrehoz 3 ellenséget, a játékos vesz 2 tornyot, és minden fejleszti 1-1 kristállyal. Ez után utukra indítja az ellenségeket. Végén elad a játékos egy tornyot.
Teszt célja	Ez a teszeset az ellenség hullám létrehozását, tornyok vételeit, fejlesztését, tüzelését és eladását teszteli. Ebben a teszesetben fogunk látni ellenség halálát is. A játékos nyer.

Teszt-eset neve	Akadály és ellenség bejut a végzet hegyéhez
Rövid leírás	A pályán elhelyez a játékos egy akadályt, amit fejleszt egy intenzitás növelő kristállyal, majd bejut az ellenség a végzet hegyéhez. Amikor az utolsó ellenség is áthaladt rajta elhasználódik.
Teszt célja	A teszeset akadály vételt, fejlesztést, működést és elhasználódást, teszteli, illetve a végzet hegyéhez való bejutást. A játékos veszít.

Teszt-eset neve	Köd és ellenség kettészelés
Rövid leírás	A pályára feltesz a Game 3 ellenséget, a játékos vesz 3 tornyot, amikor beér az első ellenség a torony hatósugarába, köd száll a tornyokra, és kettőbe vágó lövedékeket lőnek onnantól. Miközben végigérnek az ellenségek a pályán, meghalnak.
Teszt célja	Ez a teszeset a tornyokra ereszkező ködöt teszteli és a kettészelő lövedékeket.

Teszt-eset neve	Ellenség elágazáshoz ér
Rövid leírás	A pályára feltesz a Game 3 ellenséget. A pálya kialakításából addódóan 1 lépés után elágazáshoz érnek.
Teszt célja	A teszeset azt teszteli elágazásoknál merre mennek tovább az ellenségek. Ha a véletlenszerűség be van kapcsolva a programban, akkor egy véletlen utat választ, ha nem akkor a 0. elemét adja vissza az utakat tároló tömbnek.

7.4. Tesztelést támogató segéd- és fordítóprogramok specifikálása

A prototípustól adott bemenet esetén elvárt, illetve a kapott eredmények összehasonlítását saját programmal végezzük. Az összehasonlító program parancssorból futtatható, két paramétert vár: a várt és a kapott eredményeket tartalmazó fájlok neveit. A program lefutása után megmondja, hogy a két fájl megegyezik-e, és ha nem, akkor kiírja az eltérő sorok sorszámát.

A program futtatása a következő parancs kiadásával történik, a lefordított .class file mellett:

```
java TxtComparer <elso_file_neve> <masodik_file_neve>
```

7.5. Napló

Kezdet	Időtartam	Résznevők	Leírás
2014.03.26. 08:15	1,5 óra	Elekes Fuksz Seres Nagy Rédey	Konzultáció
2014.03.27. 15:00	0,5 óra	Elekes	Torony use-casek
2014.03.27. 17:00	1 óra	Rédey	7.0.1 pont dokumentálása, 7.0.2 osztálydiagram módosítása
2014.03.28. 18:00	1 óra	Seres	Módosításhoz tartozó szekvencia diagramok elkészítése
2014.03.29. 14:00	2 óra	Elekes Fuksz Seres Nagy Rédey	Skype konferencia bemeneti/kimeneti nyelvkről, pálya térképről
2014.03.30. 17:00	1 óra	Elekes	Kimeneti nyelv specifikálása, teszt-esetek megírása.
2014.03.28. 19:00	1 óra	Nagy	Enemy és Obstacle use-case-ek megírása
2014.03.30. 15:30	1,5 óra	Nagy	Bemeneti nyelv specifikálása
2014.03.30. 22:00	10 perc	Seres	7.4-es pont kitöltése
2014.03.31. 00:30	2,5 óra	Fuksz	Dokumentáció véglegesítése

8. Részletes tervezetek

8.1. Osztályok és metódusok tervezetek

8.1.1. Bullet

- Felelősség

A tornyok egy Bullet-et tárolnak, amit minden lövésnél átadnak a lövő függvénynek. Ennek a lövedéknek a feladata, hogy az ellenségnak megmondja mennyit sebez rajta.

- Ősosztályok

Object

- Interfészek

Nincs

- Attribútumok

-damage: int Az alapsebzés értéke. Ezt adják vissza a getDamage függvények, ha nincs fejlesztve az adott típusú Enemy-re a torony.

-enemyType: String A torony itt tárolja, hogy melyik ellenség típusra erősebb a sebzése.

- Metódusok

+Bullet(dmg: int, et: String) Konstruktor

+getHobbitDamage(): int Ha hobbitot sebez, ezzel a függvénykel kérdezi le a sebzés értékét.

+getHumanDamage(): int Ha embert sebez, ezzel a függvénykel kérdezi le a sebzés értékét.

+getDwarfDamage(): int Ha törpöt sebez, ezzel a függvénykel kérdezi le a sebzés értékét.

+getElfDamage(): int Ha tündét sebez, ezzel a függvénykel kérdezi le a sebzés értékét.

+setEnemy(e: String): void Beállítja az ellenséget akire specializált (enemyType).

+setDamage(dmg: int): void Beállítja a lövedék sebzését (damage).

8.1.2. Enemy

- Felelősség

Az Enemy osztály felelőssége, hogy egy adott lövedék (Bullet) hatására sebződjön, vagy ha már sokat sebződött, akkor haljon meg, valamint az, hogy celláról cellára mozgassa magát, és ha eléri a végzet helyét értesítse a Game osztályt. Tudja, hogy 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

-speed: int A két lépés között eltelt idő. static, final

- maxHp: int** A maximális életerő értéke. static, final
- modSpeed: int** Az ellenség belső idő mérője. A setModSpeed változtathatja – jellemzően negatív irányba, akadályokon.
- nextPath: Path** A soron következő path címe.
- #health: int** Életerejét tárolja ebben. Hurt függvényben csökkenti.
- #myPath: Path** Az a mező, ahol tartózkodik.
- #igame: IGame** Ezen keresztül tudja módosítani a manát, amikor meghal, illetve ha elér a végzethegyére módosítani a számlálót (Game.succeededE), hogy nőjön egyel.

- Metódusok

- +hurt(b: Bullet): void** Sebződést megvalósító metódus, abstract, minden Enemy-típusban máshogym implementálódik.
- +move(): void** mozog, a következő path-ra lép, cellát vált.
- +Enemy(ig: IGame)** konstruktur.
- +setModSpeed(msp: int): void** A modSpeed változót változtatja. Lassítani lehet vele.
- +setHealth(h: int): void** A health változó setttere.
- +void cut(): void** A kettévágásos lövésért felelős metódus. Létrehoz egy új, azonos típusú ellenséget (abstract), feleakkora életerővel. Beteszí az igam enemiesIn tömbjébe. Saját életerjét is felezi.

8.1.3. 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(b: Bullet): void** Sebződik, a kapott Bullet alapján getDamage... metódus által visszaadott értékkel csökkenti az életerejét. Ha elfogyott az életerere törli magát a pathról és az igame.enemiesIn-ből Ha 0-t kap, cut-ot hív.

8.1.4. Game

- Felelősség

A Game osztály felelőssége többek között a játék ütemezése, az idő műlásának 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. A game osztály hozzá létre az ellenségeket és indítja el őket az úton.

- Ősosztályok
Object

- Interfészok
IGame
- Attribútumok

-map: Map játék térképe

-enemiesOut: List<Enemy> A pályára még be nem lépett ellenségek. Ezt töltjük meg újabb ellenség hullám generálásakor, majd fokozatosan kiürítjük és egyesével átírjuk a tartalmat az enemiesIn-be.

-enemiesIn: List<Enemy> A pályára már belépett ellenségek. minden update hívásnál módosul(hat) a tartalom, pl. új Enemy lép a pályára vagy meghal egy-pár.

-towers: List<Tower> A tornyok, amik a pályán vannak. Update-enként végigmegyünk rajtuk és meghívjuk a shoot.

-mana: int A maradék varázserő.

-firstP: Path Az út kezdő cellája. Ennek segítségével indítjuk el az Enemy-ket a pályán, megadva nekik a címet.

-noEnemies: int Kezdeti hullámérték, amely folyamatosan nő, azt mutatja meg, hogy következő körben hány ellenséget kell létrehozni és beküldeni a pályára. Ebből megtudhatjuk azt is, hogyha a játékos teljesítette a pályát.

-succeededE: int A végzet hegyét elérte enemy-k száma. Ha egy Enemy célba ér, az IGame-n keresztül tudja növelni. Ha elér egy maximum értéket, vége a játéknak.

-hazeTime: int Az az idő, ameddig a köd tart. Ha lejár a köd, nullázzuk, ameddig a ködnek tartania kell, növeljük.

- Metódusok

+update(): void Frissíti a modellt, grafikát. X időközönként folyamatosan hívjuk. Belül az enemiesIn-en végighaladva minden elemen move-ot hívunk, towers-en ugyanígy shoot-ot, illetve ha kell, enemiesOut-ból betessük a következő elemet. Random módon a towers elemein haze-t is hív.

+initialize(String name): void Kiinduló állapot felvétele.

+Game() Konstruktor.

+makeEnemies(): void Létrehoz noEnemies db ellenséget, random típusú, ezeket betesz az enemiesOut-ba.

+getMap(): Map map getter.

8.1.5. Controller

- Felelősség

A felhasználó és a játék közötti kommunikációnak véghezvitele a felelőssége. Kristályok, tornyok, akádályok vásárlását lehet rajta keresztül megcsinálni. Ellenőriznie kell, hogy van-e a játékosnak elég várazsereje a tranzakcióhoz.

- Ősosztályok

Object

- Interfészek

Nincs

- Attribútumok

- igame: IGame** Szükséges függvények elérésére szolgál, hogy tudja módosítani a Game-et.
 - chosenField: Field** Ha Field-et választ ki, ebben a változóban tárolja.
 - chosenPath: Path** Ha Path-t választ ki, ebben a változóban tárolja.
 - chosenEnemy: String** A kiválasztott EnemyType.
- Metódusok
- +buyTower(): void** Torony vételére szolgál. Létrehoz egy tornyot, beregisztrálja a chosenField-re.
- +buyObstacle(): void** Lásd buyTower Obstacle-el és chosenPath-el.
- +buySpeed/Range/Damage/EnemyType/Intensiyty/RepairGem(): void** Kristályok vásárlása. A Towert/Obstacle-t amire vettük a chosenField/Path-en érjük el.
- +setField(f: Field): void** A chosenField-et állítja.
- +setPath(p: Path): void** A chosenPath-t állítja.
- +setEnemy(e: String): void** A chosenEnemy-t állítja.

8.1.6. IGame

- Felelősség

Az IGame interfész szolgáltatást nyújt az akadályoknak, tornyoknak, ellenségeknek, hogy rajta keresztül manát írjanak jóvá/csökkentsenek, illetve ellenségek esetén a végzet hegyét elérte ellenségek számát módosítsák. Speciális interfész a Game osztályhoz.
- Metódusok

- +changeMana(): void** A manát megváltoztató metódus.
- +incSucceeded(): void** A succeededE értékét megváltoztató metódus.
- +addTower(t: Tower): void** A hozzáad egy tornyot a towers listához.
- +removeEnemyIn(e: Enemy): void** Az ellenség törlése enemiesIn-ból.
- +removeEnemyOut(e: Enemy): void** Az ellenség törlése enemiesOut-ból.
- +removeTower(t: Tower): void** A torony törlése towers-ból.
- +addEnemyIn(e: Enemy): void** Ellenség hozzáadása enemiesIn-hez.
- +getMana(): int** A mana lekérdezése.

8.1.7. Gem

- Felelősség

A Gem osztály felel a torony fejlesztéséért. Ha a játékos vesz a toronyra/akadályra valamelyen kristályt, akkor jön létre, megkapja a torony, és beépíti magába.
- Ősosztályok
 - Object
- Interfészek
 - Nincs
- Attribútumok

+price: int Az ár, amennyi varázserőbe kerül. static, final

- Metódusok

+Gem(price: int) Konstruktor.

8.1.8. IObstacle

- Felelősség

Olyan metódusok használatát teszi lehetővé, amelyek az Obstacle típusú elemek viselkedését modellezik

- Ősosztályok

Nincs

- Metódusok

+slow(intesity: int, p: Path): void Szól p-nek, hogy lassítsa le az ellenséget intensity-vel.

+amortization(): void Amortizál, csökkenti amort értékét.

+increaseIntensity(intens: int): void Megnöveli az intensitet intens-el.

+addIOGem(IOGem: iog): void Az iog kristályt hozzáadja az akadályhoz.

+repair(): void Megjavítja az akadályt, amort értékét maximalizálja.

8.1.9. Obstacle

- Felelősség

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

- Ősosztályok

Nincs

- Interfészek

IObsacle, IPathPlaceable

- Attribútumok

-slowIntens: int A lassítás mértéke.

-myPath: Path A path, amin rajta van.

-amort: int Az elhasználódottság mértéke.

+price: int Az ára. static, final

-gems: ArrayList<IOGem> A megvett kristályok listája.

- Metódusok

+Obstacle() Konstruktor.

8.1.10. ITower

- Felelősség

A torony funkciói vannak benne. Egy olyan interfész, amin keresztül a tornyok kezelhetők.

- Metódusok

+setPaths(): void A myField-ből kiindulva a range-el lefedett területen felkeresi, és beregisztrálja a paths listába a path cellákat.

+shoot(): void A torony akkor lő, ha letelt az újratöltési idő, ekkor megnézi, hogy lőtávon belül van-e ellenség, és ha van meghívja a hurt függvényét, átadva paraméterként a bullet-et.

+addITGem(gem: ITGem): void Paraméterként megkapja a kiválasztott kristályt, bulletet frissíti, ha kell, illetve a torony adott attribútumait.

+chooseEnemy(): Enemy A torony tárolja a hatókörbe eső path cellákat. minden tick-ben végig megy rajtuk, és kiválaszt egyet, amelyiken van ellenség, és oda fog lőni. Az ellenséggel tér vissza.

+haze(): void Kódöt generál a tornyon mod range beállításával.

+clearUp(): void Beállítja a modRange-et normál, kód nélküli értékre range alapján.

+getEnemyType(): String Lekérdezi és visszaadja a Bulletben tárolt enemyType-ot.

sell(): void Torony eladása.

8.1.11. Tower

- Felelősség

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 tornyokat, illetve el is adhatja őket. Ez a legfontosabb eszköz amivel a játékos meg tudja akadályozni az ellenségek célba jutását.

- Ősosztályok

Nincs

- Interfészek

ITower, IFieldPlaceable

- Attribútumok

+price: int Az ára varázserőben. static, final

-range: int Lőtáv, hatókör.

-modRange: int Módosított hatókör.

-speedCtr: int A torony belső idő mérője. Ezt vizsgálja minden lövés előtt, hogy eltelt e elég idő.

-speed: int két lövés között eltelt minimális idő.

-bullet: Bullet A torony tárol egy lövedéket, mindig ezt lövi ki.

-gems: ArrayList<ITGem> A megvásárolt kristályokat tárolja.

-paths: ArrayList<Path> Hatósugárba eső út cellák.

-myField: Field Mező, amin áll.

-igame: IGame Egy interfész a játék logikára, amivel a bejutott ellenségek számát, és a varázserőt is lehet állítani.

- Metódusok

+Tower(igame: IGame) Konstruktur.

+upgradeSpeed(sp: int): void Fejleszti a lövési sebességét.

+upgradeRange(rng: int): void Fejleszti a lőtávot.

+upgradeEnemy(e: String): void Egy ellenségtípusra növeli a sebzést.

+upgradeDamage(dmg: int): void Növeli a sebzést.

8.1.12. Map

- Felelősség

A Map osztály a játéktér elemeit, mint cellák tárolja, egy két dimenziós tömbben. Megadja minden egyes cellához, a szomszédjai referenciáját. Nincs

- Interfészek

Nincs

- Attribútumok

-name: String A pálya neve, egyben az azonosítója.

-level: int A pálya szintje.

-grid: Array<Array<Cell>> A cellákat tartalmazó 2 dimenziós tömb.

-igame: IGame Interfész Game eléréséhez.

- Metódusok

+Map(igame: IGame) Az osztály konstruktora.

+load(name: String) Megnyitja a paraméterként kapott nevű fájlt, és abból betölti a pálya celláinak tulajdonságait, felépíti a pályát.

+getLevel(): int Visszaadja a pálya szintjét.

+getName(): String Visszaadja a pálya nevét.

+getFirstPath(): Path Visszaadja a pálya belépési pontjának referenciáját.

+getCell(i: int, j: int): Cell Adott cella lekérése.

+getGrid(): ArrayList<ArrayList<Cell>> : Játéktér lekérése.

+isLoaded(): bool : Megadja, hogy be van-e töltve.

+getHeight(): int : Magasságot adja vissza.

+getWidth(): int : Szélességet adja vissza.

+getSize(): Dimension : A pálya méreteit adja vissza.

8.1.13. 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

-neighbours: Array<Cell> : 4 elemű tömb, tárolja 4 irányban a szomszédjai referenciáját.

- Metódusok

+Cell(+cells: Array<Cell>, igame: IGame) Konstruktor, paraméterként kapja a szomszédos mezők referenciáit.

+isPath(): bool Olyan értékkel tér vissza amilyen típusú a cella (true/false megfelelés).

+getNeighbours(): Array<Cell> Getter a neighbours-höz.

setNeighbours(neig: Array<Cell>) Setter a neighbours-höz.

8.1.14. Field

- Felelősség

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

- Ősosztályok

Object → Cell

- Interfészek

Nincs

- Attribútumok

-itower: ITower A mezőn álló torony interfészű elem tárolása.

-igame: IGame : Referencia Game interfészéhez.

- Metódusok

+registerFieldPlaceable(ifp: IFieldPlaceable): void Egy új tornyot ad hozzá a mezőhöz, ifp-en meghívja a registerField-et önmagával, ami a registerITowert hívja.

+deleteFieldPlaceable(ifield: IFieldPlaceable): void Eltávolítja a tornyot a mezőről.

registerITower(itower: ITower): void Betesz a fieldbe a kapott tornyot.

+hasTower(): bool Megadja, hogy szabad-e a mező tower elhelyezéséhez.

+getITower(): ITower Visszaadja az itowert.

+Field(igame: IGame) Konstruktor.

8.1.15. Path

- Felelősség

A Path a Cell osztály leszármazottja. Az út típusú cellákat reprezentálja. Tartalmazza a rajta lévő ellen-ségeket és esetleg akadályt. minden út tudja azt is, hogy hova lehet lépni róla egy lépésben.

- Ősosztályok

Object → Cell

- Interfészek

Nincs

- Attribútumok

-obstacle: IObstacle Az esetleg az úton levő akadályt tárolja.

-enemies: ArrayList<Enemy> Az éppen áthaladó ellenségek listája.

-paths: ArrayList<Path> Következő path-ok címei.

- Metódusok

+isPath(): bool Igaz értékkel tér vissza.

+deleteIPathPlaceable(ipath: IPathPlaceable): void Kitörli a tárolójából a paraméterként kapott referenciával megegyező tárolt referenciát.

+deleteEnemy(e: Enemy): bool Kitörli enemies-ből a kapott e-t.

+deleteIObstacle(io: IObstacle): bool Akadály törlése az útról.

+registerIPathPlaceable(ipath: IPathPlaceable): void Beregisztrálja a paraméterként kapott ob-jektumot, mint saját magán tartózkodó ellenség vagy akadály. RegisterPath-t hív az ipath-on, ami a rá jellemző regisztert hívja vissza.

+hasEnemy(): bool Megmutatja, hogy van-e a cellán ellenség.

+registerEnemy(e: Enemy): void A kapott ellenséget beteszi az enemies-be.

+registerIObstacle(io: IObstacle): void A kapott akadály lesz az obstacle.

+getEnemies(): ArrayList<Enemy> Visszatér az enemies-el.

getIObstacle(): IObstacle Visszatér myIObstacle-el.

+getNext(): Path Paths-ből ad vissza egy random elemet.

+hasObstacle(): bool Megadja, hogy van-e rajta Obstacle.

+setNextPaths(np: ArrayList<Path>): void Következő utak címeinek beállítása, nextPaths setter.

8.1.16. Towerhez tartozó krsítályok: Range, Speed, Damage, EnemyType

- Felelősség

A torony lövésének hatósugarát, gyorsítását, sebzését, ellenféltípusra sebzés, növelő kristályok osztályai.

- Ősosztályok

Object → Gem

- Interfészek

ITGem

- Attribútumok

-range/ speed/ damage/ eType: int/ int/ int/ String A kristálynak megfelelő attribútum, amin fejleszt.

- Metódusok

Konstruktörök

8.1.17. Obstaclehez tartozó kristályok: Intensity, Repair

- Felelősség

Akadályra helyezhető kristályok, ami növeli a lassítás értékét vagy megjavítja az akadályt.

- Ősosztályok
Object → Gem

- Interfészek
IOGem

- Attribútumok

-intensity: int Az intensityGem-hez tartozik, intensity értékét állítja az obstacle-ben. A repairGem-nek nincs attribútuma.

- Metódusok

Konstruktörök

8.1.18. IOGem

- Felelősség

Akadályra helyezhető kristályok interfésze.

- Metódusok

+upgradeObstacle(o: Obstacle): void A kapott akadályt fejleszti. Meghívja a kristályra jellemző, obstacle-ben lévő fejlesztő függvényt.

8.1.19. ITGem

- Felelősség

Toronyra illeszthető kristályok interfésze.

- Metódusok

+upgradeTower(t: Tower): void Fejleszti a kapott t tornyot magával. Meghívja a rá jellemző upgrade... tower metódust.

+getValue(): int Visszaad egy, a torony árával képzett értéket, a torony eladásakor jóváírandó mana érték kiszámításához.

8.1.20. IFieldPlaceable

- Felelősség

Interfész a mezőre helyezhető osztályok számára. Azonosítja azokat az objektumokat, amelyeket csak a mező típusú pályaelem tartalmazhat.

- Metódusok

+registerField(field: Field): void A mezőre helyezhető objektumnak megadja paraméterben annak a mezőnek a referenciaját, amelyikre helyezve lesz.

8.1.21. IPathPlaceable

- Felelősség

Interfész az útra helyezhető osztályok számára. Azonosítja azokat az objektumokat, amelyeket csak az út típusú pályaelem tartalmazhat.

- Metódusok

+eliminate(p: Path): void Az útra helyezhető objektum eltávolítása az útról (p-ről).

+registerPath(p: Path): void Az útra helyezhető objektumnak megadja paraméterben annak az útnak a referenciaját, amelyikre helyezve lesz.

8.2. A tesztek részletes tervei, leírásuk a teszt nyelvén

8.2.1. Ellenség mozgás és sebzés

- Leírás

A tesztelő létrehoz 2 ellenséget és vesz 2 tornyot, egyiket a 3 kristállyal, a másikat Elf kristállyal. Ez után utukra indítja az ellenségeket, akik útközben meghalnak. Végén elad a játékos egy tornyot.

- Ellenőrzött funkcionális, várható hibahelyek

Ellenségek létrehozása, halála. Tornyok vétele, fejlesztése, tüzelés és eladást tesztel.

Várható hibahely: ellenség mozgatása, kristály vétel

- Tesztesethez használt térkép

testmap1.txt					
x	x	x	x	x	x
en	r	r	r	r	ex
x	x	x	x	x	x

- Bemenet

loadmap testmap1.txt
buy tower 2-1
buy tower 2-4
buy gem 2-1 speed
buy gem 2-1 range
buy gem 2-1 damage
buy gem 2-4 enemy elf
make hobbit 1-1

```

update 2
make elf 1-1
update 2
sell 2-1
exit

```

- Elvárt kimenet

```

map testmap1.txt loaded
Tower0 created
Tower1 created
SpeedGem0 created
RangeGem0 created
DamageGem0 created
EnemyTypeGem0 created
Hobbit0 created
update
Hobbit0 moves
Tower0 shoots Hobbit0
Hobbit0 damage 40
update
Hobbit0 moves
Tower0 shoots Hobbit0
Hobbit0 damage 40
Tower1 shoots Hobbit0
Hobbit0 damage 20
Hobbit0 died
Elf0 created
update
Elf0 moves
Tower0 shoots Elf0
Elf0 damage 40
update
Elf0 moves
Tower0 shoots Elf0
Elf0 damage 40
Tower1 shoots Elf0
Elf0 damage 60
Elf0 died
Tower0 sold for 80 mana
VEGE

```

8.2.2. Akadály teszt

- Leírás

A pályán elhelyez a játékos egy akadályt, amit fejleszt egy intenzitás növelő kristállyal, majd bejut az ellenség a végzet hegyéhez. Amikor az utolsó ellenség is áthaladt rajta elhasználódik.

- Ellenőrzött funkcionálitás, várható hibahelyek

Ellenségek létrehozása, akadály létrehozása, fejlesztése, elhasználódása. Végzet hegyére jutás.

Várható hibahely: akadály elhasználódása, ellenség bejutása a Végzet Hegyéhez

- Teszesethöz használt térkép

testmap2.txt:				
x	x	x	x	x

en	r	r	ex
x	x	x	x

- Bemenet

```
loadmap testmap2.txt
buy obstacle 1-1
buy gem 1-1 intensity
make hobbit 1-0
update 14
exit
```

- Elvárt kimenet

```
map testmap2.txt loaded
Obstacle0 created
IntensityGem0 created
Hobbit0 created
update
Hobbit0 moves
Obstacle0 slows Hobbit0 by 10
Obstacle0 worn out
update
Hobbit0 moves
update
Hobbit0 moves
update
Hobbit0 moves
Hobbit0 reached Mount Doom
VEGE
```

8.2.3. Kód, és ellenség kettészelés

- Leírás

A pályára feltesz a tesztelő egy Human ellenséget és egy tornyot. Frissíti a játéket, a torony lő az ellen-ségre. Beállítja a ködöt, és hogy kettészelő lövedékeket lőjenek a tornyok. Kétszer frissíti a pályát, elsőre csak lép, mert kikerült a torony hatósugarából, másodiknak újra bekerült, kettészelő lövedékkel lövi meg a toronyt.

- Ellenőrzött funkcionalitás, várható hibahelyek
Kód, kettélövő lövedék.

Várható hibahely: hatósugár újrakalibrálása, ellenség kettészelése

- Teszesethöz használt térkép

```
testmap3.txt:
```

x	x	x	x	x
en	r	r	r	ex
x	x	x	x	x

- Bemenet

```
loadmap testmap3.txt
make human 1-0
buy tower 2-4
buy gem 2-4 range
update 1
haze on
cut on
update 2
exit
```

- Elvárt kimenet

```
map testmap3.txt loaded
Human0 created
Tower0 created
RangeGem0 created
update
Human0 moves
Tower0 shoots Human0
Human0 damage 20
Haze is turned on
Cut is turned on
update
Human0 moves
update
Human0 moves
Tower0 shoots Human0
Human0 damage 0
Human0 cut in half
Human1 created
VEGE
```

8.2.4. Ellenség elágazáshoz ér

- Leírás

A pályára feltesz a tesztelő 3 különböző ellenséget. A pálya kialakításából adódóan 1 lépés után elágazáshoz érnak. Mindegyiket különböző irányba küldjük.

- Ellenőrzött funkcionálitás, várható hibahelyek

Ellenségek elágazás kezelése.

Várható hibahely: elágazás

- Tesztesethez használt térkép

```
testmap4.txt:
```

x	x	x	x	x	x
x	x	r	r	d	x
en	r	rud	r	r	ex
x	x	r	r	u	x
x	x	x	x	x	x

- Bemenet

```
loadmap testmap4.txt
make elf 2-1
make human 2-1
make dwarf 2-1
update 1
route Elf0 1
route Human0 0
route Dwarf0 2
update 1
drawMap
exit
```

- Elvárt kimenet

```
map testmap4.txt loaded
Elf0 created
Human0 created
Dwarf0 created
update
Elf0 moves
Human0 moves
Dwarf0 moves
update
Elf0 moves
Elf0 crossroad
Human0 moves
Human0 crossroad
Dwarf0 moves
Dwarf0 crossroad
/-----\
|xx|xx|xx|xx|xx|xx|
|xx|xx|xx|xx|xx|xx|
|---+---+---+---+---+
|xx|xx|e | | |xx|
|xx|xx| | | |xx|
|---+---+---+---+---+
| | | | e | | |
| | | | | | |
|---+---+---+---+---+
|xx|xx|e | | |xx|
|xx|xx| | | |xx|
|---+---+---+---+---+
|xx|xx|xx|xx|xx|xx|
|xx|xx|xx|xx|xx|xx|
\-----/
VEGE
```

8.3. A tesztelést támogató programok tervei

Készülnek a prototípus program, illetve a szövegfájl összehasonlító segédprogram fordítását és futtatását megkönnyítő batch fájlok, melyek futtatásuk után automatizáltan elvégzik ezeket a feladatokat.

Az előre elkészített, és a 8.2-es pontban leírt tesztesetek amellett, hogy futtathatóak a parancssorból a be-menet átirányításával a megfelelő fájlból (pl.: java ProtoTester < teszt1.dat), reprodukálhatjuk őket az erre a célra készített batch fájl segítségével is: test.bat. A batch fájl működése a következő: futtatás után ellenőrzi, hogy a szükséges programok (prototípus, illetve szöveges fájl összehasonlító) rendelkezésre állnak-e a megfelelő helyen, ha nem, akkor felkéri a felhasználót azoknak a lefordítására, vagy áthelyezésére, majd kilép. Amennyiben rendelkezésre állnak, megkérdezi a tesztelőtől, hogy melyik tesztesetet kívánja lefuttatni, illetve a kimenetet megjelenítse-e a konzolablakban. Amennyiben a megjelenítés mellett dönt a tesztelő, az adott teszteset lefuttatása után keletkező kimenet a konzolban jelenik meg, és billentyűlenyomásig ott is marad, így az elemezhető, az elvárt kimenettel összehozható. Ellenkező esetben, tehát ha a tesztelő úgy dönt, hogy ne jelenjen meg a prototípus kimenete, akkor egy fájlból irányítja azt a batch fájl, és az így keletkező fájlt összeveti az elvárt kimenettel a TxtComparer segédprogram segítségével (ennek specifikációja lentebb található), és megmondja, hogy egyezik-e, majd billentyűlenyomás után kilép.

A tesztelést tovább segíti a TxtComparer nevű, szöveges fájlok összehasonlítását végző program. A program Java nyelven készül, forráskódja a prototípushoz hasonlóan adott. A program a fordítás után a parancssorból futtatható, ahol két paramétert vár: a két összehasonlítandó fájl nevét (pl.: java TxtComparer file1 file2). Futtatás után, amennyiben a helyes paraméterek rendelkezésre állnak, illetve elérhetők a megadott fájlok, a program soronként összehasonlítja a két fájlt, és amennyiben azok megegyeznek, értesíti erről a felhasználót. Azon esetben, amikor a fájlok tartalmában eltérést észlel, az eltérést tartalmazó sorok számai jelennek meg a kimeneten.

8.4. Napló

Kezdet	Időtartam	Résznevők	Leírás
2014.04.03. 15:00	5 óra	Rédey	8.1 pont
2014.04.05.	3 óra	Elekes Seres	Értekezlet. 8.2 pont megírása
2014.04.05.	45 perc	Elekes	Tevékenység: Teszteset megírása. Kimeneti nyelv leírása.
2014.04.05. 21:40	1 óra	Seres	8.3 pont megírása
2014.04.07. 00:30	1,5 óra	Fuksz	Tesztesetek pályáinak megvalósítása. Dokumentáció véglegesítése.

10. Prototípus beadása

10.1. Fordítási és futtatási útmutató

10.1.1. Fájllista

Fájl neve	Méret	Keletkezés ideje	Tartalom
proto/Application.java	407	2014.03.11.	Névvkel megegyező osztály.
proto/Bullet.java	2176	2014.03.11.	Névvkel megegyező osztály.
proto/Cell.java	677	2014.03.11.	Névvkel megegyező osztály.
proto/Controller.java	4601	2014.03.11.	Névvkel megegyező osztály.
proto/DamageGem.java	459	2014.03.11.	Névvkel megegyező osztály.
proto/Dwarf.java	1325	2014.03.11.	Névvkel megegyező osztály.
proto/Elf.java	1310	2014.03.11.	Névvkel megegyező osztály.
proto/Enemy.java	4436	2014.03.11.	Névvkel megegyező osztály.
proto/EnemyTypeGem.java	504	2014.03.11.	Névvkel megegyező osztály.
proto/Field.java	1222	2014.03.11.	Névvkel megegyező osztály.
proto/Game.java	5368	2014.03.11.	Névvkel megegyező osztály.
proto/Gem.java	274	2014.03.11.	Névvkel megegyező osztály.
proto/Hobbit.java	1330	2014.03.11.	Névvkel megegyező osztály.
proto/Human.java	1323	2014.03.11.	Névvkel megegyező osztály.
proto/IFieldPlaceable.java	127	2014.03.11.	Névvkel megegyező osztály.
proto/IGame.java	470	2014.03.11.	Névvkel megegyező osztály.
proto/IntensityGem.java	460	2014.03.11.	Névvkel megegyező osztály.
proto/IObstacle.java	216	2014.03.11.	Névvkel megegyező osztály.
proto/IOGem.java	96	2014.03.11.	Névvkel megegyező osztály.
proto/IPathPlaceable.java	129	2014.03.11.	Névvkel megegyező osztály.
proto/ITGem.java	115	2014.03.11.	Névvkel megegyező osztály.
proto/ITower.java	235	2014.03.11.	Névvkel megegyező osztály.
proto/Map.java	4763	2014.03.11.	Névvkel megegyező osztály.
proto/Obstacle.java	1815	2014.03.11.	Névvkel megegyező osztály.
proto/Path.java	4517	2014.03.11.	Névvkel megegyező osztály.
proto/ProtoTester.java	17002	2014.03.11.	A proto programot tesztelhetővé tevő program.
proto/RangeGem.java	420	2014.03.11.	Névvkel megegyező osztály.
proto/RepairGem.java	297	2014.03.11.	Névvkel megegyező osztály.
proto/SpeedGem.java	461	2014.03.11.	Névvkel megegyező osztály.
proto/Tower.java	4416	2014.03.11.	Névvkel megegyező osztály.
txtComparer/TxtComparer.java	1929	2014.04.20.	A kapott kimenetet az elvárt kimenettel összehasonlító program.
test1.expected	545	2014.04.20.	Teszeset1 elvárt kimenete
test2.expected	341	2014.04.20.	Teszeset2 elvárt kimenete
test3.expected	300	2014.04.20.	Teszeset3 elvárt kimenete
test4.expected	561	2014.04.20.	Teszeset4 elvárt kimenete
test1.txt	200	2014.04.09.	Teszeset1 bemenete
test2.txt	97	2014.04.09.	Teszeset2 bemenete
test3.txt	115	2014.04.09.	Teszeset3 bemenete

Fájl neve	Méret	Keletkezés ideje	Tartalom
test4.txt	147	2014.04.09.	Teszeset4 bemenete
testmap1.txt	39	2014.04.07.	Teszeset1 térképe
testmap2.txt	27	2014.04.07.	Teszeset2 térképe
testmap3.txt	33	2014.04.07.	Teszeset3 térképe
testmap4.txt	67	2014.04.07.	Teszeset4 térképe
ProtoCompileAndRun.bat	288	2014.04.21.	Parancssoros fordítást végzi.
ProtoTestAndCompare1.bat	588	2014.04.20.	Teszeset1 futtatása és ellenőrzése
ProtoTestAndCompare2.bat	588	2014.04.20.	Teszeset2 futtatása és ellenőrzése
ProtoTestAndCompare3.bat	588	2014.04.20.	Teszeset3 futtatása és ellenőrzése
ProtoTestAndCompare4.bat	588	2014.04.20.	Teszeset4 futtatása és ellenőrzése

10.1.2. Fordítás

A fordítás a "javac" programmal történik, ezért olyan környezetben lehetséges csak, ahol ez a program elérhető. A tesztelés megkönnyítése végett a fordítás és futtatás automatizálására több batch fájl áll rendelkezésre. Ezeknek a használata a 10.1.3 pontban van részletezve.

A fordítás parancssorból is lehetséges a következő, a project gyökerében kiadott parancssal:

```
javac -d ./bin/ proto/*.java
```

Ez a parancs lefordítja a prototípust, és a lefordított állományokat a bin mappába helyezi.

A szövegfájlok összehasonlítására használt segédprogram (TxtComparer) fordításához a projekt gyökerében adjuk ki a következő parancsot:

```
javac -d ./bin/ txtComparer/TxtComparer.java
```

10.1.3. Futtatás

A futtatás a "java" programmal történik, ezért olyan környezetben lehetséges csak, ahol ez a program elérhető. Az adott batch fájlok segítségével lehetséges lefordítani és futtatni a prototípust: ehhez a ProtoCompileAndRun.bat-t futtassuk, ami a fordítás után azonnal el is indítja a prototípust.

Ha a tesztelést az előre megírt teszeseteken keresztül végezzük, akkor a batch fájlokkal ez is automatizálva van: ProtoTestAndCompare(szám).bat fájlok, amelyek a prototípust lefordítják, az adott teszt bemenetével lefuttatják, és a kimenetet egy fájlba irányítják, amit aztán a szövegfájlok összehasonlítására használt segédprogram (TxtComparer) segítségével összehasonlítanak az elvárt bemenetet tartalmazó fájllal, majd végül az eredményt kiírják.

Ha a futtatást parancssorból akarjuk végezni, lehetőség van arra is. Amennyiben a 10.1.2 pontban leírt módon végeztük a fordítást, a futtatás a következő, a project gyökerében kiadott parancssal végezhető:

```
java -classpath .\bin\ proto.Prototester
```

Ez a parancs elindítja a prototípust. Ha egy adott teszesetet akarunk lefuttatni, akkor azt a bemenet átírányításával tehetjük meg:

```
java -classpath .\bin\ proto.Prototester < test1.txt
```

Ekkor a test1.txt-ben megadott parancsok hajtódnak végre a prototípusban. Ha az adott teszeset kimenetét össze akarjuk vetni az elvárt kimenettel, akkor futtassuk a prototípust a be- és kimenet átírányításával, majd az így kapott kimeneti fájlt és az elvárt kimenetet tartalmazó fájlt megadva paraméterként, futtassuk a szövegfájlok összehasonlítására használt segédprogramot (TxtComparer):

```
java -classpath .\bin\ proto.ProtoTester < test2.txt > test2.out
java -classpath .\bin\ txtComparer.TxtComparer test2.out test2.expected
```

A szövegfájlok összehasonlítására használt segédprogram futtatásához, amennyiben a 10.1.2 pontban leírt módon végeztük a fordítást, a projekt gyökerében adjuk ki a következő parancsot:

```
java -classpath .\bin\ txtComparer.TxtComparer (egyik fajl) (masik fajl)
```

10.2. Tesztek jegyzőkönyvei

10.2.1. Teszteset1

Tesztelő neve	Elekes Tamás
Teszt időpontja	2014.04.21.

Tesztelő neve	Elekes Tamás
Teszt időpontja	2014.04.19
Teszt eredménye	Az enemyTypeGem-mel fejlesztett torony nem növelte a sebzést.
Lehetséges hibaok	A kristályt nem regisztrálta be, elírás.
Változtatások	A bullet-ben az ellenségtípusok nagy betűvel kezdődtek, míg a kristályban kis betűvel voltak.

10.2.2. Teszteset2

Tesztelő neve	Rédey Bálint, Elekes Tamás
Teszt időpontja	2014.04.21

Tesztelő neve	Elekes Tamás
Teszt időpontja	2014.04.20
Teszt eredménye	Az obstacle nem használódott el.
Lehetséges hibaok	Rossz értékre van állítva az obstacle életereje, vagy nem csökkeneti.
Változtatások	Az akadály életerejét akkorára állítottam hogy 1 ellenség is tönkretegye.

10.2.3. Teszteset3

Tesztelő neve	Elekes Tamás
Teszt időpontja	2014.04.21.

Tesztelő neve	Rédey Bálint, Elekes Tamás
Teszt időpontja	2014.04.21
Teszt eredménye	Hibás: első update után nem sebzi meg human0-t tower0, második update után kettévágja, harmadik után szintén.
Lehetséges hibaok	tower.setPaths nem működik helyesen, torony rossz ütemben lő

Változtatások	Bemeneti parancs, kimenet rosszul volt megadva, azon változtattunk.
---------------	---

Tesztelő neve	Elekes Tamás
Teszt időpontja	2014.04.19
Teszt eredménye	A haze mintha nem csökkentené a hatósugarat.
Lehetséges hibaok	A haze() metódus beállította az új range értéket, majd az upgradeRange() metódust hívta. Ami újra megváltoztatta a range-t.
Változtatások	A haze upgradeRange helyett a setPaths()-t hívja.

10.2.4. Teszteset4

Tesztelő neve	Rédey Bálint
Teszt időpontja	2014.04.21. 15:45

Tesztelő neve	Rédey Bálint
Teszt időpontja	2014.04.21
Teszt eredménye	Hibás: nem írja a kereszteződéshez érést: crossroad, kirajzolás-kor rossz helyen vannak az enemy-k
Lehetséges hibaok	Hiányzik valahol egy kiírás, drawmap vagy az ellenségek mozgása nem működik helyesen.
Változtatások	Enemy.move-ban elhelyeztem a crossroad kiírását, ami hiányzott, ehhez Path-ba elhelyeztem egy getNextPaths() metódust, ami a nextPaths listával tér vissza. minden más helyesen működött, a megadott parancsok között volt hiba: kereszteződésnél a dwarf0 visszafele lépett.

10.3. Értékelés

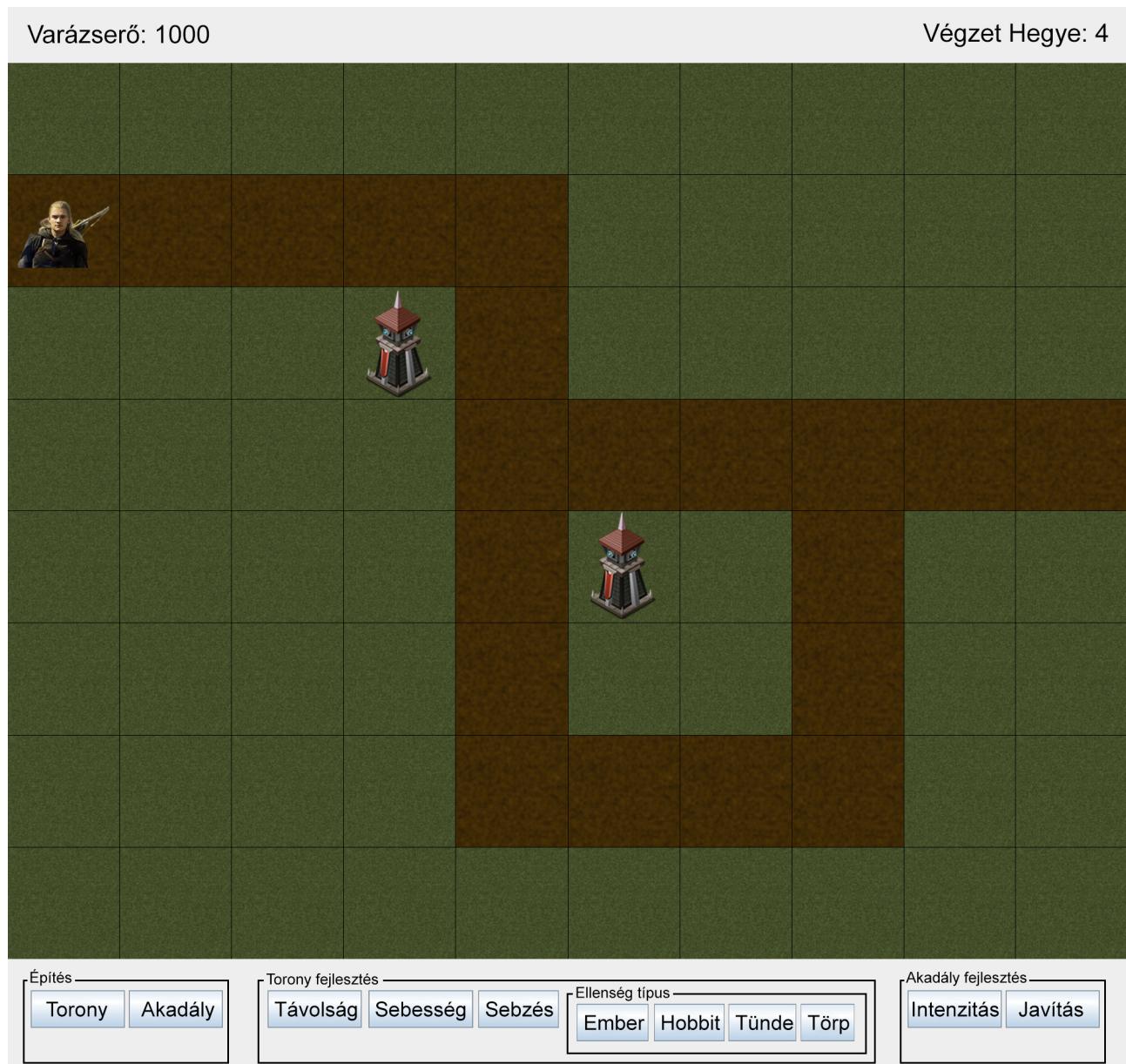
Tag	Munkaóra	Munka százalékban	Aláírás
Elekes	67,1 óra	25 %	
Fuksz	59,2 óra	24 %	
Nagy	31 óra	5 %	
Rédey	59,2 óra	24 %	
Seres	59,3 óra	22 %	

10.4. Napló

Kezdet	Időtartam	Résznevők	Leírás
2014.04.09. 08:15	1 óra	Elekes Fuksz Rédey Seres Nagy	Konzultáció
2014.04.09. 10:00	2 óra	Fuksz	Térkép betöltés implementálása.
2014.04.09. 12:00	2 óra	Elekes	Tower, Bullet, Controller, ProtoTester implementálása.
2014.04.09. 14:00	2 óra	Fuksz	Térkép kirajzolás implementálása.
2014.04.12. 20:00	1 óra	Seres	Enemy és leszármazottainak implementálása.
2014.04.18. 14:30	1 óra	Rédey	Game osztály implementálása, kommentezés
2014.04.20. 20:00	4,5 óra	Elekes Fuksz Nagy Seres Rédey	Skype konferencia
2014.04.20. 20:00	4 óra	Elekes	Bug-ok javítása.
2014.04.20. 20:00	2 óra	Seres	Enemy és leszármazottainak hibajavítása, szövegfájl összehasonlító program elkészítése, teszteléshez használt batch fájlok megírása
2014.04.21. 12:30	2 óra	Rédey	4-as, 4-es tesztesetek vizsgálása, hibák keresése, teszt jegyzőkönyvezés
2014.04.21. 14:00	1,5 óra	Elekes	Tesztesetek véglegesítése, dokumentálás.
2014.04.21. 20:00	30 perc	Seres	10.1.2, 10.1.3 pontok kitöltése.
2014.04.22. 00:00	2 óra	Fuksz	Dokumentáció véglegesítése

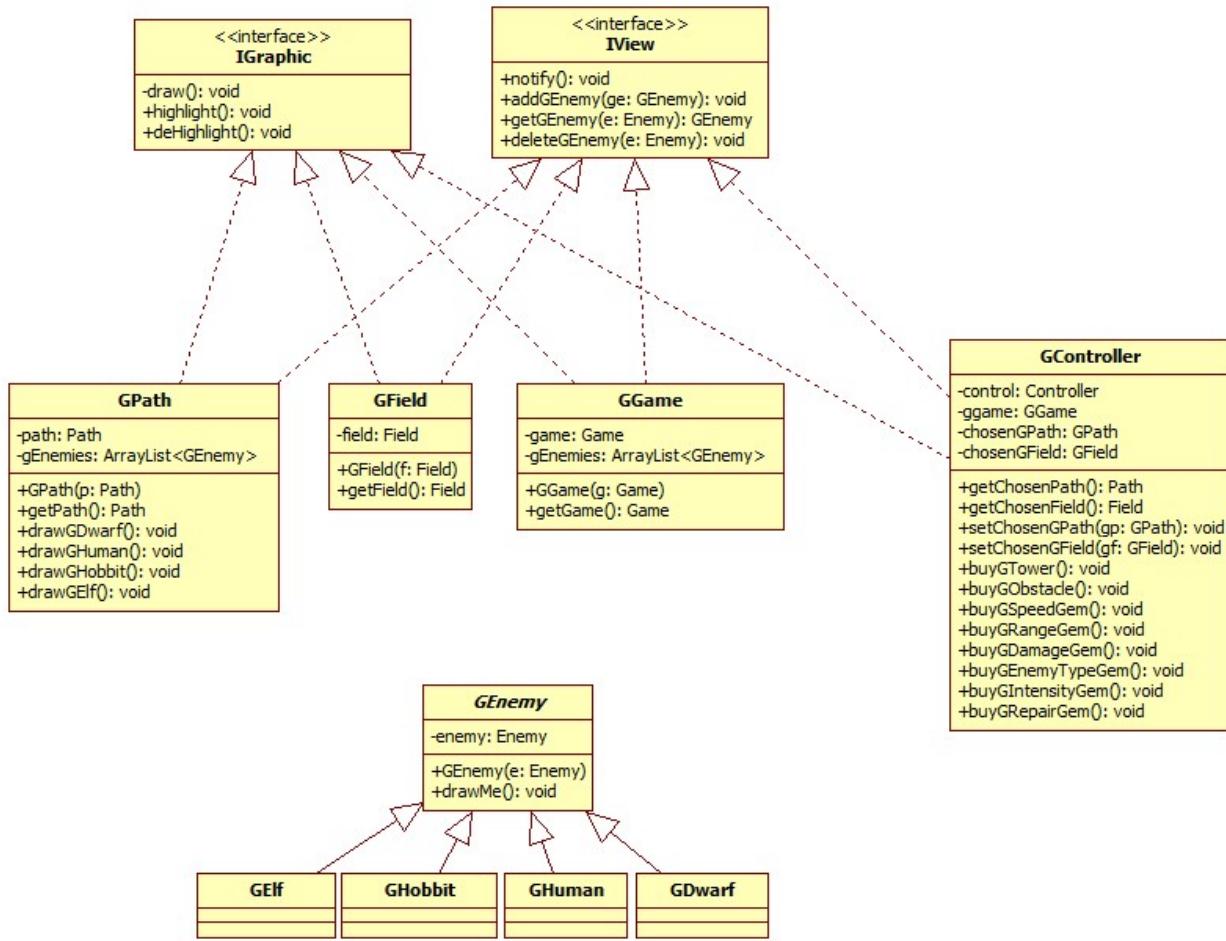
11. Grafikus felület specifikációja

11.1. A grafikus interfész



11.1. ábra. Grafikus interfész

11.2. A grafikus rendszer architektúrája



11.2. ábra. Grafikus rendszer struktúra diagram

11.2.1. A felület működési elve

A grafikus felület kialakításánál elsődleges célunk az volt, hogy a program modelljébe ne épüljön bele a megjelenítés mivolta, könnyen cserélhető legyen.

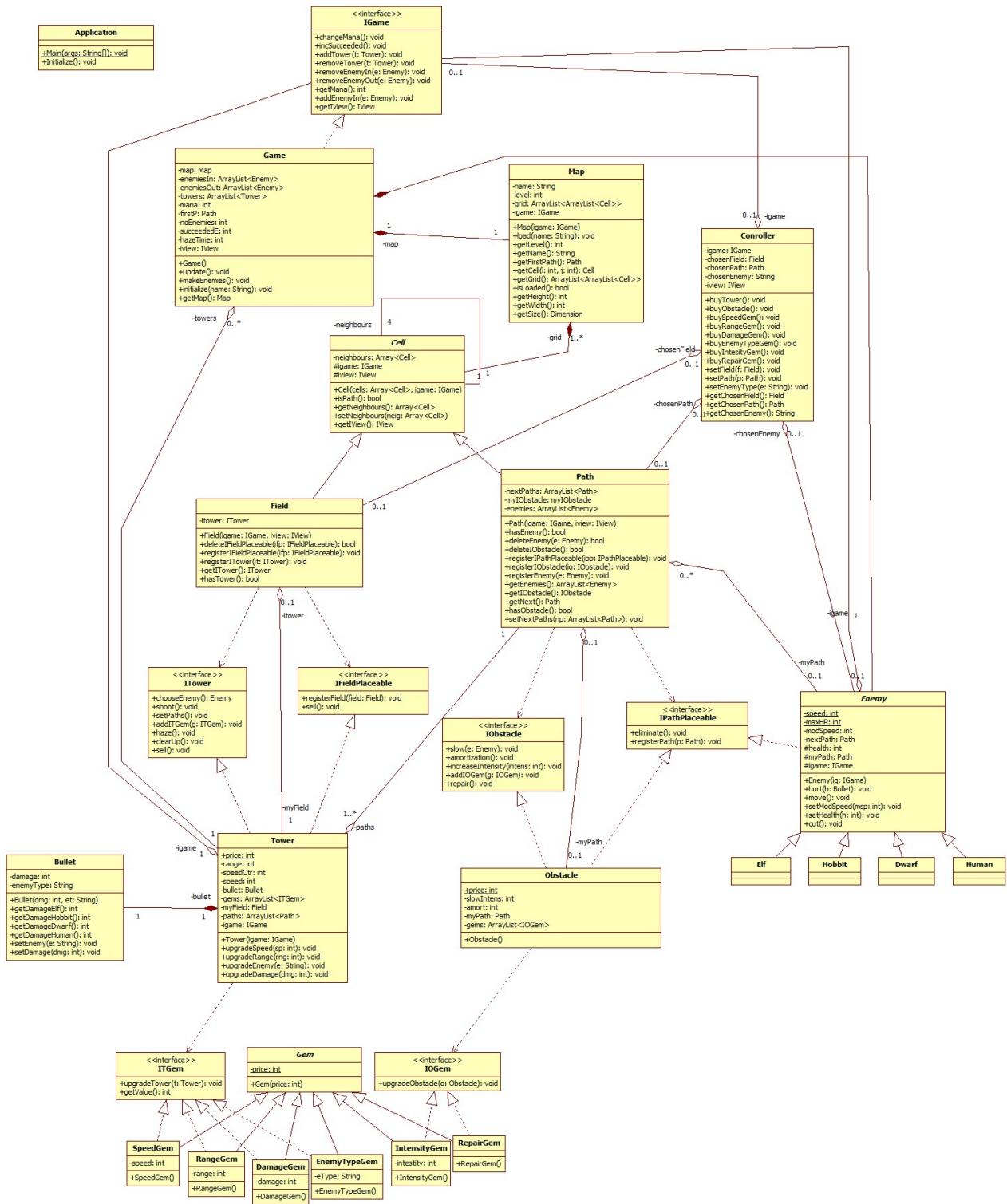
Ezért létrehoztunk a megjelenítendő elemeknek egy-egy testvér osztályt amik az objektum kirajzolásáért felelnek. Két interfésszel általánosítottuk a grafikus objektumokat.

Az IGraphic interfész megvalósító osztályok a draw() metódussal tudják kirajzolni magukat.

Az IView interfészen keresztül tudnak a testvérosztályai hivatkozni, és a notify() metóduson keresztül értesíthető, hogy változott, aminek hatására kirajzolja magát.

Ez a megoldás, miszerint modell beli objektum minden változáskor értesíti a felületet push alapú megoldás.

11.2.2. A felület osztály-struktúrája



11.3. ábra. Osztály diagram

11.3. A grafikus objektumok felsorolása

11.3.1. GPath

- Felelősség

A GPath felel a Path osztály megjelenítéséért, kirajzolásáért. minden Path-hoz tartozik egy, és mindenikhez egy Path tartozik.

- Ősosztályok

Object, Graphic abstract

- Interfészek

IView

- Attribútumok

-path: Path Referencia arra az útra, amelyikhez tartozik. Ezen keresztül kérheti le a szükséges adatokat a modellből.

-gEnemies: ArrayList<GEnemy> Az path enemies-hez tartozó GEnemy-k listája.

- Metódusok

+drawGHuman/GHobbit/GElf/GDwarf: void A különböző GEnemy-khez tartozó rajzoló függvény.

+getPath(): Path Getter path-hoz

+GPath(p: Path) Konstruktor.

11.3.2. GField

- Felelősség

A GField felel a Field osztály megjelenítéséért, kirajzolásáért. minden Field-hez tartozik egy, és mindenikhez egy Field tartozik.

- Ősosztályok

Object, Graphic abstract

- Interfészek

IView

- Attribútumok

-field: Field Referencia arra mezőre, amelyikhez tartozik. Ezen keresztül kérheti le a szükséges adatokat a modellből.

- Metódusok

+getField(): void Getter fieldhez.

+GField(f: Field) Konstruktor.

11.3.3. GGame

- Felelősség

A GGame felel a Game osztály megjelenítéséért. Elsősorban a felső sávért, vagyis a Game azon attribútumainak megjelenéséért, amiket a játékosnak célszerű látnia: mana, bejutott ellenségek száma, pálya szintje stb.

- Ősosztályok

Object, Graphic abstract

- Interfészek

IView

- Attribútumok

-game: Game A Game referencia, ami ahhoz kell, hogy meg tudja jeleníteni az adatokat.

-gEnemies ArrayList<GEnemy>: A Game enemiesIn és enemiesOut-ban lévő Enemy-khez tartozó megjelenítők listája.

- Metódusok

+getGame() Game: Getter game-hez.

+GGame(g: Game) Konstruktur.

11.3.4. GEnemy és leszármazottai: GHobbit, GHuman, GElf, GDwarf

- Felelősség

A GEnemy felel azért, hogy a GPath tudja, hogy milyen típusú Enemy-keket kell rajzolni. Ezt az enemy attribútumból tudja meg, illetve drawMe metódus különféle implementációinak segítségével üzen GEnemy a GPath-nak. minden Enemy-hez létrehozáskor társítunk egy példányt, amit elhelyezünk a GGame-ben.

- Ősosztályok

Object

- Interfészek

Nincs

- Attribútumok

-enemy: Enemy Ez a hivatkozási pont a modellbeli Enemy-re.

- Metódusok

+drawMe(gp: GPath): void Ez a metódus gondoskodik arról, hogy a megfelelő típusú Enemy kirajzolása történjen meg a Path-on. Visszahívja gp az enemy típusának megfelelő drawXXX() metódust.

+GEnemy(e: Enemy) Konstruktur.

11.3.5. Controller

- Attribútumok

-iview: IView Interfész az értesítéshez.

- Metódusok

+getView() IView: Getter az iview-hoz.

11.3.6. GController

- Felelősség

A GController irányítja a grafikus elemeket, végzi a felhasználói interakciók érvényre jutásának a felületre vonatkozó részeit, illetve kezdeményezi a modell megváltoztatását.

- Ősosztályok

Object, Graphic abstract

- Interfészek

IView

- Attribútumok

-control: Controller Ez a hivatkozási pont a modellhez.

-chosenGPath: Path A kijelölt út, pl akadály vétele.

-chosenGField: Field A kijelölt mező, pl torony vétele.

-ggame: GGame Ez a hivatkozási pont a grafika másik részéhez.

- Metódusok

+set chosenGField/GPath(gf/gp GField/GPath) void: GField/GPath setter.

+get chosenGField/GPath(): void GField/GPath getter.

+buyGSpeed/Range/Damage/EnemyType/Intesity/RepairGem(): void Gem-ek vétele.

+buyTower/Obstacle() Tower/Obstacle vétele.

11.3.7. IView

- Felelősség

Az IView interfész teremt kapcsolatot a modellből a grafikus felület felé. Segítségével lehet értesíteni a felületet a modellbeli változásokról. Ehhez különböző értesítő metódusokat tartalmaz.

- Ősosztályok

Nincs

- Interfészek

Nincs

- Metódusok

+notify() void: Alap értesítő metódus.

- +addGEnemy(ge: GEnemy)** void: Hozzáadunk az iview-hoz egy GEnemy-t.
- +getGEnemy(e: Enemy)** void: Lekérjük az e-hez tartozó GEnemy-t az iview-ból.
- +deleteGEnemy(e: Enemy)** void: Töröljük az e-hez tartozó GEnemy-t az iview-ból.

11.3.8. Graphic

- **Felelősség**
Az Graphic a rajzoló abstract ősosztály. Különböző rajzolásokat tesz lehetővé.
- **Ősosztályok**
Nincs
- **Interfészek**
Nincs
- **Metódusok**

- draw()** void: Alap rajzoló metódus.
- +highlight()** void: Kijelölő metódus, pl mezőre kattintáshoz.
- +deHighlight()** void: Highlight inverze.

11.3.9. Cell

- **Attribútumok**
- #iview: IView** Interfész a grafika értesítéséhez.
- #igame: IGame** A Path és Field igame-je felkerült ide, protected lett.
- **Metódusok**

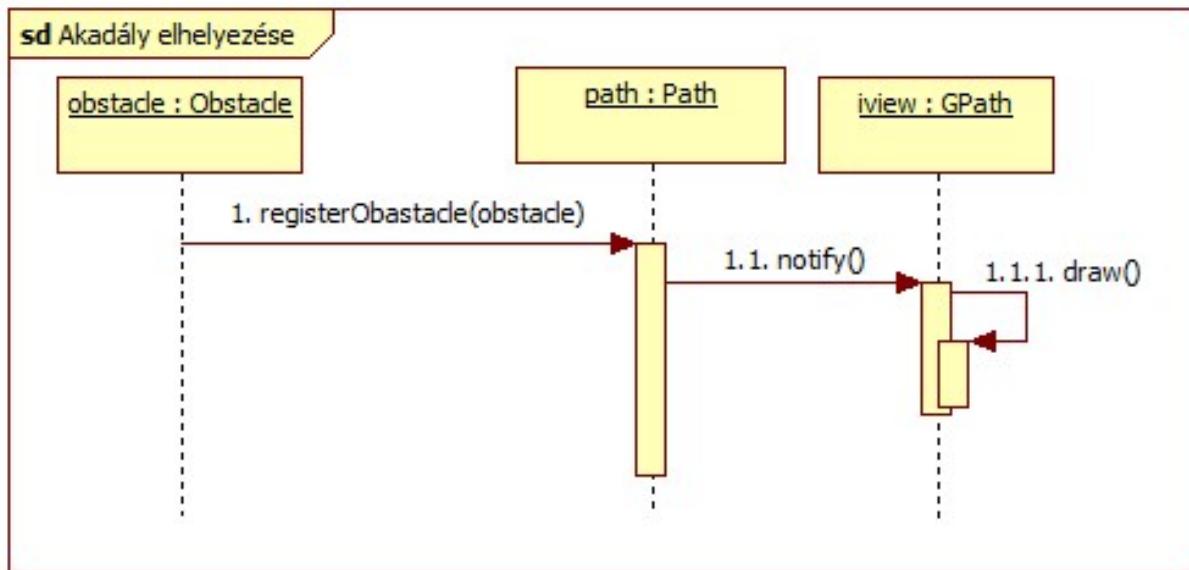
- +getIView()** IView: Getter az iview-hoz.

11.3.10. Game és IGame

- **Attribútumok**
- iview: IView** Game-ben interfész az értesítéshez.
- **Metódusok**
- +getIView(): IView** IGame-be getter, hogy mások is elérjék, akik a Gameet elérík az interfészén keresztül.

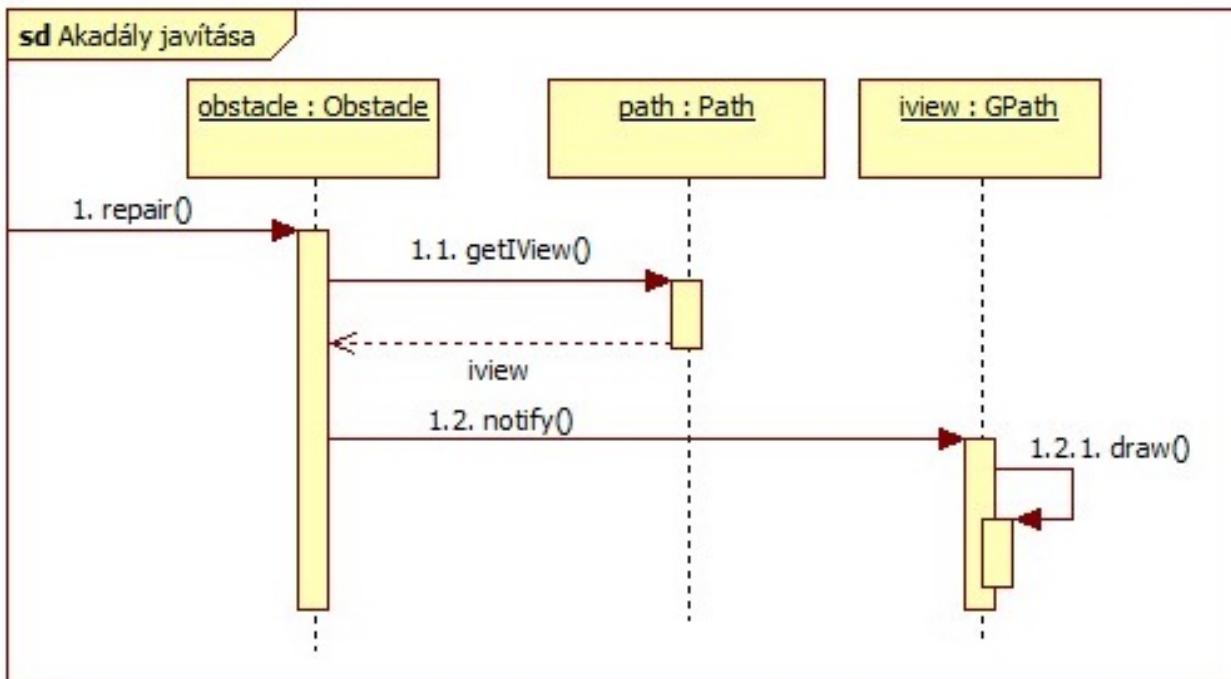
11.4. Kapcsolat az alkalmazói rendszerrel

11.4.1. Akadály elhelyezése



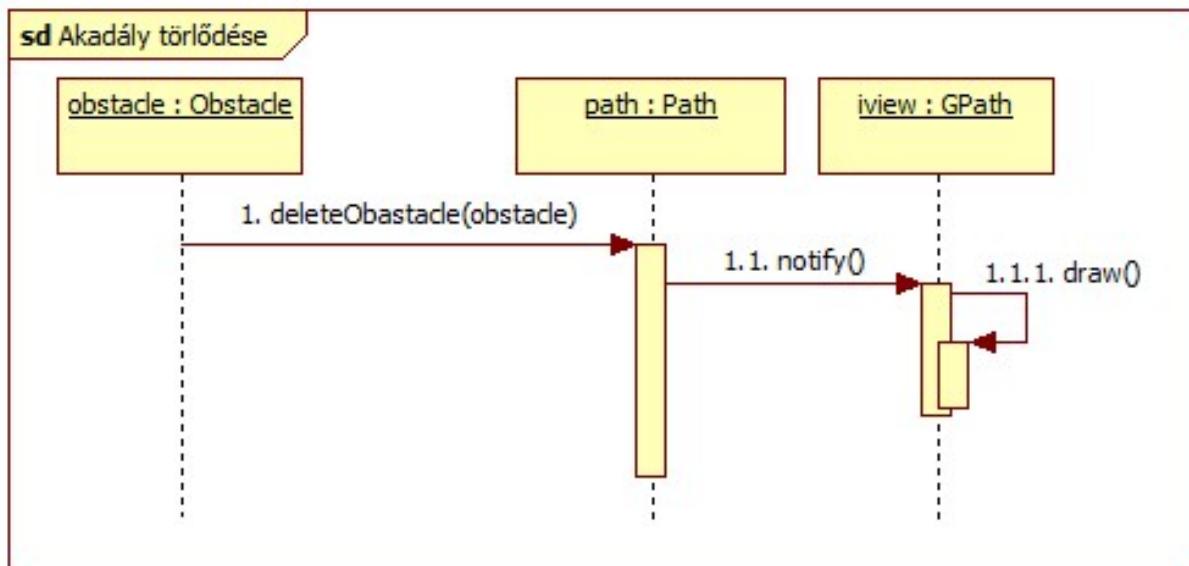
11.4. ábra. Akadály elhelyezése szekvenciadiagram

11.4.2. Akadály javítása



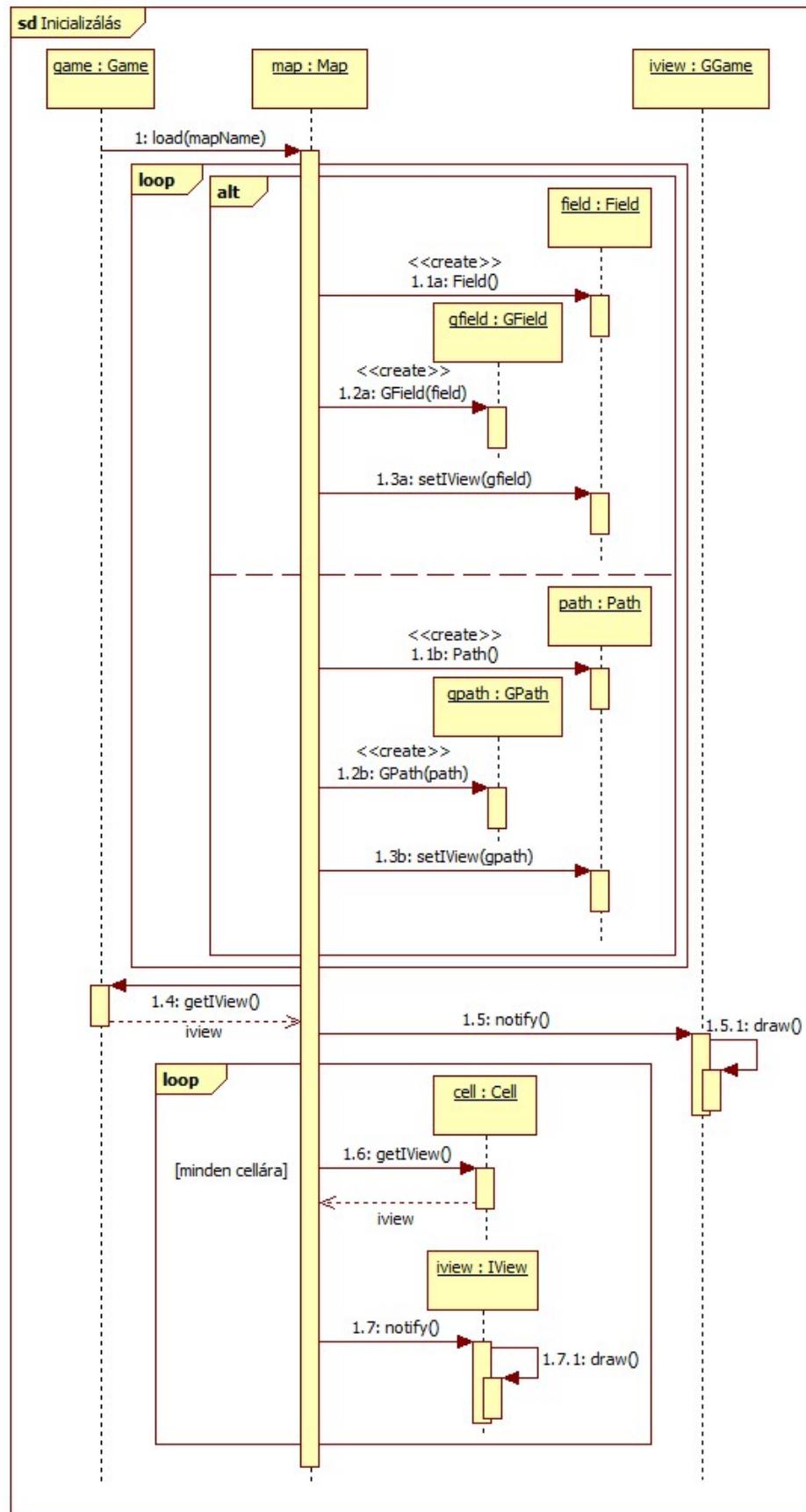
11.5. ábra. Akadály javítása szekvenciadiagram

11.4.3. Akadály törlődése

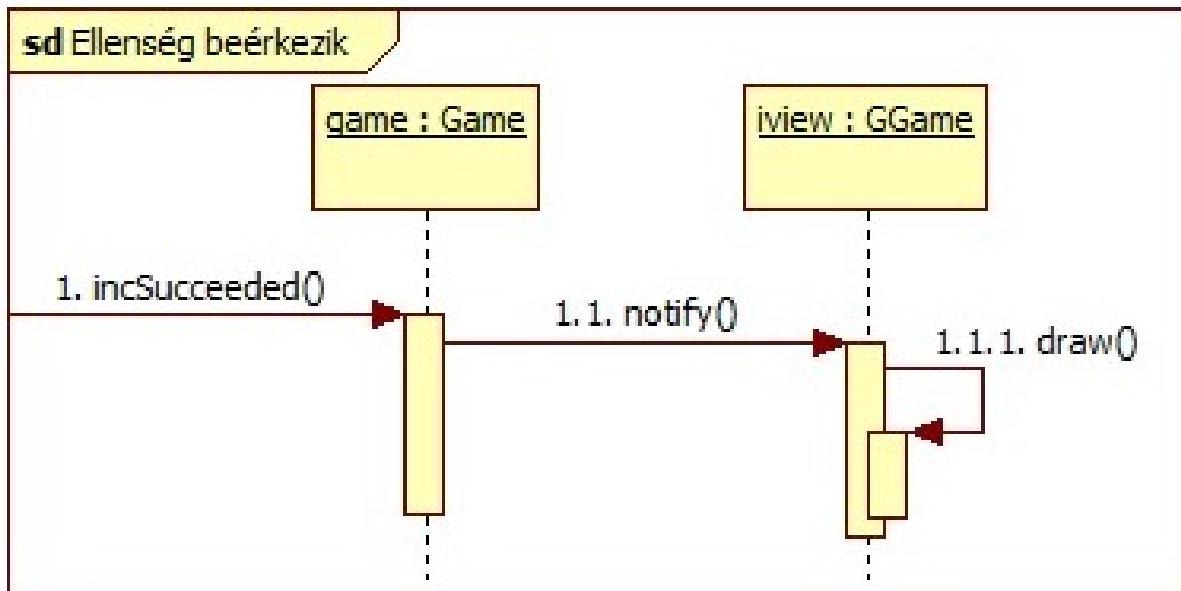


11.6. ábra. Akadály törlődése szekvenciadiagram

11.4.4. Betöltés

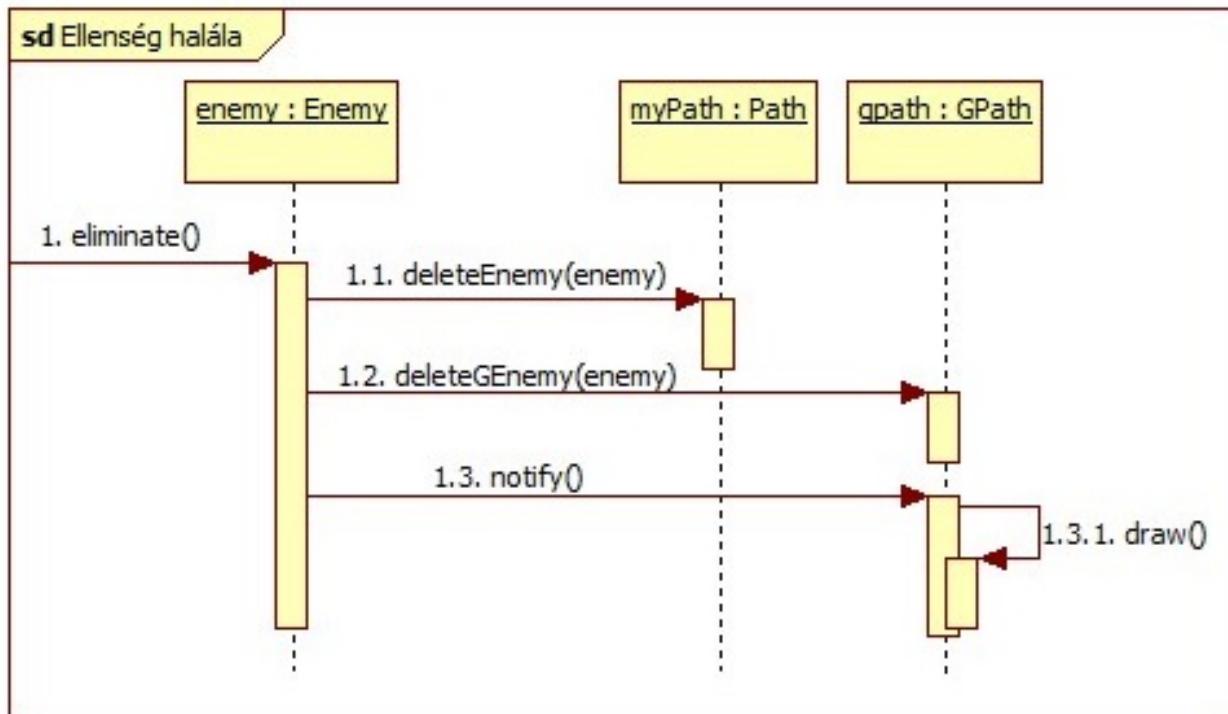


11.4.5. Ellenség beérkezik



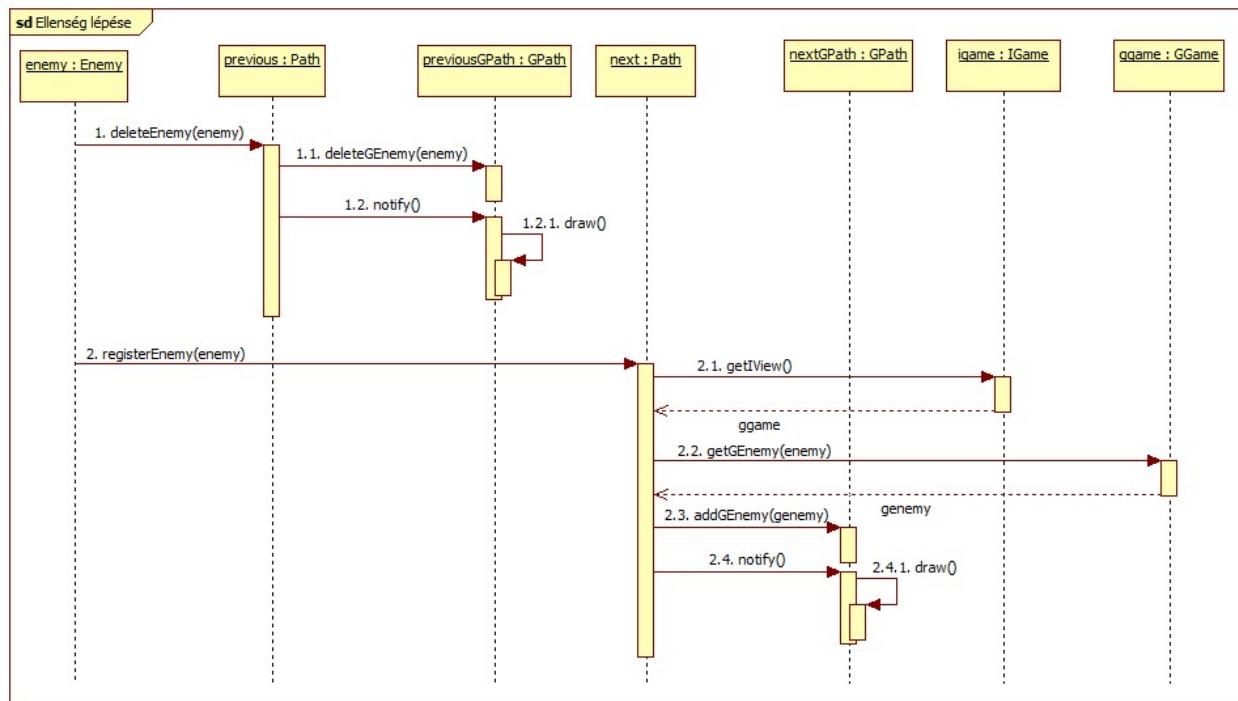
11.8. ábra. Ellenség beérkezik szekvenciadiagram

11.4.6. Ellenség halála



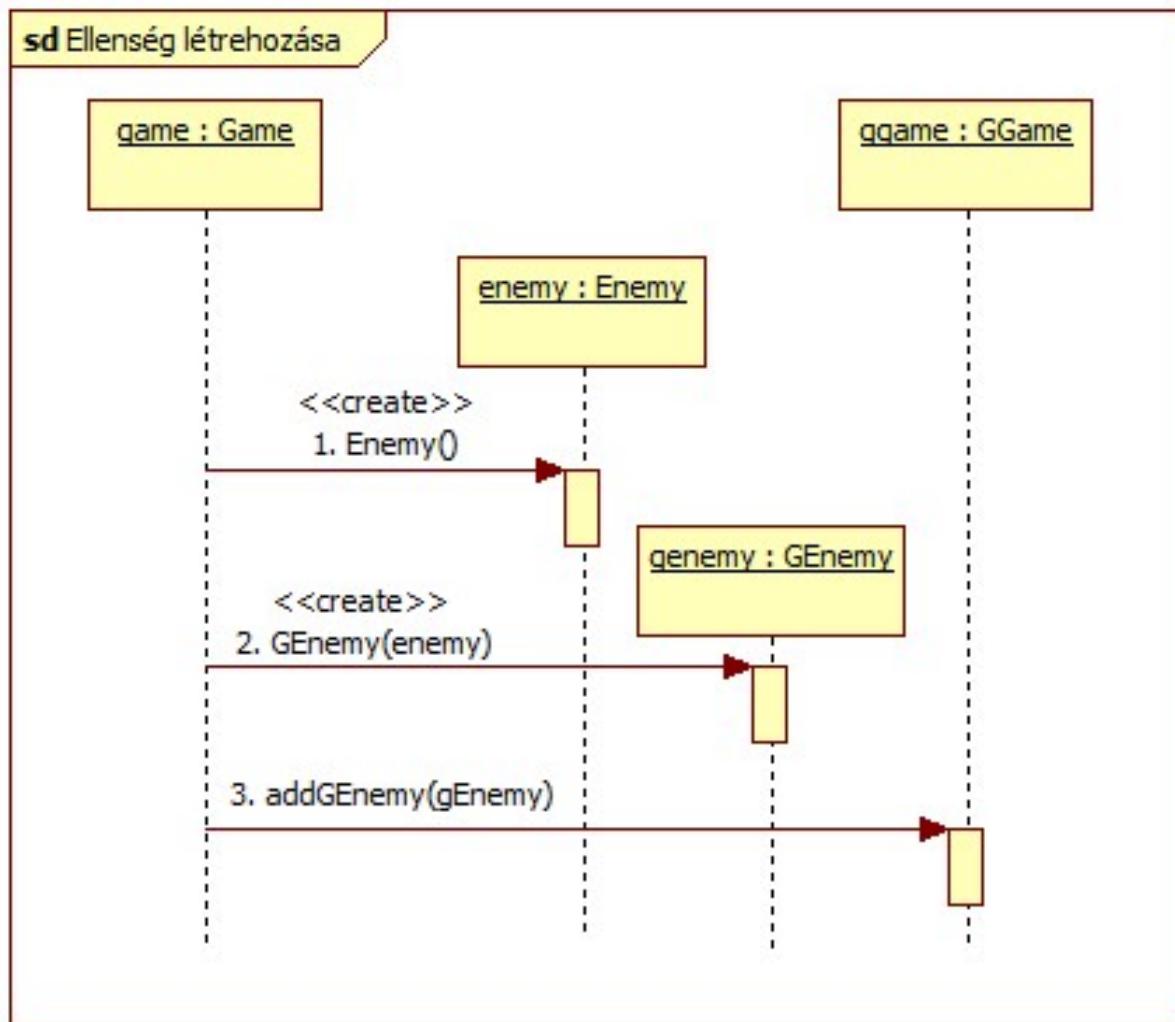
11.9. ábra. Ellenség halála szekvenciadiagram

11.4.7. Ellenség lépése



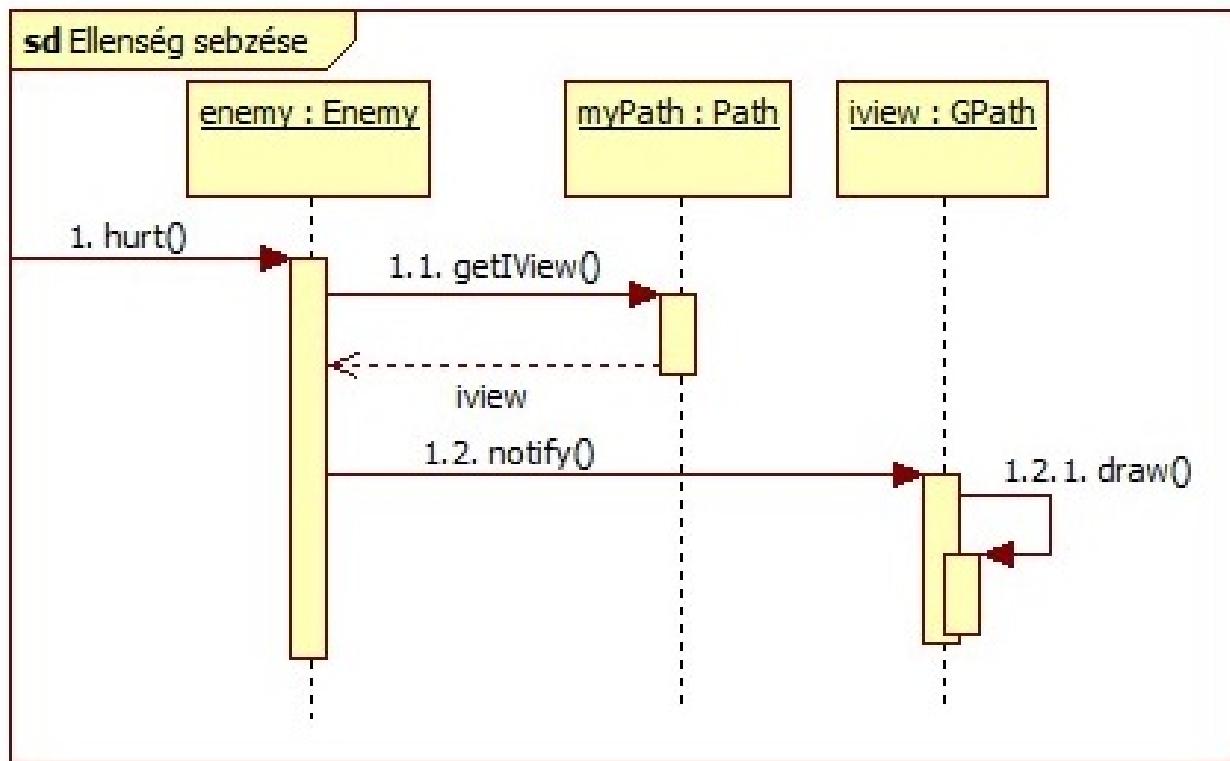
11.10. ábra. Ellenség lépése szekvenciadiagram

11.4.8. Ellenség létrehozása



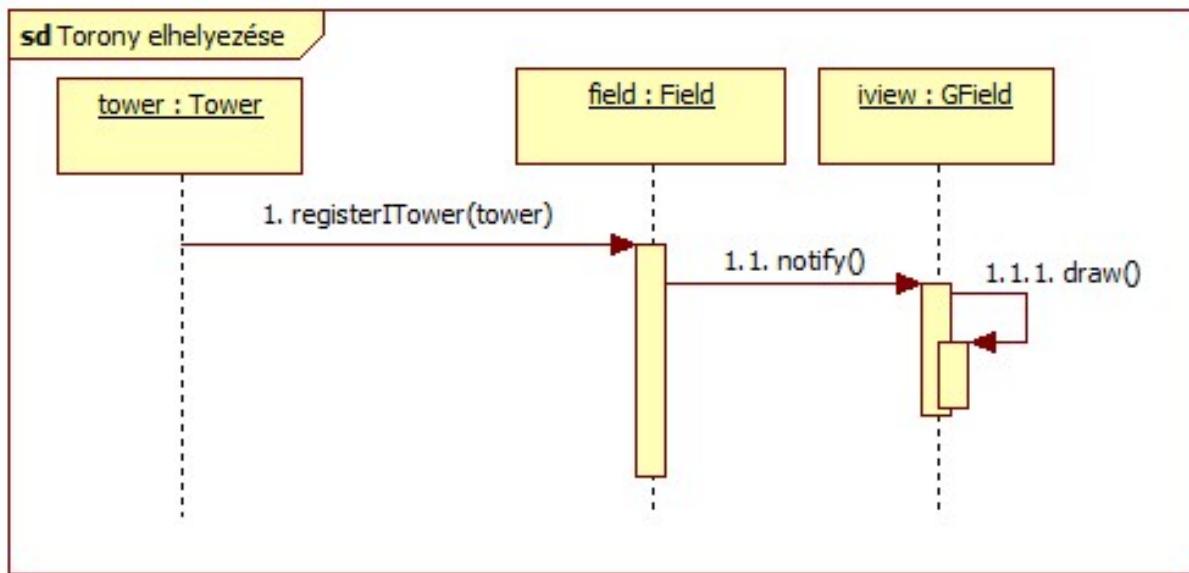
11.11. ábra. Ellenség létrehozása szekvenciadiagram

11.4.9. Ellenség sebzése



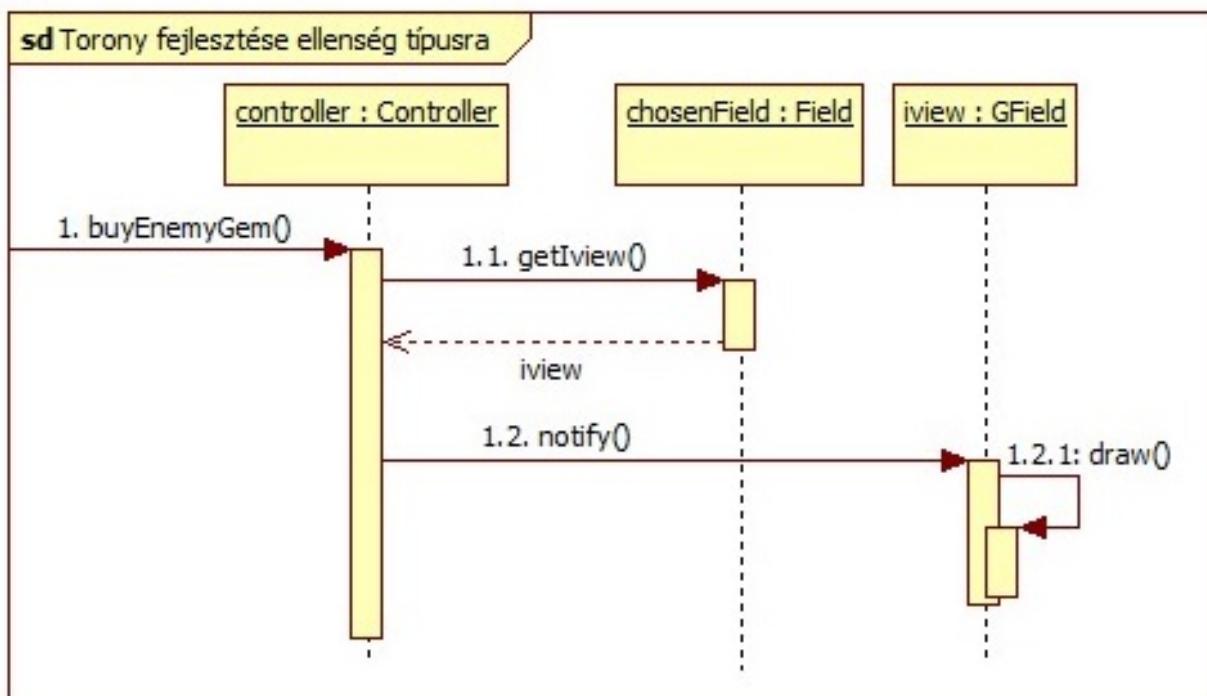
11.12. ábra. Ellenség sebzése szekvenciadiagram

11.4.10. Torony elhelyezése



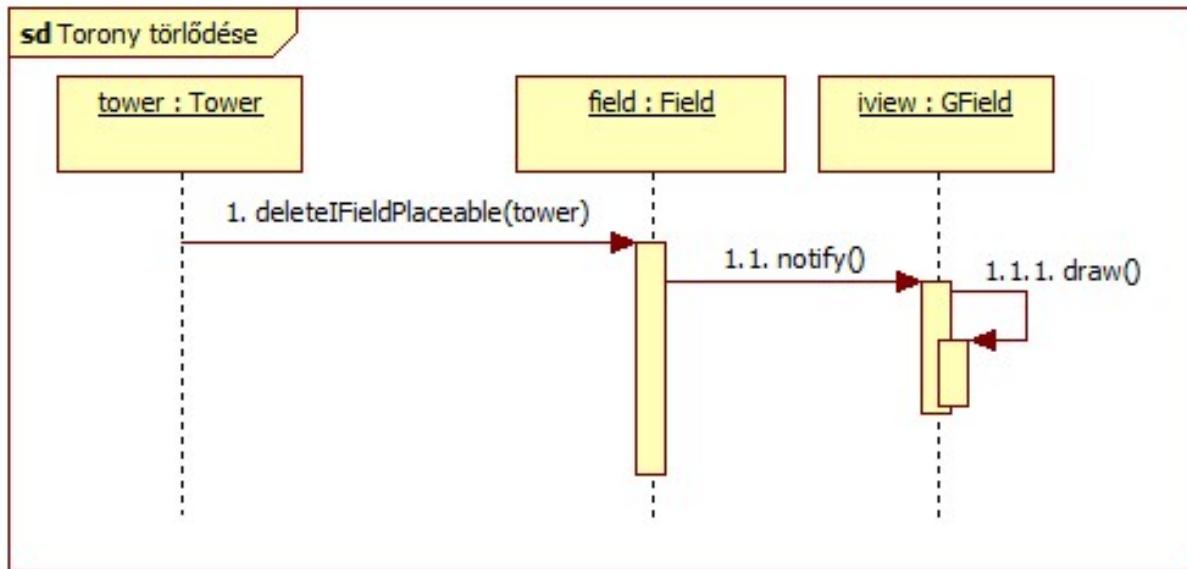
11.13. ábra. Torony elhelyezése szekvenciadiagram

11.4.11. Torony fejlesztése ellenség típusra



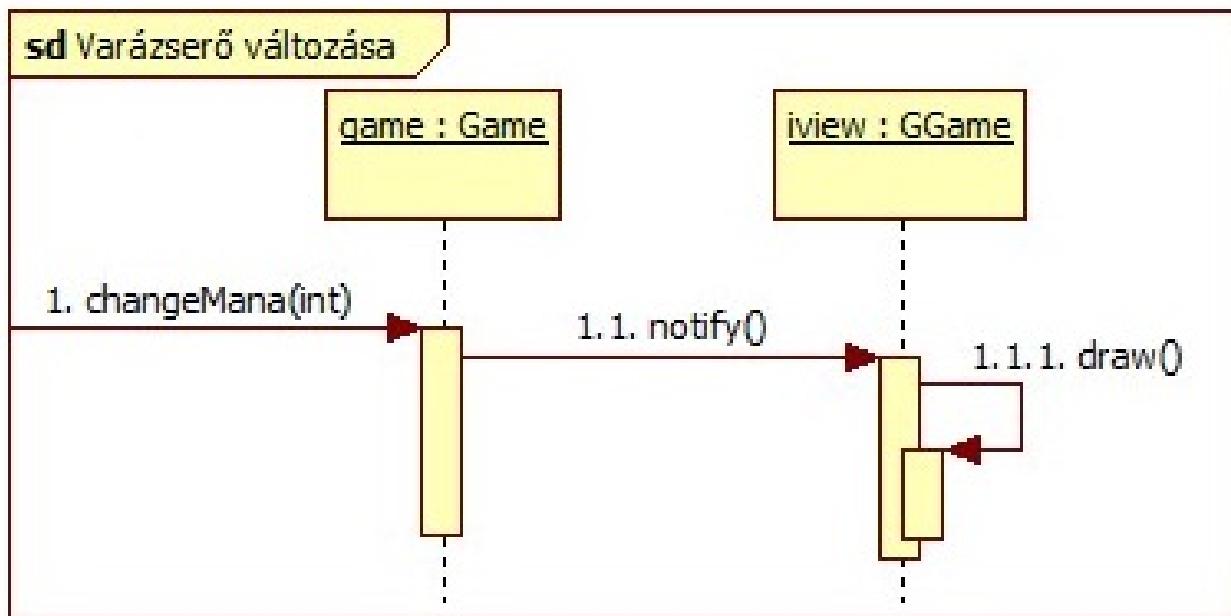
11.14. ábra. Torony fejlesztése ellenség típusra szekvenciadiagram

11.4.12. Torony törlődése



11.15. ábra. Torony törlődése szekvenciadiagram

11.4.13. Varázserő változása



11.16. ábra. Varázserő változása szekvenciadiagram

11.5. Napló

Kezdet	Időtartam	Résznevők	Leírás
2014.04.23. 08:15	1,5 óra	Elekes Fuksz Nagy Rédey Seres	Konzultáció
2014.04.26 11:00	4 óra	Rédey	Grafikus osztályok, MVC kapcsolatok megal-kotása, 11.2.2
2014.04.26. 17:00	3 óra	Seres	Grafikus vázlat készítése a játékról
2014.04.26. 20:00	1 óra	Elekes	Grafikus felület specifikációja
2014.04.27 14:00	2 óra	Rédey	Grafikus osztályok finomítása, 11.3 objektum katalógus
2014.04.27. 22:00	4 óra	Seres	Grafikus működés szekvencia diagramjai
2014.04.28. 03:00	2 óra	Fuksz	Dokumentáció véglegesítése

13. Grafikus felület specifikációja

13.1. Fordítási és futtatási útmutató

13.1.1. Fájllista

Fájl neve	Méret	Keletkezés ideje	Tartalom
graph/Application.java	7143	2014.03.11.	Névvkel megegyező osztály.
graph/Bullet.java	1836	2014.03.11.	Névvkel megegyező osztály.
graph/Cell.java	1195	2014.03.11.	Névvkel megegyező osztály.
graph/Controller.java	4911	2014.03.11.	Névvkel megegyező osztály.
graph/DamageGem.java	418	2014.03.11.	Névvkel megegyező osztály.
graph/Dwarf.java	1203	2014.03.11.	Névvkel megegyező osztály.
graph/Elf.java	1107	2014.03.11.	Névvkel megegyező osztály.
graph/Enemy.java	4311	2014.03.11.	Névvkel megegyező osztály.
graph/EnemyTypeGem.java	463	2014.03.11.	Névvkel megegyező osztály.
graph/Field.java	1088	2014.03.11.	Névvkel megegyező osztály.
graph/Game.java	5923	2014.03.11.	Névvkel megegyező osztály.
graph/GCell.java	1144	2014.03.11.	Névvkel megegyező osztály.
graph/GController.java	11912	2014.03.11.	Névvkel megegyező osztály.
graph/GDwarf.java	261	2014.03.11.	Névvkel megegyező osztály.
graph/GElf.java	253	2014.03.11.	Névvkel megegyező osztály.
graph/Gem.java	274	2014.03.11.	Névvkel megegyező osztály.
graph/GEnemy.java	794	2014.03.11.	Névvkel megegyező osztály.
graph/GField.java	2834	2014.03.11.	Névvkel megegyező osztály.
graph/GGame.java	3025	2014.03.11.	Névvkel megegyező osztály.
graph/GHobbit.java	264	2014.03.11.	Névvkel megegyező osztály.
graph/GHuman.java	261	2014.03.11.	Névvkel megegyező osztály.
graph/GPath.java	4732	2014.03.11.	Névvkel megegyező osztály.
graph/Graphic.java	594	2014.03.11.	Névvkel megegyező osztály.
graph/Hobbit.java	1210	2014.03.11.	Névvkel megegyező osztály.
graph/Human.java	1200	2014.03.11.	Névvkel megegyező osztály.
graph/IFieldPlaceable.java	127	2014.03.11.	Névvkel megegyező osztály.
graph/IGame.java	470	2014.03.11.	Névvkel megegyező osztály.
graph/IntensityGem.java	419	2014.03.11.	Névvkel megegyező osztály.
graph/IObstacle.java	216	2014.03.11.	Névvkel megegyező osztály.
graph/IOGem.java	96	2014.03.11.	Névvkel megegyező osztály.
graph/IPathPlaceable.java	129	2014.03.11.	Névvkel megegyező osztály.
graph/ITGem.java	115	2014.03.11.	Névvkel megegyező osztály.
graph/ITower.java	267	2014.03.11.	Névvkel megegyező osztály.
graph/IView.java	853	2014.03.11.	Névvkel megegyező osztály.
graph/Map.java	4753	2014.03.11.	Névvkel megegyező osztály.
graph/Obstacle.java	1488	2014.03.11.	Névvkel megegyező osztály.
graph/Path.java	5241	2014.03.11.	Névvkel megegyező osztály.
graph/RangeGem.java	379	2014.03.11.	Névvkel megegyező osztály.
graph/RepairGem.java	256	2014.03.11.	Névvkel megegyező osztály.
graph/ResourcesCache.java	2244	2014.03.11.	Névvkel megegyező osztály.

Fájl neve	Méret	Keletkezés ideje	Tartalom
graph/SpeedGem.java	420	2014.03.11.	Névvel megegyező osztály.
graph/Tower.java	4254	2014.03.11.	Névvel megegyező osztály.
gameImages/field.jpg	53033	2014.05.07.	Mező textúra.
gameImages/game_over.jpeg	3634	2014.05.11.	Játék vége kép.
gameImages/path.jpg	27463	2014.05.07.	Út textúra.
gameImages/dwarf.png	6336	2014.05.07.	Törp kép.
gameImages/elf.png	8279	2014.05.07.	Tündér kép.
gameImages/hobbit.png	5902	2014.05.07.	Hobbit kép.
gameImages/human.png	8592	2014.05.07.	ember kép.
gameImages/tower1.png	14177	2014.05.07.	Alap torony kép.
gameImages/tower2.png	14783	2014.05.07.	Fejlesztett torony kép.
gameImages/tower3.png	14211	2014.05.07.	Fejlesztett torony kép.
gameImages/tower4.png	12619	2014.05.07.	Fejlesztett torony kép.
gameImages/tower5.png	14496	2014.05.07.	Fejlesztett torony kép.
map1.txt	39	2014.05.07.	Játék térkép.
map2.txt	27	2014.05.07.	Játék térkép.
map3.txt	33	2014.05.07.	Játék térkép.
map4.txt	67	2014.05.07.	Játék térkép.
map5.txt	404	2014.05.07.	Játék térkép.
map6.txt	405	2014.05.07.	Játék térkép.
map7.txt	275	2014.05.07.	Játék térkép.
GraphicCompileAndRun.bat	1128	2014.05.06.	Program fordítása és futtatása

13.1.2. Fordítás

A fordítás a "javac" programmal történik, ezért olyan környezetben lehetséges csak, ahol ez a program elérhető. A tesztelés megkönnyítése végett a fordítás és futtatás automatizálására egy batch fájl áll rendelkezésre. Ennek a használata a 13.1.3 pontban van részletezve.

A fordítás parancssorból is lehetséges a következő, a project gyökerében kiadott parancssal:

```
if exist bin rmdir bin /s /q
mkdir bin
javac -d ./bin/ graph/*.java
```

Ez a parancs lefordítja a programot, és a lefordított állományokat a bin mappába helyezi.

13.1.3. Futtatás

A futtatás a "java" programmal történik, ezért olyan környezetben lehetséges csak, ahol ez a program elérhető. Az adott batch fájlok segítségével lehetséges lefordítani és futtatni a programot: ehhez a GraphicCompileAndRun.bat-t futtassuk, ami a fordítás után azonnal el is indítja a programot.

Ha a futtatást parancssorból akarjuk végezni, lehetőség van arra is. Amennyiben a 13.1.2 pontban leírt módon végeztük a fordítást, a futtatás a következő, a project gyökerében kiadott parancssal végezhető:

```
java -classpath .\bin\ graph.Application
```

Ez a parancs elindítja a programot.

13.2. Értékelés

Tag	Munkaóra	Munka százalékban	Aláírás
Elekes	79,16 óra	22 %	
Fuksz	67,25 óra	22 %	
Nagy	33,99 óra	10 %	
Rédey	73,75 óra	22 %	
Seres	90,32 óra	24 %	

13.3. Napló

Kezdet	Időtartam	Résznevők	Leírás
2014.04.30. 08:15	1 óra	Elekes Nagy	Konzultáció
2014.05.03. 13:00	7 óra	Rédey	Korábbi dokumentumokban szereplő osztályleírások és objektumkatalógusok aktualizálása a dátum szerinti időponti állás alapján.
2014.05.05. 13:00	2 óra	Seres	Grafikus osztályok felvétele, implementálása.
2014.05.05. 19:00	7 óra	Seres	További grafikus osztályok felvétele, implementálása.
2014.05.06. 20:00	7 óra	Seres	Grafikus osztályok implementálása.
2014.05.08. 12:00	2 óra	Seres	A modell bővítése a grafikus programrésszel való együttműködés céljából. Grafikus felület javítása.
2014.05.09. 16:00	2,5 óra	Seres	További funkciók implementálása.
2014.05.10. 15:00	1,5 óra	Elekes	Hibajavítás, játék folytonos futása.
2014.05.10. 15:00	7 óra	Elekes	Programkód javítása, grafikus felület rendezése, játék vége, új játék.
2014.05.11. 16:00	1 óra	Fuksz	Új pályák létrehozása.
2014.05.11 19:00	2 óra	Fuksz	Régebbi dokumentációk formai frissítése.
2014.05.11 21:00	2 óra	Seres	Program véglegesítése.
2014.05.12 01:00	1,5 óra	Fuksz	Dokumentáció véglegesítése.

14. Összefoglalás

14.1. Projekt összegzés

Tag	Munkaidő (óra)
Elekes	80
Fuksz	68
Nagy	40
Rédey	74
Seres	91
Összesen:	353

Fázis	Forrássor
Szkeleton	2091
Protó	2576
Grafikus	3793

- Mit tanultak a projektből konkrétan és általában?
- Mi volt a legnehezebb és a legkönnyebb?
- Összhangban állt-e az idő és a pontszám az elvégzendő feladatokkal?
- Ha nem, akkor hol okozott ez nehézséget?
- Milyen változtatási javaslatuk van?
- Milyen feladatot ajánlanának a projektre?