

Tartalomjegyzék:

2. Követelmény, projekt, funkcionalitás.....	2
3. Analízis modell kidolgozása.....	13
4. Analízis modell kidolgozása.....	30
5. Szkeleton tervezése.....	46
6. Szkeleton beadás.....	68
7. Prototípus koncepciója.....	73
8. Részletes tervezek.....	94
10. Prototípus beadása.....	135
11. Grafikus felület specifikációja.....	140
13. Grafikus változat beadása.....	152
14. Összefoglalás.....	155

2. Követelmény, projekt, funkcionálitás

2.1 Bevezetés

2.1.1 Cél

A dokumentum célja a tanszék (megrendelő) által kiadott, A két torony nevet viselő projekt követelményeinek formális leírása. A dokumentum tartalmazza a kiadott feladat részletes leírását, definiálja a későbbiekben előforduló szakkifejezéseket, ismerteti a projekt fejlesztésének fázisait továbbá leírja az elkészítendő szoftver főbb use-case-eit.

2.1.2 Szakterület

A szoftver a játékpiastra készül, célja a felhasználók szórakoztatása. A játék a tower defense játékok kategóriába sorolható.

2.1.3 Definíciók, rövidítések

Konzultáció: A tanszék által adott időpontban megbeszélés a konzulenssel, aki kívülről irányítja a fejlesztés menetét, segíti a hibák kijavításában.

Értekezlet: Amikor több mint egy csapattag döntéseket szeretne hozni a projektről, a csapatvezető értekezletet hív össze. A hozott döntéseket minden esetben dokumentálni kell.

2.1.4 Hivatkozások

- <http://en.wikipedia.org>
- <http://docs.oracle.com/javase/7/docs/api/>
- <https://www.iit.bme.hu/~stuser/>
- J.R.R. Tolkien által kiadott könyvek

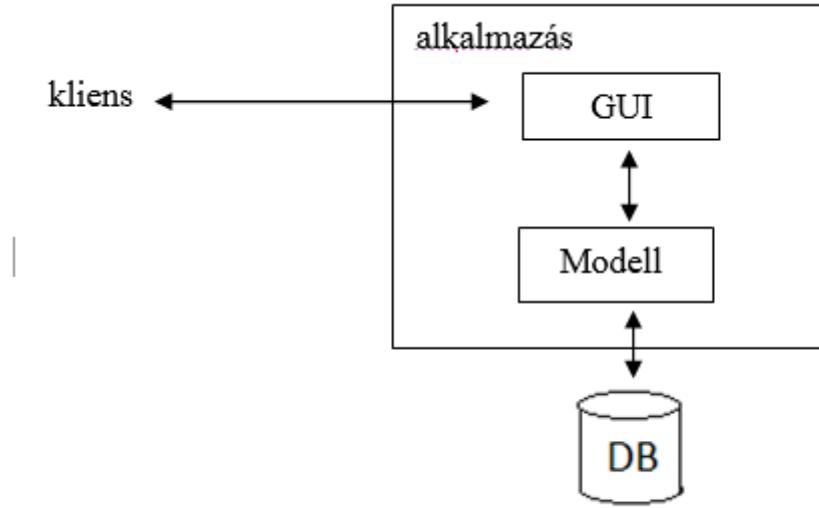
/

2.1.5 Összefoglalás

A dokumentum elején rövid áttekintést kapunk a ránk váró projekt felépítéséről, ezt követi egy részletes funkcionális leírás a program megértéséhez és egyes részeinek ismeretéhez. Ezután betekintést kapunk a követelmények, korlátozások és a use-case-k leírásába. A végén a megismerhetjük a csapat összedolgozási formáit és az egyes tagok naplózott munkáit.

2.2 Áttekintés

2.2.1 Általános áttekintés



Az alkalmazást felbonthatjuk grafikus felhasználói rétegre, illetve a felület mögött álló modell rétegre.

A modell és a grafikus felület egyensúlyt tart egymással.

GUI: grafikus felhasználói felület. Ezen keresztül kommunikál a felhasználó a programmal.

Modell: az alkalmazás mögött álló logikai modell. Ez a réteg valósítja meg a különböző adatstruktúrákat és algoritmusokat, amelyek a program magját képezik.

DB: külső adatbázis, a program forráskódján kívüli adatok. Ebben a projektben a mentett fájlok összessége.

2.2.2 Funkciók

Rövid leírás a játékról

Ez a program egy grafikus stratégiai játék. A játékos célja, hogy az ellenség ne juthasson el a pálya elejétől a végéig. A játékos rendelkezik varázserővel, melyet tornyok és akadályok építésére, valamint a tornyok fejlesztésére használhat fel. Az ellenség a pályán meghatározott úton halad, amire a játékos akadályokat helyezhet el. A tornyok meghatározott időnként tüzelnek az ellenségre. A játék az idő múlásával egyre nehezedik, több és erősebb ellenséggel kell megküzdeni. A játékos veszít, ha az ellenség akár egy tagja is eléri a célt, nyer ha ezt megakadályozza.

Fontosabb elemei, tárgyai a játéknak

Pálya: A játék pályája téglalap alakú, mely mezőből és útakból áll. A mezőre tornyokat, az utakra akadályokat építhetünk. Az ellenség csak az úton mozoghat. Több különböző útvonalon is el lehet jutni célhoz, ezzel tovább nehezítve a játéket.

Torony: A tornyok képezik a játékban a védekezés legfőbb eszközét, ezek képesek sebezni az ellenséget, akik megpróbálnak eljutni céljukhoz, a Végzet Hegyéhez. minden torony rendelkezik három fő tulajdonsággal: a tüzelés gyakorisága, a torony hatótávolsága és a sebzés nagysága. A tornyok mezőre építhetők, és varázskövekkel fejleszthetők.

Akadály: Az akadályok lassítják az ellenség haladását. Utakra lehet őket lehelyezni, és varázskövekkel lehet fejleszteni.

Varázserő: A játékos rendelkezik varázserővel. Varázserejéből tornyokat és akadályokat építhet, valamint varázsköveket vásárolhat.

Varázserőnk az egyes ellenfelek elpusztításával nő, építkezés illetve varászkővétel esetén csökken. Amennyiben varázserőnk elfogy, nem tudunk se építkezni, se vásárolni.

Varázskő: A játékos a varázserejéből varázskövetek tud venni. A varázskövekkel lehet a tornyokat és az akadályokat fejleszteni. A tornyok ereje háromféleképpen növelhető:

- a tüzelési gyakoriság növelése
- a tüzelési rádiusz (a torony hatótávolságának) növelése
- a sebzés nagyságának növelése

A sebzés nagyságát növelő köveknek több fajtája is van. Vannak drágábbak, melyek minden szövetségesre (ember, tündér, hobbit, törp) hatással vannak, de vannak olyanok is, amelyek csak bizonyos fajokat sebznek meg.

Az akadályok fejlesztésével az ellenség haladtát még tovább lassíthatjuk. Az akadályfejlesztő-köveknek szintén több fajtaja létezik. Hasonlóan a sebzésnövelő kövekhez, itt is létezik drágább, mely minden fajzatot lassít, és létezik olcsóbb, ami azonban csak bizonyos fajokra gyakorol hatást. (pl. a tündekirálynő csak a tündék figyelmét vonja el)

A gyűrű szövetségének tagjai (az ellenség): A szövetség tagjai az emberek, a tündék, a törpök és a hobbitok. Céljuk az Egy Gyűrűt eljuttassák a Végzet Hegyéhez, hogy ott megsemmisítsék. A játékos maga Szarumán, akinek a célja, hogy a szövetségesek küldetését megakadályozza.

A szövetség különböző tagjaira, az egyes varázskövek másképp hathatnak. Az ellenség csak az utakon és csak előre halad, a céljukhoz, a Végez Hegyéhez, több különböző útvonalon is eljuthatnak.

A játékprogram célja

A játékban Szarumánt irányítjuk, a cél a szövetségesek legyőzése. A játék mechanikája a stratégiai védekezés köré öszpontosul. A folyamatosan áramló ellenségeket kell legyőzni, a játékos által lehelyezett tornyok és akadályok segítségével. Ha a szövetségesek egy egysége eléri a Végzet Hegyét, a játékos veszít, azonban ha sikerül megállítania a szövetségesek hadsereget, Sauron fog uralkodni középföldén.

2.2.3 Felhasználók

- Játékosok, akik élvezik Tolkien világát
- Kódoló csapat

2.2.4 Korlátozások

- Kevés idő áll rendelkezésre, ráadásul sok egyéb egyetemi teendő is van
- Kicsi, és közel azonosan elenyésző tapasztalattal rendelkező fejlesztők
- A csapat ritkán tud teljes létszámban maximális befektetéssel dolgozni
 - Nincs, aki készítene a játékhoz vizuális és audiális anyagot

2.2.5 Feltételezések, kapcsolatok

- A felhasználó képes kezelni a JDK-t
- Alapvető számítógépes játékismeret

2.3 Követelmények

2.3.1 Funkcionális követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Use-case	Komment
1	Új játékot lehet kezdeni	bemutatás	alapvető	Előírás	Új játék kezdése	
2	Ki lehet lépni a játékból	bemutatás	alapvető	Előírás	Kilépés a játékból	
3	Lehet menteni játék közben.	bemutatás	fontos	Csapat	Játékállás mentése	
4	Be lehet tölteni már lementett játékot.	bemutatás	fontos	Csapat	Játékállás betöltése	
5	Végredményt el lehet menteni.	bemutatás	opcionális	Csapat	Pontszám mentése	pontozási stratégia kidolgozása szükséges
6	Meg lehet nézni a ranglistát.	bemutatás	opcionális	Csapat	Scoreboard megtekintése	

2.3.2 Erőforrásokkal kapcsolatos követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
1	Java JDK a fejlesztői gépeken	konzultáció	alapvető	Csapat	
2	Java JRE a felhasználói gépeken	felhasználó által	alapvető	Csapat	terjesztéshez nem szükséges
3	R4J terem gépein fusson	konzultáció	fontos	Csapat	
4	Eclipse IDE	konzultáció	preferált	Csapat	más IDE is elfogadható

2.3.3 Átadással kapcsolatos követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
1	Hercules server	konzultáció	alapvető	Csapat	
2	.jar file futtatása	konzultáció	alapvető	Csapat	

2.3.4 Egyéb nem funkcionális követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
1	Hordozható	bemutatás	közepes	Csapat	
2	Optimalizált kód	gyűléseken	magas	Csapat	
3	Szép felület	bemutatás	alacsony	Csapat	

2.4 Lényeges use-case-ek

2.4.1 Use-case leírások

Use-case neve	Új játék kezdése
Rövid leírás	Új játékot lehet kezdeni
Aktorok	Játékos
Forgatókönyv	Játékos rákattint az új játék gombra.

Use-case neve	Játék
Rövid leírás	Maga a játékmenet.
Aktorok	Játékos
Forgatókönyv	Játékos az új játék vagy a játék betöltése gombra kattint.

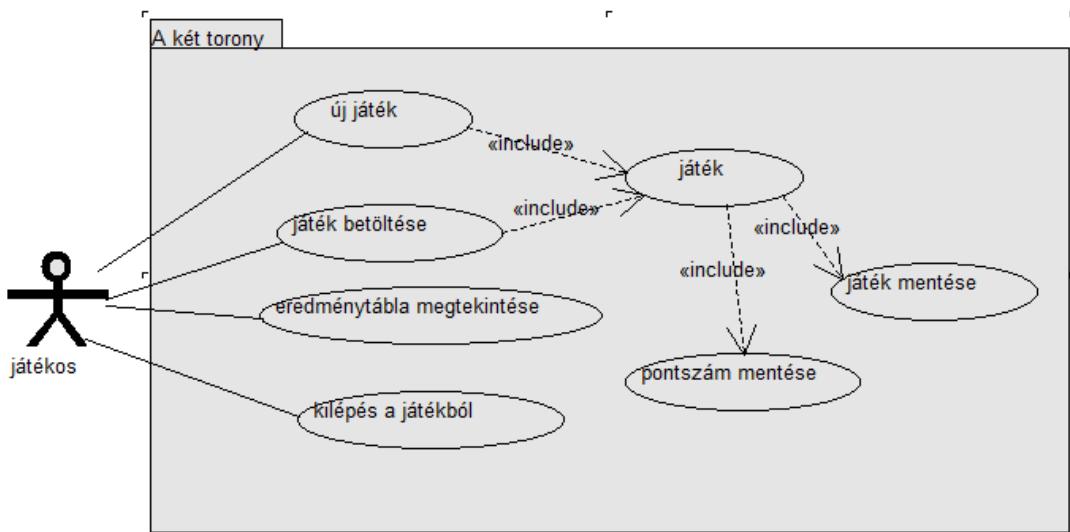
Use-case neve	Játék mentése
Rövid leírás	Játék aktuális állásának a lementése.
Aktorok	Játékos
Forgatókönyv	Játékos játék közben rákattint a mentés gombra.

Use-case neve	Játék betöltése
----------------------	-----------------

Rövid leírás	Játékos folytathat egy már megkezdett játékot.
Aktorok	Játékos
Forgatókönyv	A játékos a <i>játék betöltése</i> gombra kattint.

Use-case neve	Kilépés a játékból
Rövid leírás	A program bezárasa.
Aktorok	Játékos
Forgatókönyv	A játékos a <i>kilépés</i> gombra kattint.

2.4.2 Use-case diagram



2.5 Szótár

Pálya/Map: A játéktér amelyen a játszhatunk, ide helyezhetőek el a tárgyak és itt mozognak az egységek is.

Út: A pálya része, ezen mozognak az egységek valamint ide helyezhetőek el az akadályok is.

Mező: Azon része a pályának ahova tornyot építhetünk.

Torony: Általunk lerakható elem a játékban, egységek sebzésére képes, különböző típusai léteznek.

Akadály: A felhasználó által lerakható elem, az ellenséges egységek lassítására szolgál.

Szövetségesek/ellenségek: Az úton haladó, különböző egységek, melyek a kiinduló ponttól a cél felé haladnak, valamint életerővel rendelkeznek, melynek elfogytával megsemmisülnek.

Creep: A feladatkiírásban szövetségesnek nevezett egységek egy másik elnevezése.

Életerő/HP: minden szövetséges egység rendelkezik ezzel, ami a tornyuktól bekapott sebzéstől csökken, ha ez elfogy az egység elpusztul.

Indulópont: Kezdőpont, ez az a hely a pályán ahonnan elindulnak az úton az ellenséges egységek.

Végcél/Végzet Hegye: Ide tartanak az ellenséges egységek, ha akár csak egy egység is eléri ezt, a játék véget ér és vesztettünk.

Sebzés: Ez egy mértékegység amely azt hivatott mutatni, hogy a védelmi eszközök mekkora kárt tudnak okozni az egységek életerejében.

Játékos: A program használója, azaz a felhasználó.

Varázserő/Mana: A felhasználó az ellenségek elpusztításával szerezhet, majd elköltheti tornyok és akadályok építésére, varázskő vásárlására.

Haladási sebesség: minden ellenséges egység rendelkezik egy bizonyos sebességgel amivel a cél felé halad, ez egységenként eltérő lehet.

Varázskő: Ezzel az elemmel erősítheti meg a felhasználó a tornyait illetve akadályait.

Hatótávolság: A lerakható tornyoknak, azon tulajdonsága, ami megmutatja, hogy milyen messziről képesek elkezdeni sebzést.

Tüzelési gyakoriság: A tornyok azon tulajdonsága, hogy milyen időközönként képesek lövést ereszteni a hatótávolságon belül lévő ellenségekre.

Wave/horda: Az egy hullámban érkező creep-ek együttes elnevezése, úgy értsd: creep csapat.

Idő: Ez méri a játék kezdetétől eltelt időt, ez alapján vannak időzítve az egyes hordák indítási időpontjai és az eredmény kiszámításában is felhasználjuk

Highscore/eredmény: a játék megnyerését követően a játékos által elért összpontszám az idő és teljesítmény függvényében.

Ember: A végcél felé igyekvő ellenség egyik egysége, különböző tulajdonságokkal felruházva.

Tünde: A végcél felé igyekvő ellenség egyik egysége, különböző tulajdonságokkal felruházva.

Törp: A végcél felé igyekvő ellenség egyik egysége, különböző tulajdonságokkal felruházva.

Hobbit: A végcél felé igyekvő ellenség egyik egysége, különböző tulajdonságokkal felruházva.

Tile/ csempe: A pályát felépítő építőkockák. Az csempék azonos terüettel rendelkeznek, a csempe mezőt, vagy utat reprezentál.

2.6 Projekt terv

2.6.1 Erőforrások

Szoftver:

- Google Drive
- Skype
- Eclipse
- Github
- OpenAmeos
- JUnit
- Netbeans

Hardver:

- mindenki a saját számítógépét használja a fejlesztésre
- R4J gépe teszteléshez
- Hercules szerver

2.6.2 Csapattagok és feladatköreik

Név	Feladatkör
Iklódi Eszter	csapatvezető, a fő feladatok kiosztása, koordinálása, dokumentumok felülvizsgálata, vita esetén ő a végső döntéshozó
Molnár Bence	információk beszerzése az ismeretlen dolgokról, program felépítésével kapcsolatos kérdések, dokumentáció az értekezletekről, csapatvezető által kiosztott munka elvégzése
Németh Zsolt	játékfunkciók kidolgozása, egyes részek egyszerűsítése, játék típusával kapcsolatos korábbi tapasztalatok felhasználása, dokumentáció készítése, csapatvezető által kiosztott munka elvégzése
Radnai Balázs	elsősorban a kódolás és a grafikus ötletek megvalósítása
Srajner Ferenc	Konfigurációs management, implementálás, külföldi oldalakon való böngészés a projektet segítő dolgok után

2.6.3 Kommunikáció és megosztás

A fejlesztés számára fontos az ötletek megosztása. Ezt a leghatékonyabban élőben tudja megoldani a csapat. A heti rendszerességgel gyűlések ezt biztosítják. A gyűléseken meghozott döntések feljegyzésre kerülnek. Az ezen kívül felmerülő kérdéseket az interneten keresztül beszéli meg a csapat. Amennyiben valaki a csapatból nincs net közelben, telefonon kell felkeresni.

Skype: alkalmas gyűlés lebonyolítására, vagy kiegészítésére, ha valaki nem érhető el a csapatból

Facebook: amennyiben azonnali válasz kell, és nem fontos a dokumentálás

Google Drive: Az elvégzendő feladatok kiírása, a Napló helye és a dokumentáció szerkesztésére és tárolására alkalmas rendszer

A forrásfájlok megosztására a github-ot választotta a csapat, így mindenki könnyen hozzáférhet, és a verziókezelést is támogatja, ezzel könnyítve a munkát.

2.6.4 Véghajtás lépései

Skeleton: Célja bizonyítani, hogy a megtervezett modell a definiált feladatnak egy modellje. Az metódusoknak csak az interfésze definiált. Amennyiben meghívna egy metódust, az kiírja a nevét az ernalomra, majd meghívja a program futásához szükséges többi metódust. Elágazás esetén a tesztelő dönthet az irány kiválasztásáról. A szkeletonnak képesnek kell lennie a szekvencia diagramok és forgatókönyvek ellenőrzésére.

Prototípus: Már kész program, eltekintve a grafikus felülettől. minden funkciója helyesen működik. A metódusok már a végső algoritmusukkal vannak ellátva. Különös figyelmet kap az interfészlogika, a felépítés, és, hogy ez milyen módon teszi lehetővé az alkalmazás helyes futását.

Grafikus program: Hozzáadja a prototípushoz a grafikus felületet. A felület megalakításában nagyobb hangsúlyt kap a funkcionalitás, mint a megjelenés.

2.6.5 Határidők, és ütemezés

A határidőket és a konzultációk idejét a tanszék határozta meg, ezek betartása szigorúan fontos. Ezeknél kívül a csapatvezető tetszőleges mennyiséggű értekezletet tarthat.

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. 14.	
10	ápr. 22.	Prototípus - beadás
11	ápr. 28.	Grafikus felület specifikációja - beadás
12	máj. 5.	
13	máj. 12.	Grafikus változat - beadás
14	máj. 16.	Összefoglalás - beadás

	Dátum	Jelleg
1	febr. 12.	8:00 fakultatív eligazító
2	febr. 19.	konzultáció
3	febr. 26.	konzultáció
4	márc. 5.	konzultáció
5	márc. 12.	konzultáció
6	márc. 19.	konzultáció
7	márc. 26.	szkeleton bemutató - konzultáció
8	ápr. 2.	konzultáció
9	ápr. 9.	konzultáció
10	ápr. 16.	konzultáció
11	ápr. 23.	protó bemutató - konzultáció
12	ápr. 30.	munkaszünet
13	máj. 7.	konzultáció
14	máj. 14.	grafikus bemutató - konzultáció

2.7. Napló

Kezdet	Időtartam	Résznevők	Leírás
2014.02.19. 8:30	0,5 óra	Molnár Németh Srajner Iklódi	Értekezlet. Alapkoncepció és követelmények átbeszélése, ötletelés
2014.02.20.18:00	2 óra	Iklódi	use-case elemek meghatározása, leírása, dokumentum töltése
2014.02.20.18:00	2 óra	Németh	Funkciók (2.2.2)
2014.02.20. 18:15	2 óra	Srajner	dokumentáció bővítése, Projekt terv, Követelmények (>2.3.1)
2014. február 21.	1,5 óra	Molnár	dokumentáció áttekintése, formázása, funkciók (2.2.2) kiegészítése, feladatkörök, 2.1.*
2014. február 21.	0,5 óra	Radnai	áttekintés, appróbb formai módosítások.
2014.02.22. 10.00	1 óra	Németh	Szótár (2.5)
2014.02.22. 10:00	1 óra	Molnár	Szótár (2.5)
2014.02.22. 10:30	1 óra	Srajner	Projekt terv kiegészítés (2.6.3 +) (2.1.3)
2014.02.22. 12:45	2,5 óra	Iklódi	dokumentáció felülvizsgálata

Analízis modell kidolgozása

3.1 *Objektum katalógus*

Ez a fejezet arról szól, hogy az objektumokról rövid, informális leírást adjon. A következő fejezetben foglalkozunk az örökléssel és interfészkekkel. Ebben a fejezetben ezekre nem térünk ki.

3.1.1 Creep

Ez az osztály testesíti meg az utakon haladó szövetségeseket. Felelős az egységek mozgatásában, és a hozzá közeli tornyoknak is ő az, aki jelzi, hogy löhet. A torony tárolja a szövetséges típusát, így tudja majd eldönteni hogy kire célozzon. és így tudja kiszámolni, a lassítások és sebzések mértékét is.

3.1.2 Dwarf, Elf, Human, Hafling

A creep osztály által megtettesített szövetségesek megfelelő kirajzolása miatt van rájuk szükség, illetve ha a játék fejlesztése során valamilyen működésbeli különbséget szeretnénk a fajok között, akkor azt ezen objektumok segítségével könnyebben kivitelezhetjük.

3.1.3 Game

Központi objektum. Itt fut a program fő szála, ami a többi, játékban levő objektumot létrehozza, majd a játék során a kezelő függvényeit meghívja.

3.1.4 MagicStone

A fejlesztésekért felelős varázskövek adatait tárolja el. Egy ilyen objektum példány egy adott típushoz tartozik, ez alapján fejleszthető majd egy torony vagy egy akadály. Tárolja még azt is, hogy az adott típusból hány darab vásárolható.

3.1.5 Projectile

A torony által létrehozott lövedék. A mozgásáért felelős, tehát követi a célpontját, és ha odaér, akkor sebzi.

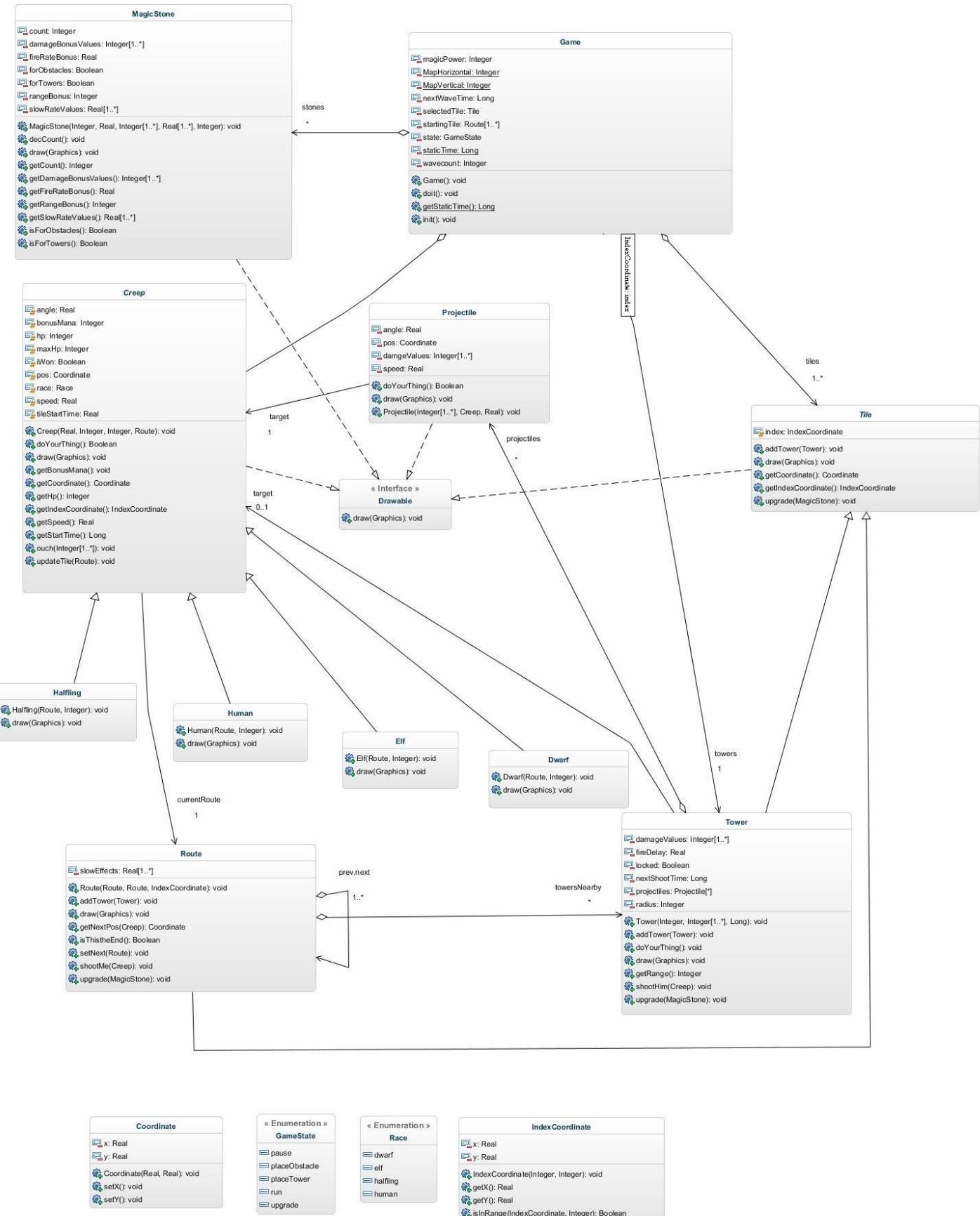
3.1.6 Route

Azok a mezők a Pályán melyeken a szövetségesek haladnak. Egy mező egy példány. Tudja hogy az útvonalban melyik mező a következő illetve az előző, és ez alapján a szövetségesek útvonalának számítását ő végzi.

3.1.7 Tower

A tornyok működéséért felelős, tehát az éppen aktuális célpont felé haladó Projectile objektum létrehozása és az aktuális célpont kiválasztása az elérhető szövetségesek közül a feladata.

3.2 Statikus struktúra diagramok



A négy különálló osztály segédosztályként, illetve állapotokhoz definiált enumként szerepel a modellben.

3.3 Osztályok leírása

3.3.1 Creep

Felelősség

Az ellenségek absztrakt ōsosztálya. Attribútumai leírják az ellenség specifikus tulajdonságait, továbbá. A creep mindenkor tudja, hogy melyik Route-on tartózkodik. Fontos, hogy a tornyokhoz a creepek regisztrálják be magukat célpontként, méghozzá a currentRoute attribútumukon shootMe metódusán keresztül. A creep felelős a saját maga mozgatásáért, és sebzés hatására csökkenti az életerejét.

Ősosztályok

Interfészek

Drawable

Attribútumok

- **double angle:** azt a szöget adja meg radiánban, amennyivel az egység elfordul.
- **int bonusMana:** az az érték, ami az egység megölésekor a játékos varázserejéhez adódik.
- **Route currentRoute:** Az a Route-mező, amin az egység jelenleg tartózkodik
- **int hp:** a szövetséges életereje
- **Boolean iWon:** ha az egység az utolsó Route-mezőre kerül, akkor true.
- **int maxHp:** az egység kiinduláskor meglévő életereje.
- **Coordinate pos:** Az egység pozíciója a képernyőn, pixel egységekben, de nem kerekítve.
- **Race race:** az egység faja.
- **double speed:** megmondja, hogy az egység egy másodperc alatt mennyi Route-mezőn halad végig, ha nincs módosító tényező.
- **double tileStartTime:** Az időt tárolja, amikor az egység a currentRoute-ra került. Ahhoz hogy számoljunk vele, ki kell vonni belőle a Game objektum staticTime attribútumát.

Metódusok

- **Creep(double speed, int maxHp, int bonusMana, Route spawnPoint):** konstruktur, beállítja az egység életterejét és alapsebességét.
- **boolean doYourThing():** ez a függvény regisztrálja be magát a tornyokhoz, amelyek elérik. Visszatérési értéke true, ha elfogyott a hp-ja, illetve ha a játékos beérte a célba
- **int getBonusMana():** visszaadja, hogy ha megölték mennyi varázserőt kap a játékos érte
- **Coordinate getCoordinate():** visszaadja a pos attribútumot
- **int getHp():** visszaadja a hp attribútum értékét
- **IndexCoordinate getIndexCoordinate():** visszaadja a currentRoute index-koordinátáját
- **double getSpeed():** visszaadja az egység sebességét
- **long int getStartTime():** megmondja, hogy mennyi ideje tartózkodik a currentRoute-on
- **void ouch(int [] damageValues):** kiolvassa a fajtájához tartozó sebzési értéket, és csökkenti a hp-ját

- **void updateTile(Route t):** frissíti, hogy melyik Route-mezőn van éppen

3.3.2 Game

Felelősség

Ez az osztály a játék magja. Itt hozzuk létre és tároljuk el a legfontosabb objektumokat, illetve a játék fő vezérlése is innen történik. Ebből az osztályból egy példány keletkezik a játékban. Itt tároljuk el a játékoshoz tartozó adatokat is, mint például, hogy mennyi varázserővel rendelkezik. Innen indítjuk a wave-eket, és innen vezéreljük a lövéseket is. A Game maga egy állapotgép, a vásárlások, fejlesztések illetve a játékból való ki-be lépések kezelését állapotautomataként valósítja meg.

Ősosztályok

Interfészek

Attribútumok

- **static int MapHorizontal:** a pálya Tile-jait tartalmazó tömb vízszintes mérete.(package elérhetőségű)
- **static int MapVertical:** a pálya Tile-jait tartalmazó tömb függőleges mérete.(package elérhetőségű)
- **ArrayList<Creep> creeps:** a pályán jelenlevő Szövetségeseket tárolja, halálukkor kitörli belőle
- **int magicPower:** a játékos varázsereje
- **long nextWaveTime:** megmondja, hogy mennyi idő kell még a következő wave-ig
- **Tile selectedTile:** tárolja a játékos által kijelölt mezőt
- **ArrayList<Route> startingTile:** azokat a Route-mezőket tartalmazza, melyekről a szövetségesek indulhatnak.
- **GameState state:** a game állapotát jelzi
- **static long staticTime:** abszolút viszonyítási idő az egész játékban. Ahhoz szükséges, hogy amikor a játék áll, akkor az idő alapján számoló függvények az újraindításkor ne észleljék hogy eltelt idő.
- **MagicStone stones[]:** eltárolja e különböző hatású varázsköveket
- **public static Tile[][] tiles:** minden útnak és toronynak a lehetséges helyeit tartalmazza.
- **HashMap<IndexCoordinate, Tower> towers:** ebben tároljuk a tornyokat, koordináta szerinti kulcsokkal
- **int wavecount:** számon tartjuk, hogy hányadik wave-nél járunk

Metódusok

- **Game():** konstruktur, létrehozza a tárolókat
- **void doit():** legfőbb vezérlő ciklus, a játék alatt lényegében ez fut folyamatosan
- **init():** inicializálja a játék változóit, létrehozza az utat ahol a creepek haladnak
- **static long getStaticTime():** le lehet kérdezni az abszolút időt

3.3.3 Halfling, Human, Elf, Dwarf

Felelősség

Mindahányan a Creep-től örökölnek. Amiben különböznek egymástól, az a konstruktorkuk és a draw függvényük, amely minden egységet másképp jelenít meg.

A konstruktur megkapja a kezdőpontot és azt, hogy hányadik wave-nél járunk. Ezekből az adatokból kiszámítja az attribútumainak a kezdőértékeit, majd meghívja az ösosztály konstruktörét. A felelőssége és a szerepkör ugyanaz, mint a szülőnél, a különböző fajok csak a szülő speciális leszármazottai.

Ősosztályok

Creep

Interfészek

Drawable

Attribútumok

- minden attribútuma megegyezik a Creep attribútumaival

Metódusok

- **Halfling(Route spawnPoint, int waveCount):**
- **Hobbit(Route spawnPoint, int waveCount):**
- **Dwarf(Route spawnPoint, int waveCount):**
- **Elf(Route spawnPoint, int waveCount):**
- **Draw(Graphics g):** kirajzol majd az ablakunkra

3.3.4 IndexCoordinate

Felelősség

Minden Tile-hoz tartozik egy index, ami a helyzetét határozza meg a pálya mezőinek tömbjében, illetve könnyen számítható belőle szomszédosság illetve távolság.

Ősosztályok

-

Interfészek

-

Attribútumok

- **int x:** x dimenziójú index
- **int y:** y dimenziójú index

Metódusok

- **boolean isInRange(IndexCoordinate idx, int range)**: megadja, hogy egy adott indexkoordináta range-en belül találhető-e tőle (hány lépésből lehet eljutni oda)
- **IndexCoordinate(int xx, int yy)**: konstruktor, alapérték beállítása

3.3.5 MagicStone

Felelősség

Varázskövek, ennek az osztálynak a segítségével történnek a fejlesztések. A varázskő számon tartja, hogy milyen fejlesztésekre képes, illetve hogy melyik creep-re milyen mértékű hatást gyakorol. A MagicStone osztály egy példánya lényegében az azonos tulajdonságú varázskövek egy csoportját valósítja meg. Egy varázskő felhasználásakor csökken a példány számlálója, ezzel számon tartva, hogy hánnyal elérhető van még belőle. Tárolja még hogy a fejlesztés toronyra illetve akadályra vonatkozik-e. Ez alapján lehet majd egy adott torony vagy akadály fejlesztés során eldönthetni hogy megjelenik-e a listában.

Ősosztályok

Interfészek

Drawable

Attribútumok

- **int count**: megmondja, hogy a játékos még maximum hányat tud felhasználni ebből a fajtából.
- **int[] damageBonusValues**: itt tárolja, hogy melyik creep-et mennyivel sebzi.
- **double fireRateBonus**: ennyivel növeli a torony tüzelési gyakoriságát.
- **boolean forObstacles**: megmondja hogy az adott varázskő felhasználható-e akadály fejlesztésére.
- **boolean forTowers**: megmondja hogy az adott varázskő felhasználható-e torony fejlesztésére.
- **int rangeBonus**: ennyivel növeli a torony hatósugarát.
- **double[] slowRateValues**: itt tárolja, hogy melyik creep-et mennyivel lassítja.

Metódusok

- **MagicStone(int rangeBonus, double fireRateBonus, int[] DamageBonusValues, double[] slowRateValues, int count, boolean forTowers, boolean forObstacles)**: konstruktor, melyben beállítjuk az attribútumok kezdőértékét
- **void decCount()**: eggyel csökkenti a count értékét
- **int getCount()**: visszaadja a conunt értékét
- **int[] getDamageBonusValues()**: megadja, hogy melyik creep-et mennyivel sebzi
- **double getFireRateBonus()**: megadja a torony tüzelési gyakoriságát
- **boolean getForObstacles()**: igazzal tér vissza, ha akadályra használható a kő
- **boolean getforTowers()**: igazzal tér vissza, ha tornyokra használható a kő
- **int getRangeBonus()**: megadja a torony lövési hatósugarát
- **double[] getSlowRateValues()**: megadja, hogy melyik creep-et mennyivel lassítja

3.3.6 Projectile

Felelősség

Ez az osztály valósítja meg a lövedéket. A torony hozza létre, azonban létrehozás után actor-ként viselkedik. Felelős a saját maga mozgatásáért, és ő hívja meg a creep sebző függvényét is. Ha elérte célpontját szól a toronynak, hogy befejezte küldetését.

Ősosztályok

Interfészek

Drawable

Attribútumok

- **double angle:** a lövedék sebességének az iránya
- **int[] damageValues:** eltárolja, hogy melyik fajt milyen mértékben tudja sebezni
- **Coordinate pos:** az aktuális pozícióját tárolja
- **double speed:** a lövedék sebességének a nagysága
- **Creep target:** a lövedék célpontja, öt fogja követni, majd megsebezni

Metódusok

- **Projectile(Creep target, int[] damageValues, double speed):** konstruktur, ez a függvény inicializálja a lövedéket
- **boolean doYourThing():** ez a függvény felelős a creep sebzéséért, és önmaga mozgatásáért. Visszatérési értéke true, ha elérte a creepet.
- **void Draw(Graphics g):** kirajzolja a lövedéket

3.3.7 Route

Felelősség

Azokat a mezőket testesíti meg, melyeken a szövetségesek mozogni tudnak. feladata a rajta áthaladó Creepeknek meghatározni az útvonalukat, illetve rajta keresztül történik a tornyok értesítése a közelben levő egységekről.

Ősosztályok

Tile

Interfészek

Drawable

Attribútumok

- **Route next:** a következő Route-mezőre mutat (ide küldi őket tovább)
- **Route prev:** az előző Route-mezőre mutat(innen jönnek a creepek erre)
- **double[] slowEffects:** megadja, hogy az adott fajú szövetségesekre mekkora lassító hatással van.
- **ArrayList<Tower> towersNearby:** azokat a tornyokat tárolja, akiknek beleesik a hatósugarába

Metódusok

- **Route(Route prev, IndexCoordinate idx):** konstruktor, megadja hogy melyik az előző Route-mező, és az elhelyezkedését a pályán index-koordinátában.
- **void addTower(Tower t):** egy torony felvétele a towersNeabyListába.
- **void draw(Graphics g):** kirajzolja majd a monitorra az akadályokat.
- **Coordinate getNextPos(Creep it):** visszaadja egy adott szövetséges helyzetét a jelenlegi időpillanatban
- **boolean isThisTheEnd():** megmondja hogy ez egy utolsó Mező-e a route
- **void setNext(Route nxt):** beállítja a next attribútumot
- **void shootMe(Creep it):** egy Creep tud jelezni vele, hogy itt van, így a Route továbbítja a kérést azoknak a tornyoknak akik ezt az adott mezőt elérík.(towersNearby)
- **upgrade(MagicStone ms):** végre hajtja a paraméter által meghatározott változtatásokat a tornyon: (megszorozza a lassítás mértékét a jelenlegi értékkel minden egyes fajhoz egyenként)

3.3.8 Tile

Felelősség

A pálya alkotóelemei. Lehet őket fejleszteni, illetve be lehet regisztrálni hozzájuk Tower-eket. A Gameben tároljuk őket. A Tile tudja magáról, hogy milyen indexen helyezkedik el ebben a tömbben, illetve ki tudja számolni a koordinátáját is.

Ősosztályok

Interfészek

Drawable

Attribútumok

- **protected IndexCoordinate index:** a Game tiles tárolójában elfoglalt indexe

Metódusok

- **void addTower(Tower t):** absztrakt metódus, nincs implementálva
- **void draw(Graphics g):** absztrakt metódus, nincs implementálva
- **Coordinate getCoordinate():** kiszámolja az index alapján a koordinátáját
- **IndexCoordinate getIndexCoordinate():** visszaadja az index attribútumot
- **void upgrade(MagicStone ko):** absztrakt metódus, nincs implementálva

3.3.9 Tower

Felelősség

A tornyokat megvalósító osztály, a Tile leszármazottja. Állapotgépként viselkedik. Lőni akkor tud, ha van kit, és a lövési gyakoriság attribútuma is engedélyezi. Azt, hogy ki lehet az aktuális célpontja, azt kívülről kapja meg, ő maga ezek közül választja ki a ténylegeset. A torony tud lőni, de a lövő függvény vezérlése kívülről történik. Ha van érvényes célpont, és lőhet is, akkor a torony létrehoz egy lövedéket. A torony szólítja fel a lövedékeket mozgásra.

Ősosztályok

Tile

Interfészek

Drawable

Attribútumok

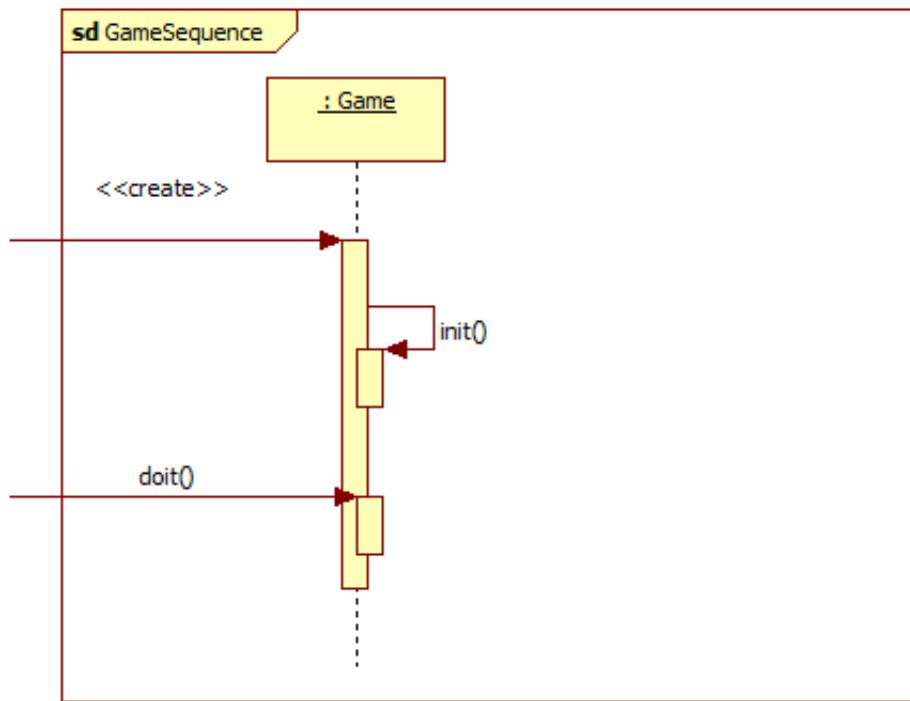
- **int [] damageValues:** eltárolha, hogy az egyes fajokat milyen mértékben sebzi
- **double fireDelay:** tüzelési gyakoriságot
- **boolean locked:** igaz, ha a torony már eldöntötté hogy kit lő, és még ténylegesen tudja is lőni azt.
- **long int nextShotTime:** ennyi idő van még a következő lövésig
- **ArrayList<Projectiele> projectiles:** a kilött, még nem célbaért Projectileokat tárolja
- **int radius:** torony hatósugara
- **Creep target:** ezt az ellenséget lövi

Metódusok

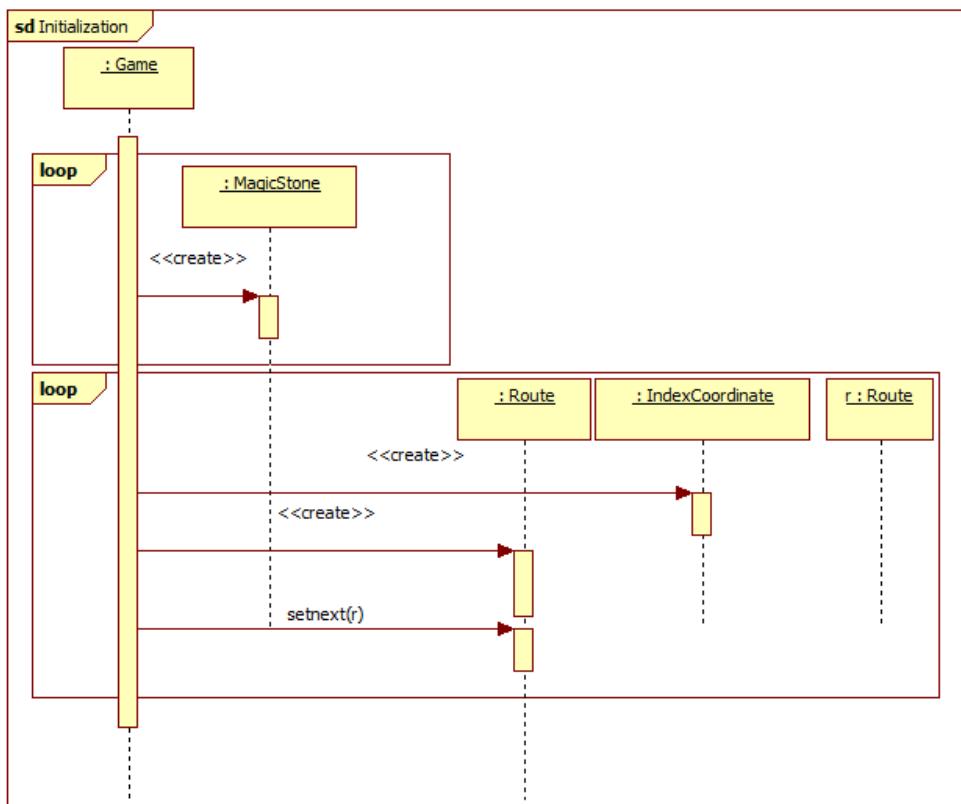
- **Tower(int radius, int[] damageValues, long int fireDelay):** konstruktor, beállítja a toronyhoz tartozó alap értékeket.
- **void doYourThing():** ez a függvény felelős a lövésért, illetve a projectile tömb kezeléséért.
- **draw(Graphics g):** kirajzolja a tornyot a képernyőre
- **void shootHim(Creep it):** a Torony eldönti hogy a paraméterben kapott Creep ideálisabb célpont-e, mint az eddigi legjobb(az összes elérhető Creepból befut egy ilyen kérés, így egy maximumkeresést valósít meg tehát) Ha locked igaz, akkor ellenőrzi hogy még tudja-e lőni a célpontot, és ha nem, akkor hamisra állítja. ha még elérhető, akkor nem csinál semmit a függvény.
- **int getRange():** vissza adja a torony hatósugarát
- **void addTower(Tower t):** Tile-tól öröklött függvény, a függvény törzse üres
- **void upgrade(MagicStone stone):** a stone rávonatkozó tulajdonságaival fejleszti magát

3.4 Szekvencia diagramok

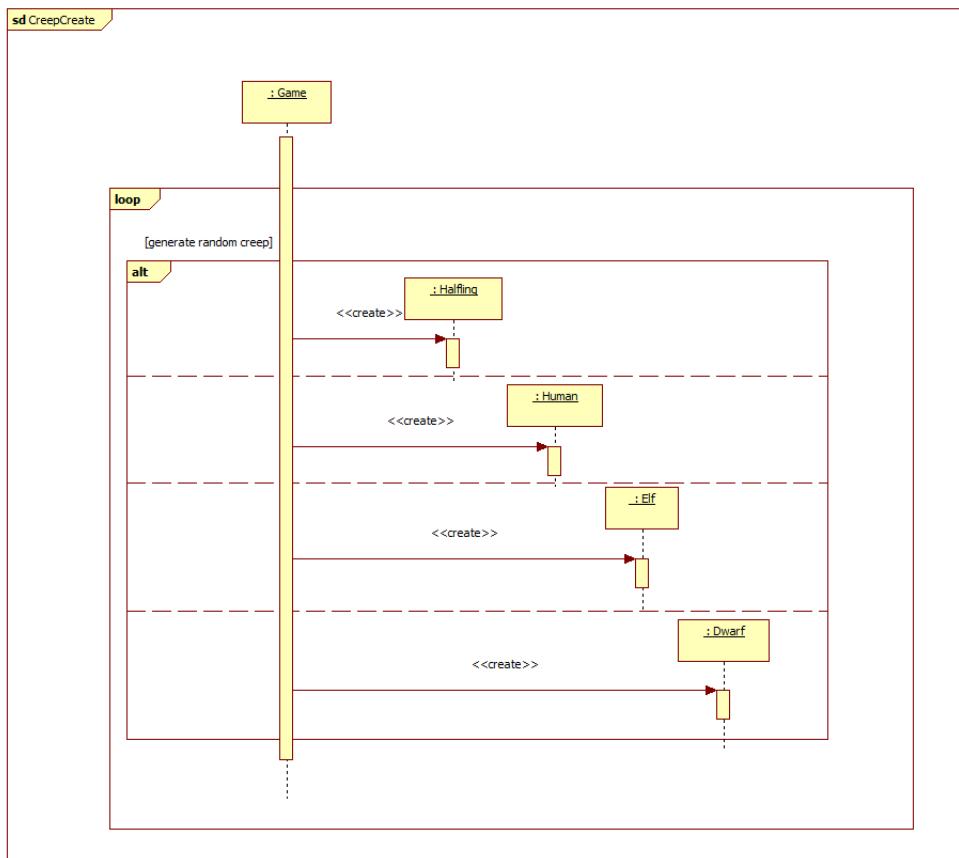
3.4.1 Játék szekvenciája



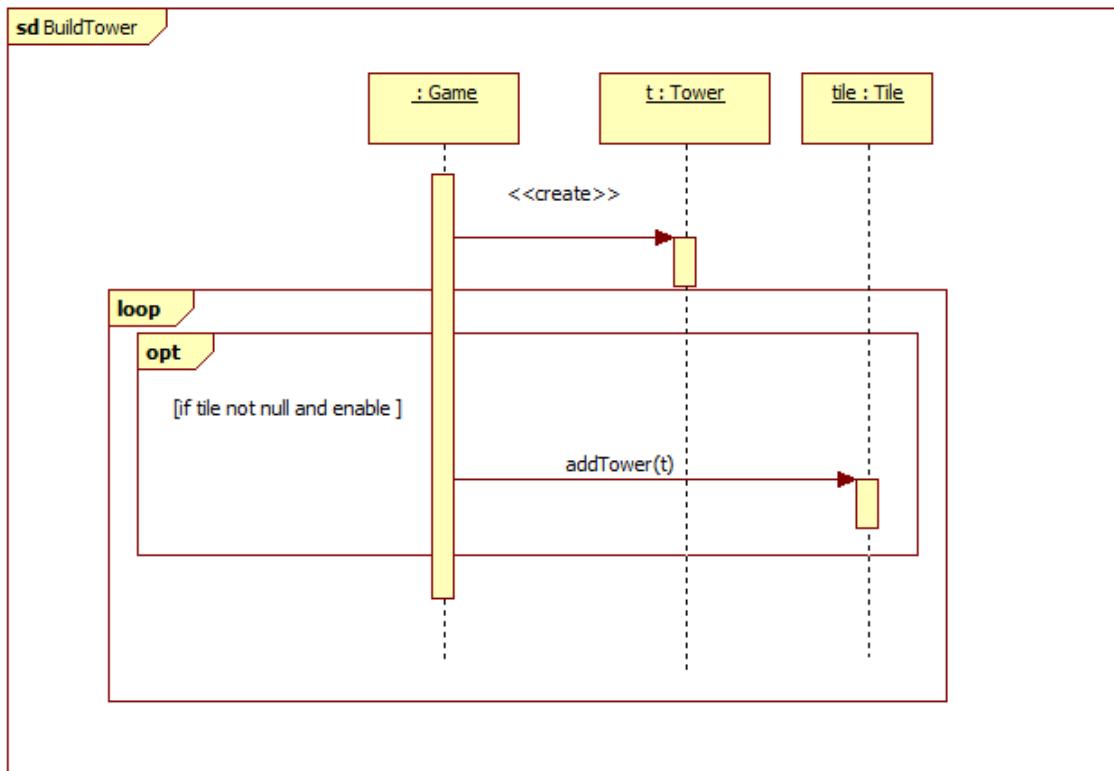
3.4.2 Inicializálás



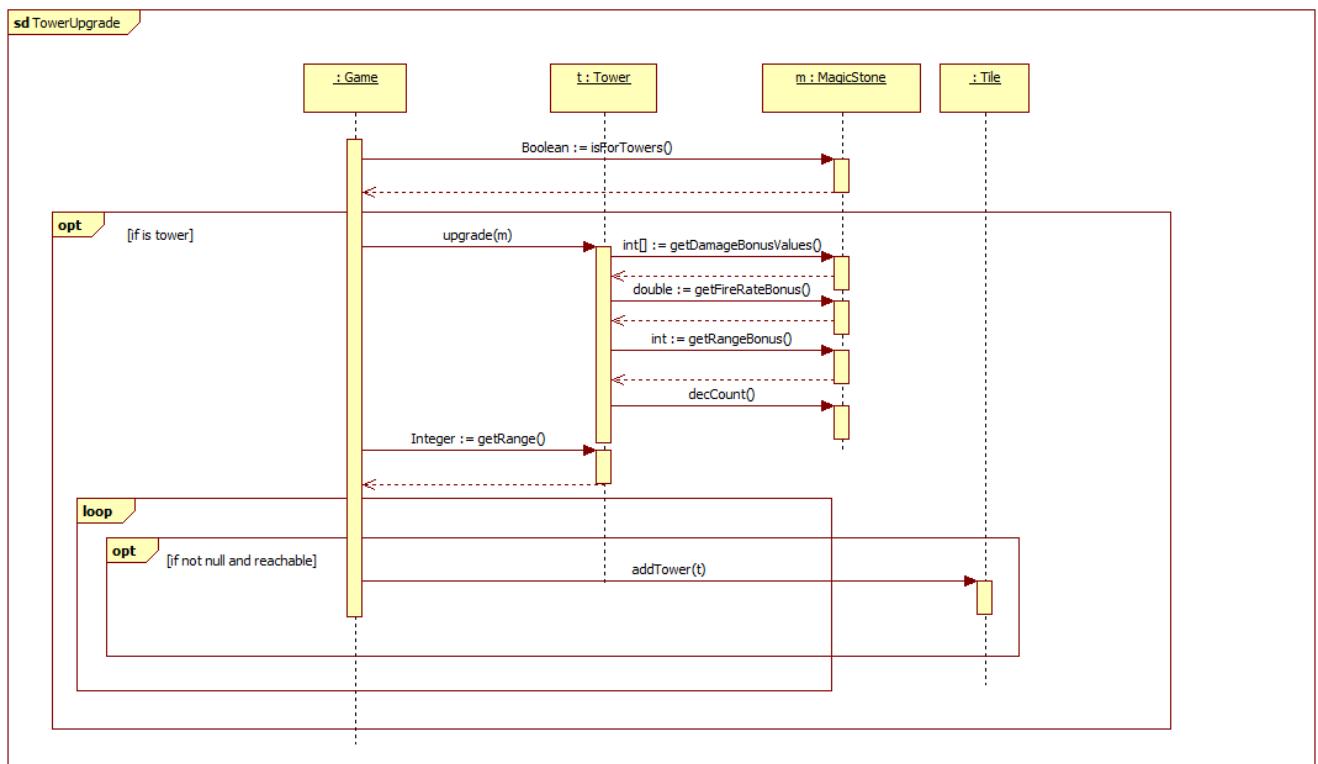
3.4.3 Szövetséges létrehozása



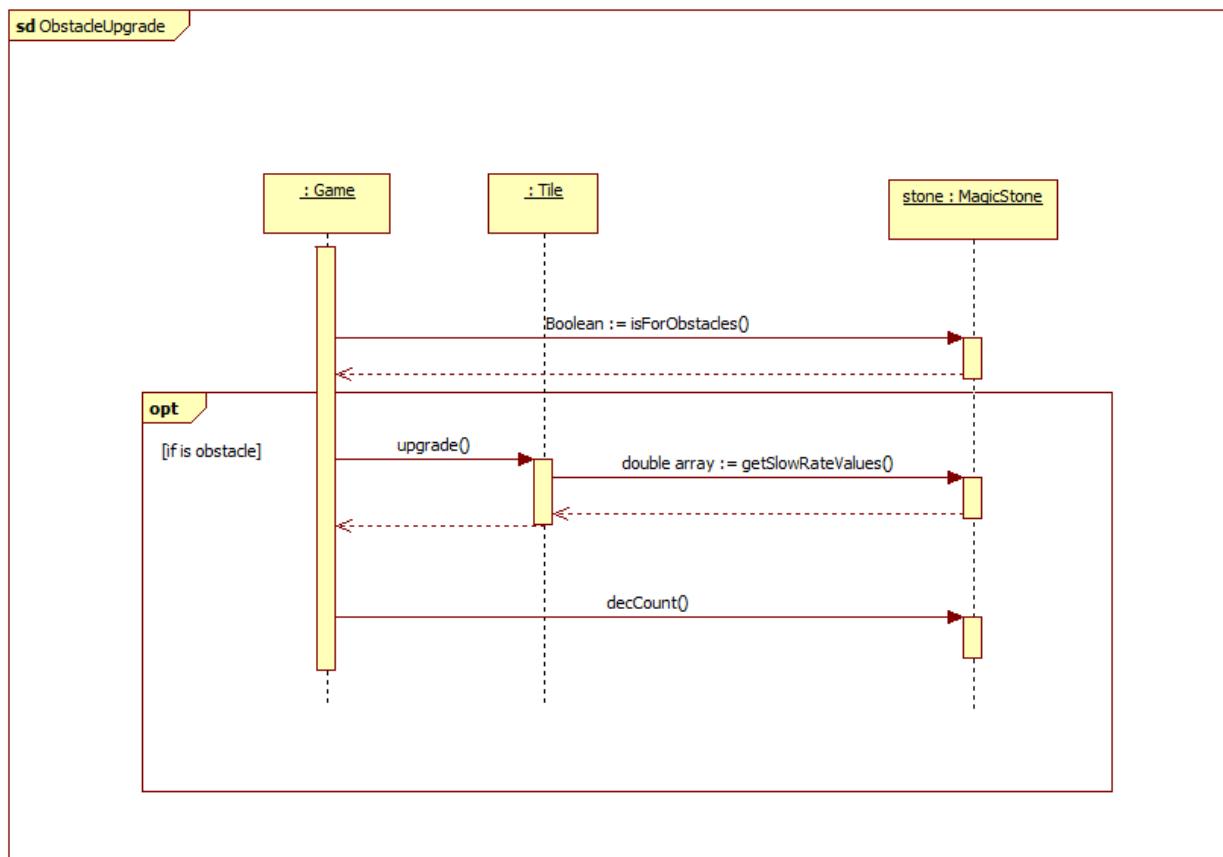
3.4.4 Toronyépítés



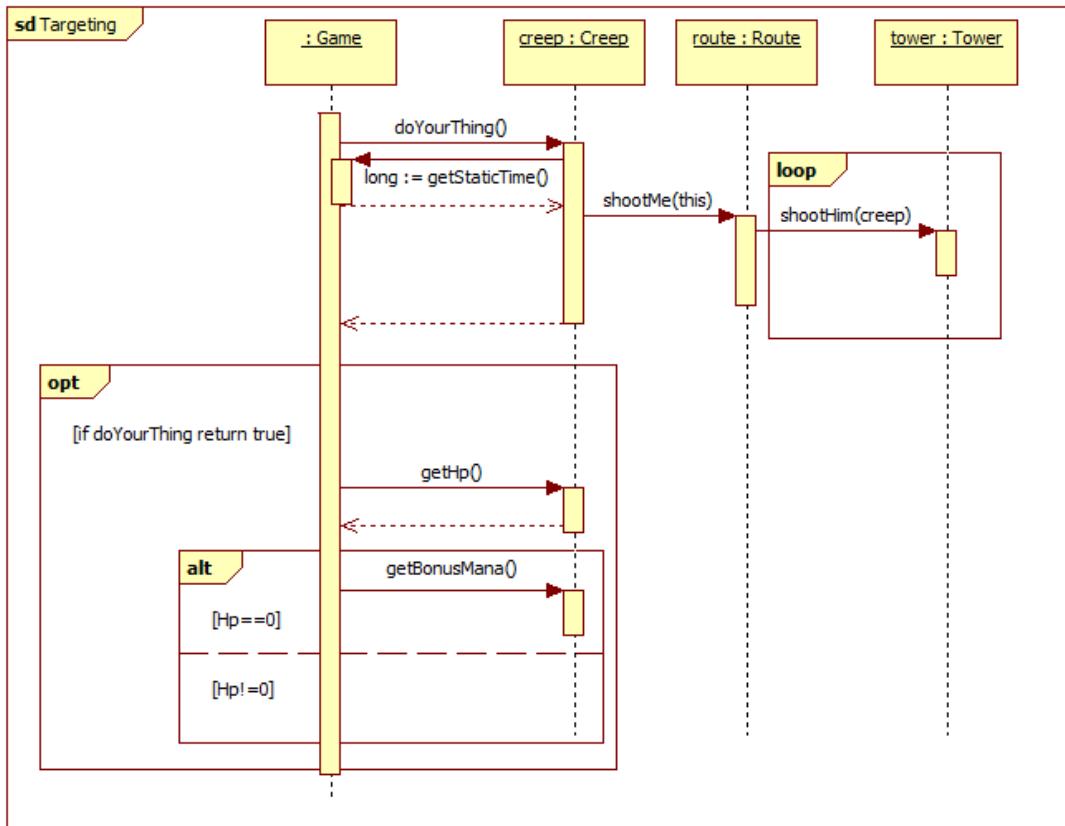
3.4.5 Toronyfejlesztés



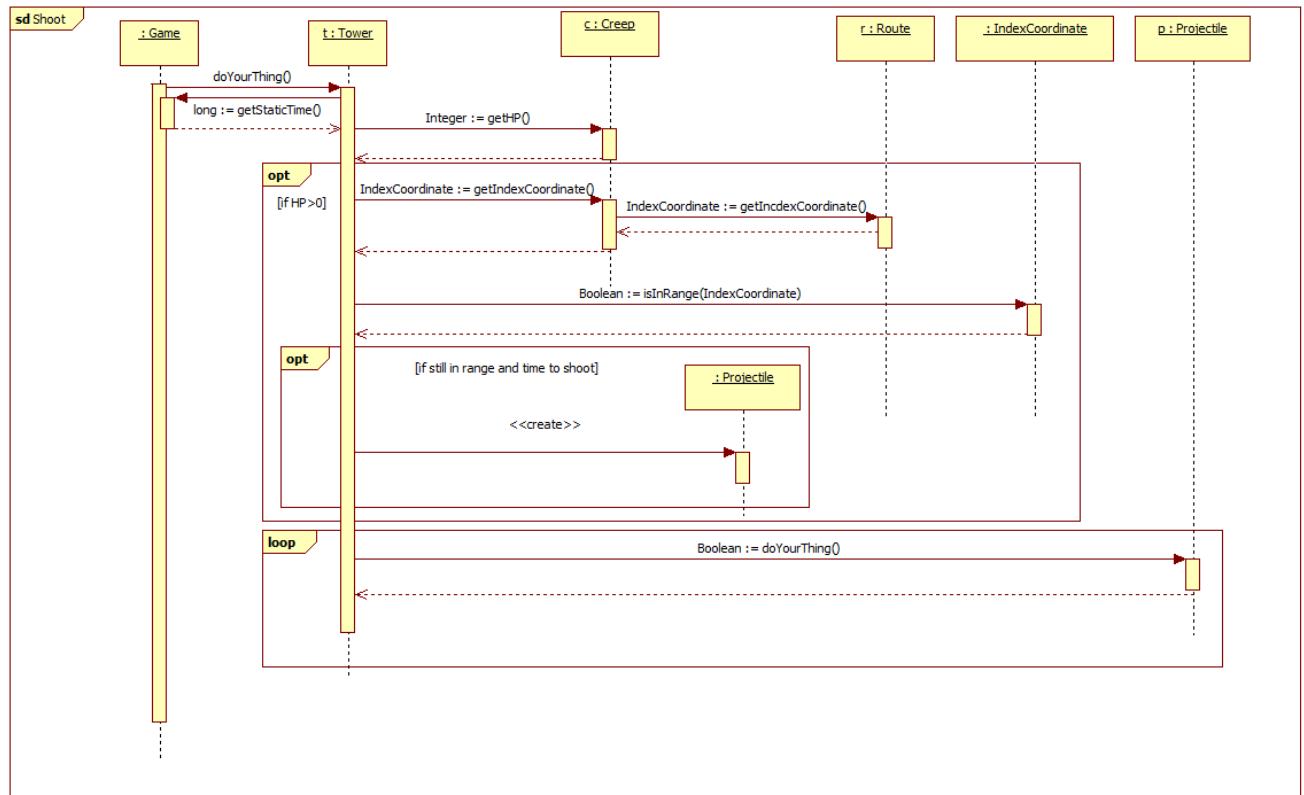
3.4.6 Akadály fejlesztés



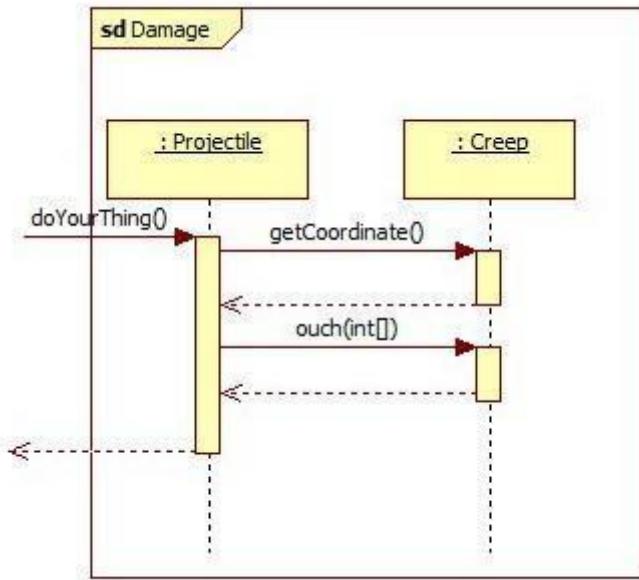
3.4.7 Célzás



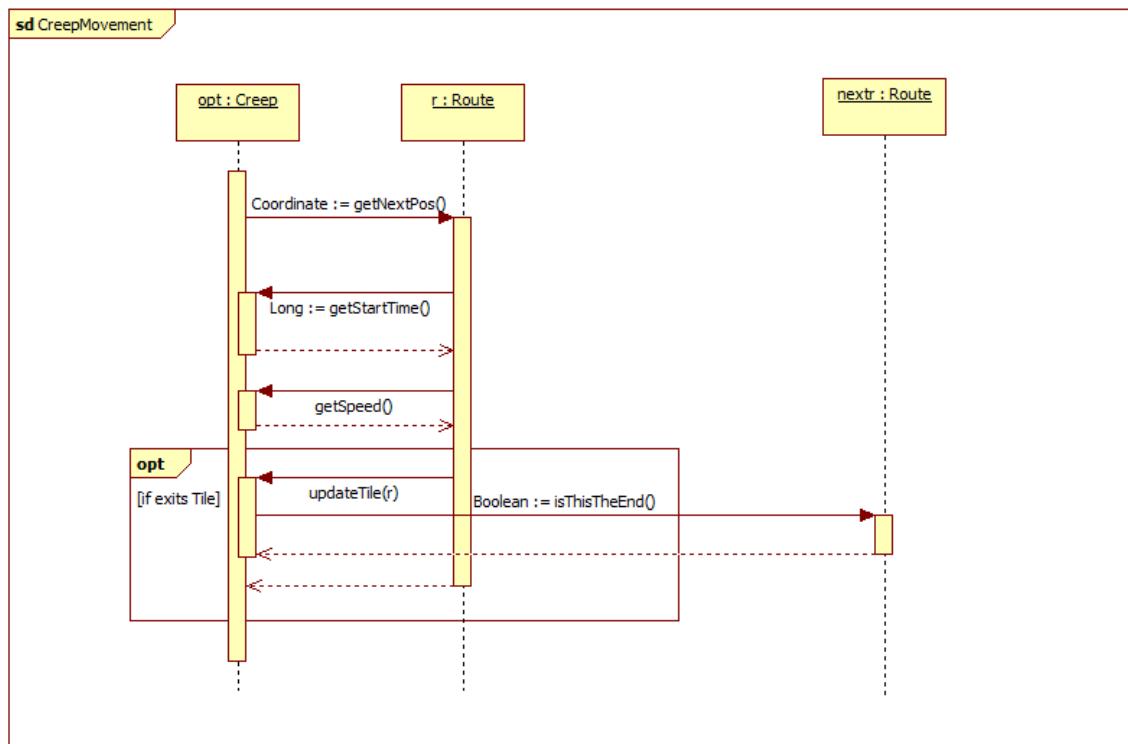
3.4.8 Lövés



3.4.9 Sebzés



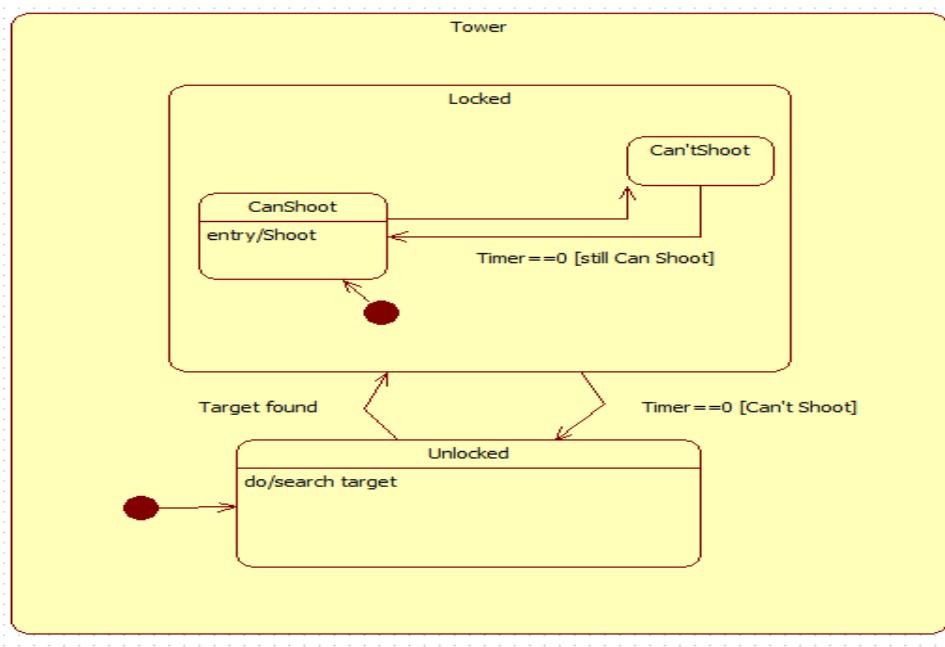
3.4.10 Szövetséges mozgatása



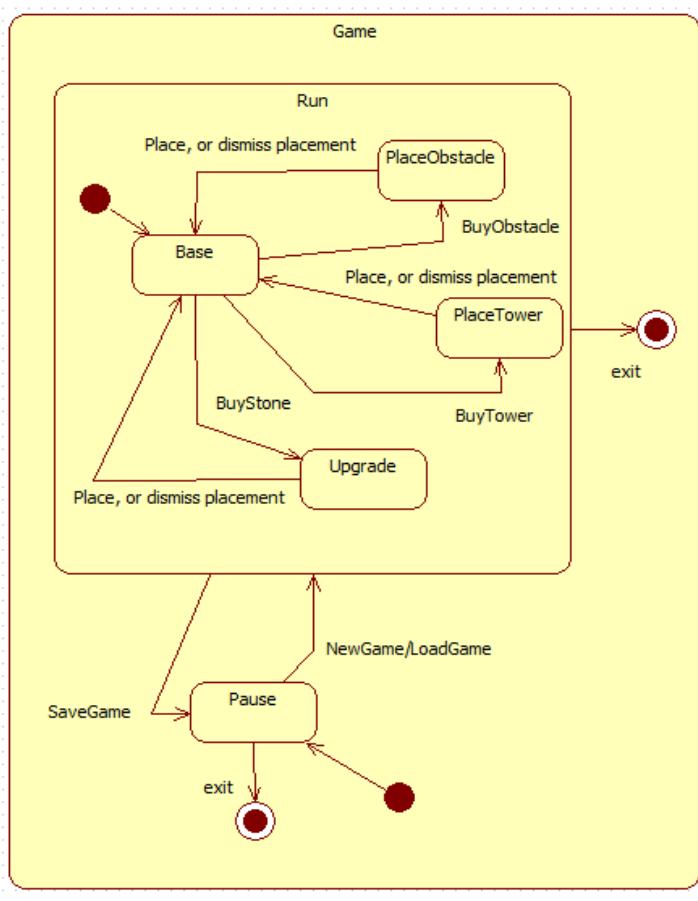
A szekvenciadiagramokon a Drawable interfészről megvalósító osztályok draw függvényeit nem tüntettük fel, mert azok meghívása a grafikus vezérlés dolga, amiről csak a későbbi fejezetekben lesz szó.

3.5 State-chartok

3.5.1 Torony állapota



3.5.2 Game állapot



3.6 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2014.02.26. 8:00	2 óra	Iklodi Molnár Németh Radnai Srajner	Konzultáció a konzulenssel, analízis modell fő vázának egyeztetése
2014.02.27. 13:00	2 óra	Iklodi Molnár Németh Radnai Srajner	gyűlés, objektumok összegyűjtése
2014. 02. 27. 15:00	2 óra	Srajner Radnai Iklodi	osztálydiagram első verziójának megrajzolása, és hozzá egyes funkciók megtervezése
2014.03.01. 15:00	5 óra	Iklodi Molnár Németh Radnai	Osztálydiagramm átbeszélése és megalkotása
2014.03.01. 20:00	5 óra	Iklodi Molnár Németh Srajner	Szekvenciadiagrammok elkészítése, osztálydiagramm és szekvencia diagramm összehangolása
2014.03.02. 12:00	0,5 óra	Srajner	State Chartok digitális megjelenítése
2014.03.02. 11:00	3 óra	Iklodi	State Chartok megrajzolása, dokumentumok szerkesztése
2014.03.02. 11.00	2 óra	Németh	Szekvenciadiagrammok, state-chartok beszerkesztése, dokumentumok ellenőrzése

2014.03.02. 17:00	5,5 óra	Iklódi	Dokumentum szerkesztése, szekvencia diagramok, osztálydiagram módosítása
2013.03.02. 17:00	5 óra	Radnai	dokumentum átvizsgálása, hiányok pótlása, szekvencia diagramok lefedettségének vizsgálata
2014.03.02. 18:00	4,5 óra	Molnár Srajner	Dokumentum szerkesztése, osztálydiagram és szekvencia diagram készítése

4. Analízis modell kidolgozása 2

4.1 Objektum katalógus

Ez a fejezet arról szól, hogy az objektumokról rövid, informális leírást adjon. A következő fejezetben foglalkozunk az örökléssel és interfészekkel. Ebben a fejezetben ezekre nem térünk ki.

3.1.1 Creep

Ez az osztály testesíti meg az utakon haladó szövetségeseket. Felelős az egységek mozgatásában, és a hozzá közeli tornyoknak is ő az, aki jelzi, hogy lőhet. A torony tárolja a szövetséges típusát, így tudja majd eldönteni hogy kire célozzon. és így tudja kiszámolni a lassítások és sebzések mértékét is.

3.1.2 Dwarf, Elf, Human, Hafling

A creep osztály által megtestesített szövetségesek megfelelő inicializálása miatt van rájuk szükség, illetve ha a játék fejlesztése során valamilyen működésbeli különbséget szeretnénk a fajok között, akkor azt ezen objektumok segítségével könnyebben kivitelezhetjük.

3.1.3 Game

Központi objektum. Itt fut a program fő szála, ami a többi, játékban levő objektumot létrehozza, majd a játék során a kezelő függvényeit meghívja.

3.1.4 MagicStone

A fejlesztésekért felelős varázskövek adatait tárolja el. Egy ilyen objektum példány egy adott típushoz tartozik, ez alapján fejleszthető majd egy torony vagy egy akadály. Tárolja még azt is, hogy az adott típusból hány darab vásárolható.

3.1.5 Projectile

A torony által létrehozott lövedék. A mozgásáért felelős, tehát követi a célpontját, és ha odaér, akkor sebzi.

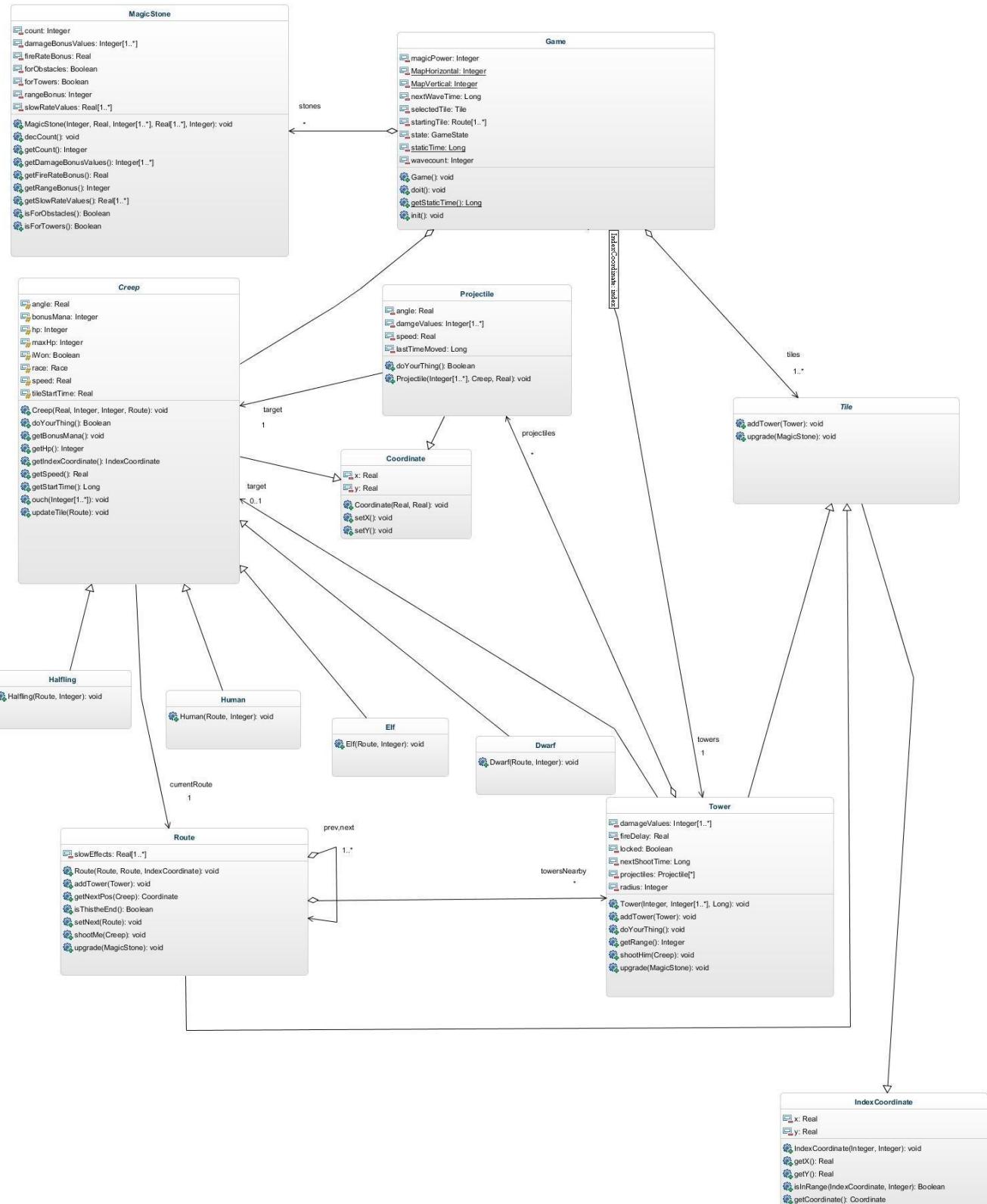
3.1.6 Route

Azok a mezők a Pályán melyeken a szövetségesek haladnak. Egy mező egy példány. Tudja hogy az útvonalban melyik mező a következő illetve az előző, és ez alapján a szövetségesek útvonalának számítását ő végzi.

3.1.7 Tower

A tornyok működéséért felelős, a feladata tehát az éppen aktuális célpont felé haladó Projectile objektum létrehozása és az aktuális célpont kiválasztása az elérhető szövetségesek közül.

3.2 Statikus struktúra diagramok



A Gamestate és a Race két enumerátor. A GameState a Game aktuális állapotát tárolja, amelyekről bővebben a 4.5.2. pontban olvashatunk. A Race pedig a Creep objektum dinamikus típusát jelöli.



3.3 Osztályok leírása

3.3.1 Creep

Felelősség

Az ellenségek absztrakt ōsosztálya. Attribútumai leírják az ellenség specifikus tulajdonságait. A creep mindenkor tudja, hogy melyik Route-on tartózkodik. Fontos, hogy a tornyokhoz a creepek regisztrálják be magukat célpontként, méghozzá a currentRoute attribútumuk shootMe metódusán keresztül. A creep felelős saját maga mozgatásáért, és sebzés hatására csökkenti az életterejét.

Ősosztályok

Coordinate

Interfészek

Attribútumok

- double angle:** azt a szöget adja meg radiánban, amennyivel az egység elfordul.
- int bonusMana:** az az érték, ami az egység megölésekor a játékos varázserjéhez adódik.
- Route currentRoute:** Az a Route-mező, amin az egység jelenleg tartózkodik
- int hp:** a szövetséges életereje
- Boolean iWon:** ha az egység az utolsó Route-mezőre kerül, akkor true.
- int maxHp:** az egység kiinduláskor meglévő életereje.
- Race race:** az egység faja.
- double speed:** megmondja, hogy az egység egy másodperc alatt mennyi Route-mezőn halad végig, ha nincs módosító tényező.
- double tileStartTime:** azt az időpontot tárolja, amikor az egység a currentRoute-ra került. Ahhoz hogy számolunk vele, ki kell vonni belőle a Game objektum staticTime attribútumát.

Metódusok

- Creep(double speed, int maxHp, int bonusMana, Route spawnPoint):** konstruktor, beállítja az egység életterejét és alapsebességét.
- boolean doYourThing():** ez a függvény regisztrálja be magát a tornyokhoz, amelyek elérik. Visszatérési értéke true, ha elfogyott a hp-ja, illetve ha a játékos beért a célba
- int getBonusMana():** visszaadja, hogy ha megölték mennyi varázserőt kap a játékos érte

- **int getHp():** visszaadja a hp attribútum értékét
- **IndexCoordinate getIndexCoordinate():** visszaadja a currentRoute index-koordinátáját
- **double getSpeed():** visszaadja az egység sebességét
- **long int getStartTime():** megmondja, hogy mennyi ideje tartózkodik a currentRoute-on
- **void ouch(int [] damageValues):** kiolvassa a fajtájához tartozó sebzési értéket, és csökkenti a hp-ját
- **void updateTile(Route t):** frissíti, hogy melyik Route-mezőn van éppen

3.3.2 Game

Felelősség

Ez az osztály a játék magja. Itt hozzuk létre és tároljuk el a legfontosabb objektumokat, illetve a játék fő vezérlése is innen történik. Ebből az osztályból egy példány keletkezik a játékban. Itt tároljuk el a játékoshoz tartozó adatokat is, mint például, hogy mennyi varázserővel rendelkezik. Innen indítjuk a wave-eket, és innen vezéreljük a lövéseket is. A Game maga egy állapotgép, a vásárlások, fejlesztések illetve a játékból való ki-be lépések kezelését állapotautomataként valósítja meg.

Ősosztályok

Interfészek

Attribútumok

- **static int MapHorizontal:** a pálya Tile-jait tartalmazó tömb vízszintes mérete.(package elérhetőségű)
- **static int MapVertical:** a pálya Tile-jait tartalmazó tömb függőleges mérete.(package elérhetőségű)
- **ArrayList<Creep> creeps:** a pályán jelenlevő Szövetségeseket tárolja, halálukkor kitörli belőle
- **int magicPower:** a játékos varázsereje
- **long nextWaveTime:** megmondja, hogy mennyi idő kell még a következő wave-ig
- **Tile selectedTile:** tárolja a játékos által kijelölt mezőt
- **ArrayList<Route> startingTile:** azokat a Route-mezőket tartalmazza, melyekről a szövetségesek indulhatnak.
- **GameState state:** a game állapotát jelzi
- **static long staticTime:** abszolút viszonyítási idő az egész játékban. Ahhoz szükséges, hogy amikor a játék áll, akkor az idő alapján számoló függvények az újraindításkor ne észleljék hogy eltelt idő.
- **MagicStone stones[]:** eltárolja e különböző hatású varázsköveket
- **public static Tile[][] tiles:** minden útnak és toronynak a lehetséges helyeit tartalmazza.
- **HashMap<IndexCoordinate, Tower> towers:** ebben tároljuk a tornyokat, koordináta szerinti kulcsokkal
- **int wavecount:** számon tartjuk, hogy hányadik wave-nél járunk

Metódusok

- **Game()**: konstruktor, létrehozza a tárolókat
- **void doit()**: legfőbb vezérlő ciklus, a játék alatt lényegében ez fut folyamatosan
- **init()**: inicializálja a játék változóit, létrehozza az utat ahol a creepek haladnak
- **static long getStaticTime()**: le lehet kérdezni az abszolút időt

3.3.3 Halfling, Human, Elf, Dwarf

Felelősség

Mindahányan a Creep-től örökölnek. Amiben különböznek egymástól, az a konstruktorukfüggvényük, amely minden egységet másképp jelenít meg.

A konstruktor megkapja a kezdőpontot és azt, hogy hányadik wave-nél járunk. Ezekből az adatokból kiszámítja az attribútumainak a kezdőértékeit, majd meghívja az ösosztály konstruktorát. A felelősség és a szerepkör ugyanaz, mint a szülőnél, a különböző fajok csak a szülő speciális leszármazottai.

Ősosztályok

Coordinate -> Creep

Interfészek

-

Attribútumok

- minden attribútuma megegyezik a Creep attribútumaival

Metódusok

- **Halfling(Route spawnPoint, int waveCount)**:
 - **Hobbit(Route spawnPoint, int waveCount)**:
 - **Dwarf(Route spawnPoint, int waveCount)**:
 - **Elf(Route spawnPoint, int waveCount)**:
- létrehoznak egy-egy szövetséget az adott helyen, waveCount alapján számolt tulajdonságokkal.

3.3.4 IndexCoordinate

Felelősség

Minden Tile-hoz tartozik egy index, ami a helyzetét határozza meg a pálya mezőinek tömbjében, illetve könnyen számítható belőle szomszédosság illetve távolság.

Ősosztályok

Interfészek

Attribútumok

- **int x:** x dimenziójú index
- **int y:** y dimenziójú index

Metódusok

- **boolean isInRange(IndexCoordinate idx, int range):** megadja, hogy egy adott indexkoordináta range-en belül találhető-e tőle (hány lépésből lehet eljutni oda)
- **IndexCoordinate(int xx, int yy):** konstruktur, alapérték beállítása
- **Coordinate getCoordinate():** kiszámolja az index alapján a koordinátát a képernyőn

3.3.5 MagicStone

Felelősség

Varázskövek, ennek az osztálynak a segítségével történnek a fejlesztések. A varázskő számon tartja, hogy milyen fejlesztésekre képes, illetve hogy melyik creep-re milyen mértékű hatást gyakorol. A MagicStone osztály egy példánya lényegében az azonos tulajdonságú varázskövek egy csoportját valósítja meg. Egy varázskő felhasználásakor csökken a példány számlálója, ezzel számon tartva, hogy hánnyal elérhető van még belőle. Tárolja még hogy a fejlesztés toronyra illetve akadályra vonatkozik-e. Ez alapján lehet majd egy adott torony vagy akadály fejlesztés során eldönteni hogy megjelenik-e a listában.

Ősosztályok

Interfészek

Attribútumok

- **int count:** megmonja, hogy a játékos még maximum hányat tud felhasználni ebből a fajtából.
- **int[] damageBonusValues:** itt tárolja, hogy melyik creep-et menyivel sebzi.
- **double fireRateBonus:** ennyivel növeli a torony tüzelési gyakoriságát.
- **boolean forObstacles:** megmondja hogy az adott varázskő felhasználható-e akadály fejlesztésére.
- **boolean forTowers:** megmondja hogy az adott varázskő felhasználható-e torony fejlesztésére.
- **int rangeBonus:** ennyivel növeli a torony hatósugarát.
- **double[] slowRateValues:** itt tárolja, hogy melyik creep-et mennyivel lassítja.

Metódusok

- **MagicStone(int rangeBonus, double fireRateBonus, int[] DamageBonusValues, double[] slowRateValues, int count, boolean forTowers, boolean forObstacles):** konstruktor, melyben beállítjuk az attribútumok kezdőértékét
- **void decCount():** eggyel csökkenti a count értékét
- **int getCount():** visszaadja a conunt értékét
- **int[] getDamageBonusValues():** megadja, hogy melyik creep-et mennyivel sebzi
- **double getFireRateBonus():** megadja a torony tüzelési gyakoriságát
- **boolean getForObstacles():** igazzal tér vissza, ha akadályra használható a kő
- **boolean getforTowers():** igazzal tér vissza, ha tornyakra használható a kő
- **int getRangeBonus():** megadja a torony lövési hatósugarát
- **double[]getSlowRateValues():** megadja, hogy melyik creep-et mennyivel lassítja

3.3.6 Projectile

Felelősség

Ez az osztály valósítja meg a lövedéket. A torony hozza létre, azonban létrehozás után actor-ként viselkedik. Felelős a saját maga mozgatásáért, és ő hívja meg a creep sebző függvényét is. Ha elérte célpontját szól a toronynak, hogy befejezte küldetését.

Ősosztályok

Coordinate

Interfészek

-

Attribútumok

- **double angle:** a lövedék sebességének az irányá
- **int[] damageValues:** eltárolja, hogy melyik fajt milyen mértékben tudja sebezni
- **long lastTimeMoved:** eltárolja hogy mikor számol
- **double speed:** a lövedék sebességének a nagysága
- **Creep target:** a lövedék célpontja, őt fogja követni, majd megsebezni

Metódusok

- **Projectile(Creep target, int[] damageValues, double speed):** konstruktor, ez a függvény inicializálja a lövedéket
- **boolean doYourThing():** ez a függvény felelős a creep sebzéséért, és önmaga mozgatásáért. Visszatérési értéke true, ha elérte a creepet.

3.3.7 Route

Felelősség

Azokat a mezőket testesíti meg, melyeken a szövetségesek mozogni tudnak. feladata a rajta áthaladó Creepeknek meghatározni az útvonalukat, illetve rajta keresztül történik a tornyok értesítése a közelben levő egységekről.

Ősosztályok

IndexCoordinate -> Tile

Interfészek

-

Attribútumok

- **Route next:** a következő Route-mezőre mutat (ide küldi öket tovább)
- **Route prev:** az előző Route-mezőre mutat(innen jönnek a creepek erre)
- **double[] slowEffects:** megadja, hogy az adott fajú szövetségesekre mekkora lassító hatással van.
- **ArrayList<Tower> towersNearby:** azokat a tornyokat tárolja, akiknek beleesik a hatósugarába

Metódusok

- **Route(Route prev, IndexCoordinate idx):** konstruktur, megadja hogy melyik az előző Route-mező, és az elhelyezkedését a pályán index-koordinátában.
- **void addTower(Tower t):** egy torony felvétele a towersNeabyListába.
- **Coordinate getNextPos(Creep it):** visszaadja egy adott szövetséges helyzetét a jelenlegi időpillanatban
- **boolean isThisTheEnd():** megmondja hogy ez egy utolsó Mező-e a route
- **void setNext(Route nxt):** beállítja a next attribútumot
- **void shootMe(Creep it):** egy Creep tud jelezni vele, hogy itt van, így a Route továbbítja a kérést azoknak a tornyoknak aikik ezt az adott mezőt elérík.(towersNearby)
- **upgrade(MagicStone ms):** végrehajtja a paraméter által meghatározott változtatásokat a tornyon: (megszorozza a lassítás mértékét a jelenlegi értékkel minden egyes fajhoz egyenként)

3.3.8 Tile

Felelősség

A pálya alkotóelemei. Az IndexCoordinate osztályból örökölnek, így az ősosztályuk összes függvénye meghívható rájuk is. A Tile-okat lehet fejleszteni is, ami az akadályépítésnél fontos, illetve egy Tile el tudja tárolni, hogy mely tornyoknak esik a hatósugarán belülre.

Ősosztályok

IndexCoordinate

Interfészek

-

Attribútumok

Metódusok

- **void addTower(Tower t):** absztrakt metódus, nincs implementálva
- **void upgrade(MaicStone ko):** absztrakt metódus, nincs implementálva

3.3.9 Tower

Felelősség

A toronyokat megvalósító osztály, a Tile leszármazottja. Állapotgépként viselkedik. Lőni akkor tud, ha van kit, és a lövési gyakoriság attribútuma is engedélyezi. Azt, hogy ki lehet az aktuális célpontja, azt kívülről kapja meg, ő maga ezek közül választja ki a ténylegeset. A torony tud lőni, de a lövő függvény vezérlése kívülről történik. Ha van érvényes célpont, és lőhet is, akkor a torony létrehoz egy lövedéket. A torony szólítja fel a lövedékeket mozgásra.

Ősosztályok

IndexCoordinate -> Tile

Interfészek

-

Attribútumok

- **int [] damageValues:** eltárolha, hogy az egyes fajokat milyen mértékben sebzi
- **double fireDelay:** tüzelési gyakoriságot
- **boolean locked:** igaz, ha a torony már eldöntötté hogy kit lő, és még ténylegesen tudja is lőni azt.
- **long int nextShotTime:** ennyi idő van még a következő lövésig
- **ArrayList<Projectiele> projectiles:** a kilött, még nem célbaért Projectileokat tárolja
- **int radius:** torony hatósugara
- **Creep target:** ezt az ellenséget lövi

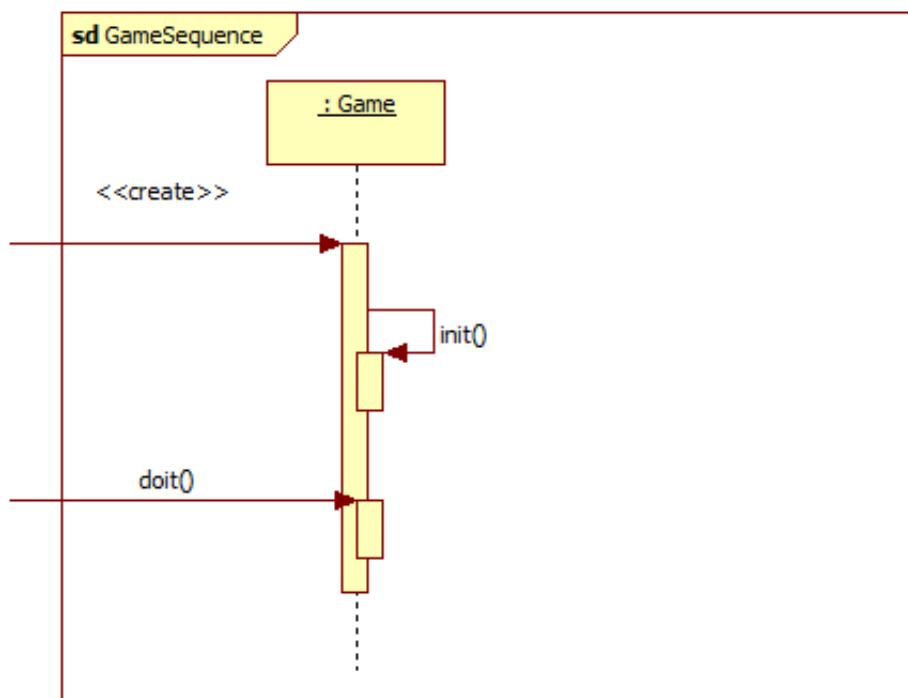
Metódusok

- **Tower(int radius, int[] damageValues, long int fireDelay):** konstruktor, beállítja a toronyhoz tartozó alap értékeket.
- **void doYourThing():** ez a függvény felelős a lövésért, illetve a projectile tömb kezelésért.

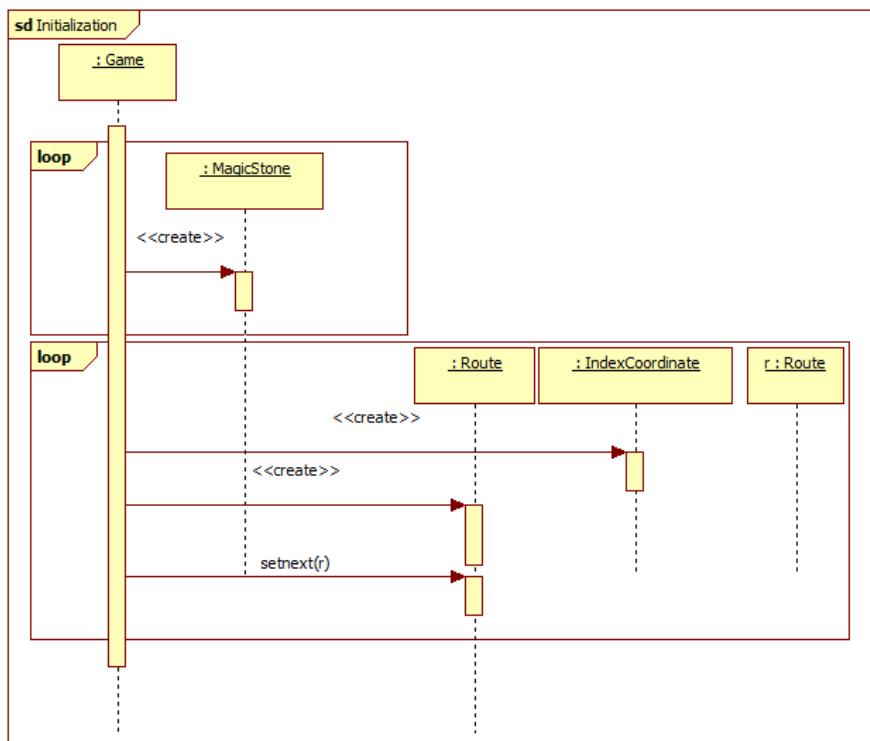
- **void shootHim(Creep it):** a Torony eldönti hogy a paraméterben kapott Creep ideálisabb célpont-e, mint az eddigi legjobb(az összes elérhető Creepból befut egy ilyen kérés, így egy maximumkeresést valósít meg tehát) Ha locked igaz, akkor ellenőrzi hogy még tudja-e lőni a célpontot, és ha nem, akkor hamisra állítja. ha még elérhető, akkor nem csinál semmit a függvény.
- **int getRange():** vissza adja a torony hatósugarát
- **void addTower(Tower t):** Tile-tól örökölött függvény, a függvény törzse üres
- **void upgrade(MagicStone stone):** a stone rávonatkozó tulajdonságaival fejleszti magát

3.4 Szekvencia diagramok

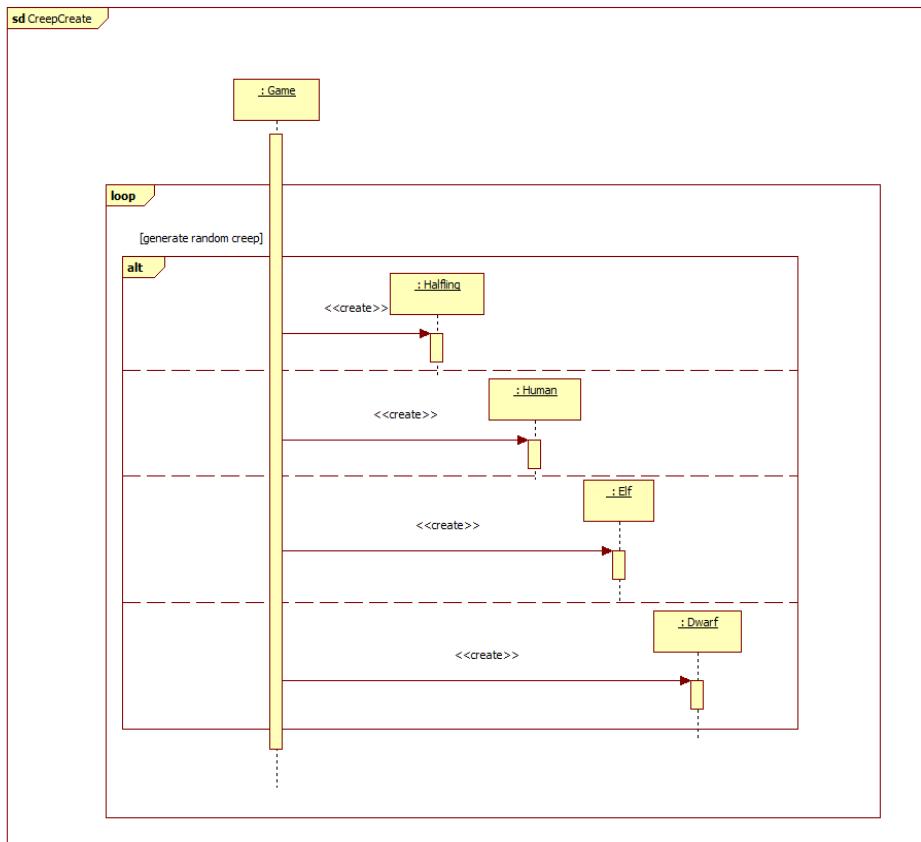
4.4.1 Játék szekvenciája



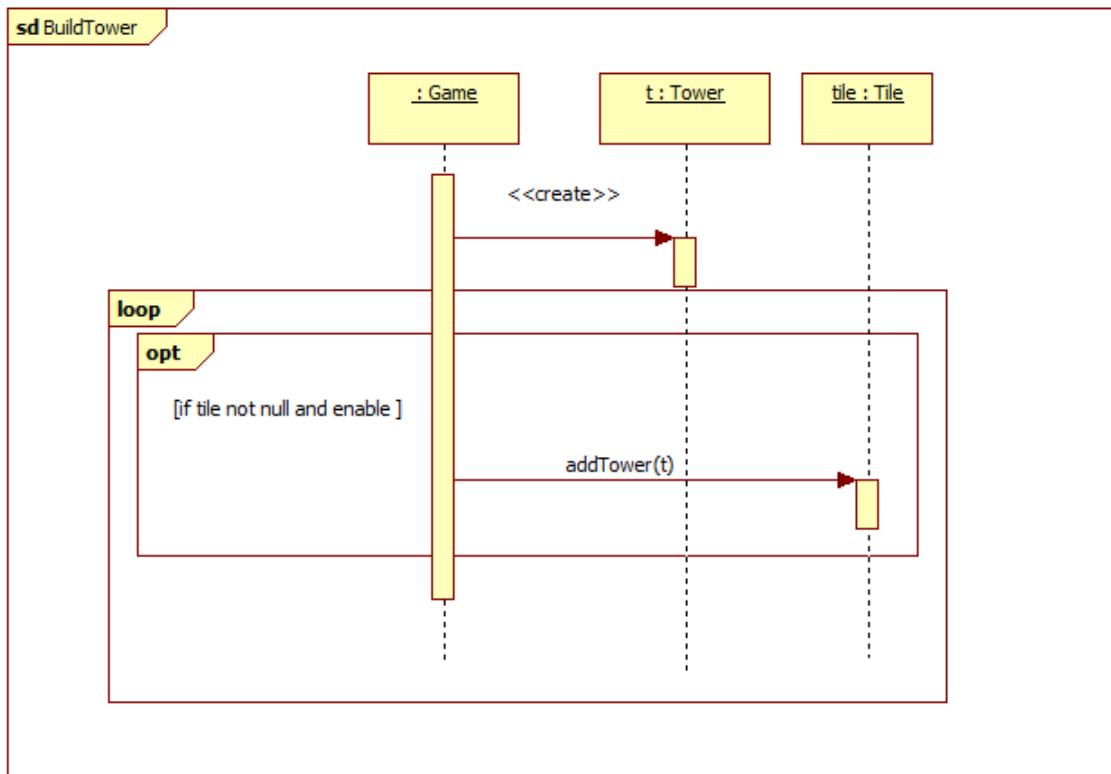
4.4.2 Inicializálás



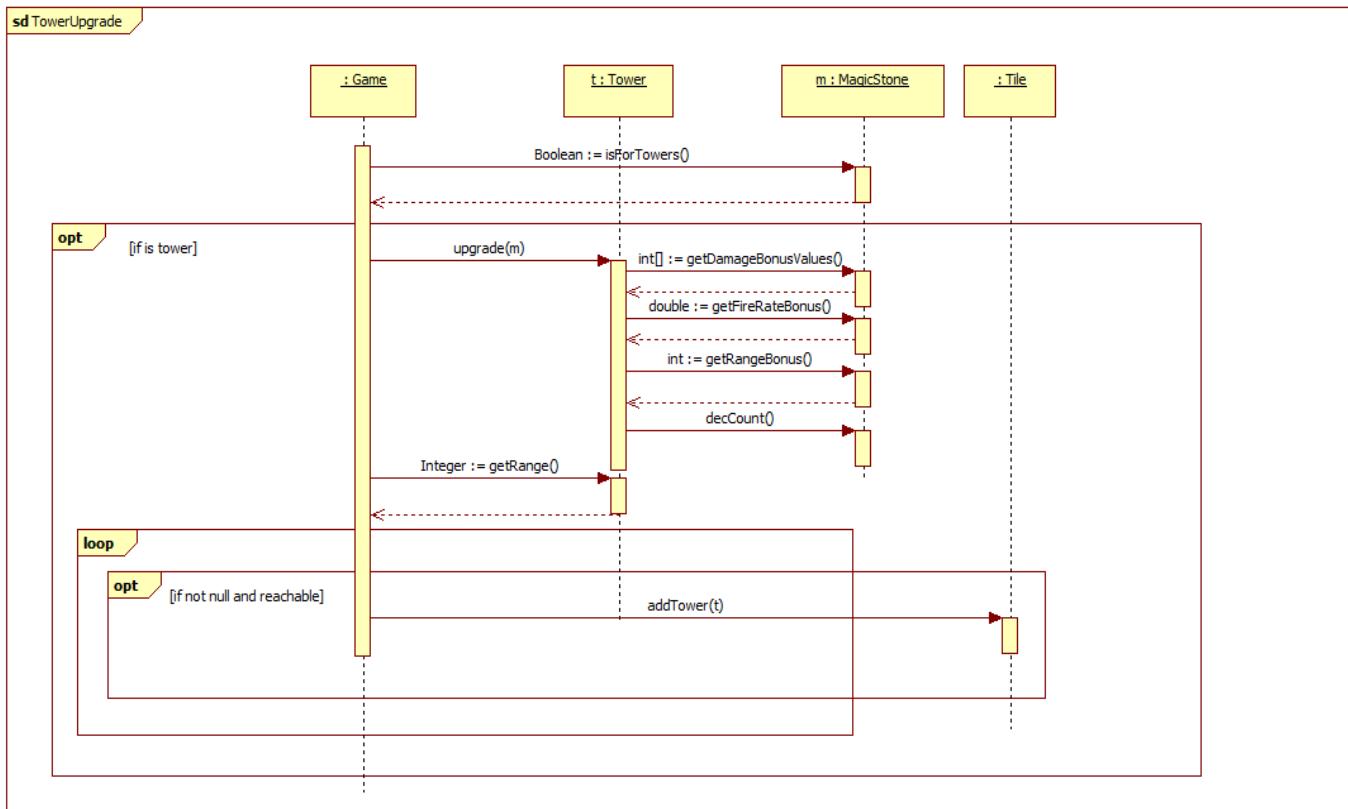
4.4.3 Szövetséges létrehozása



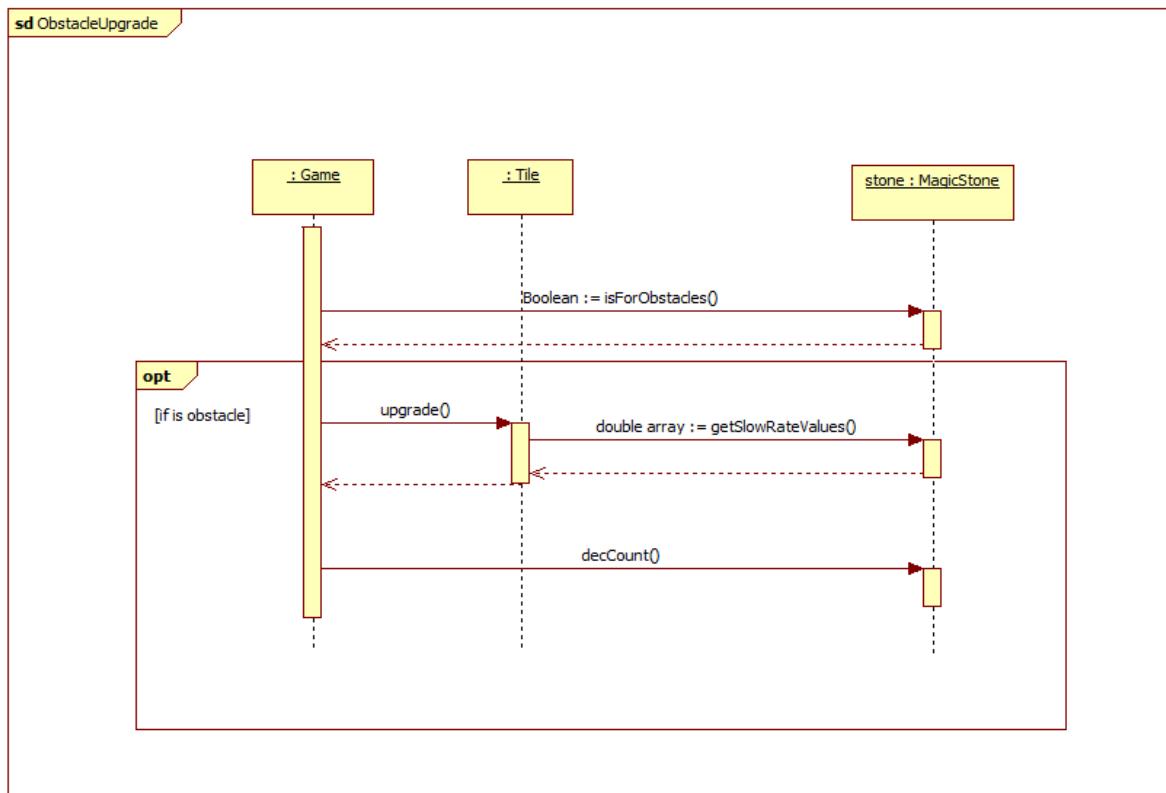
4.4.4 Toronyépítés



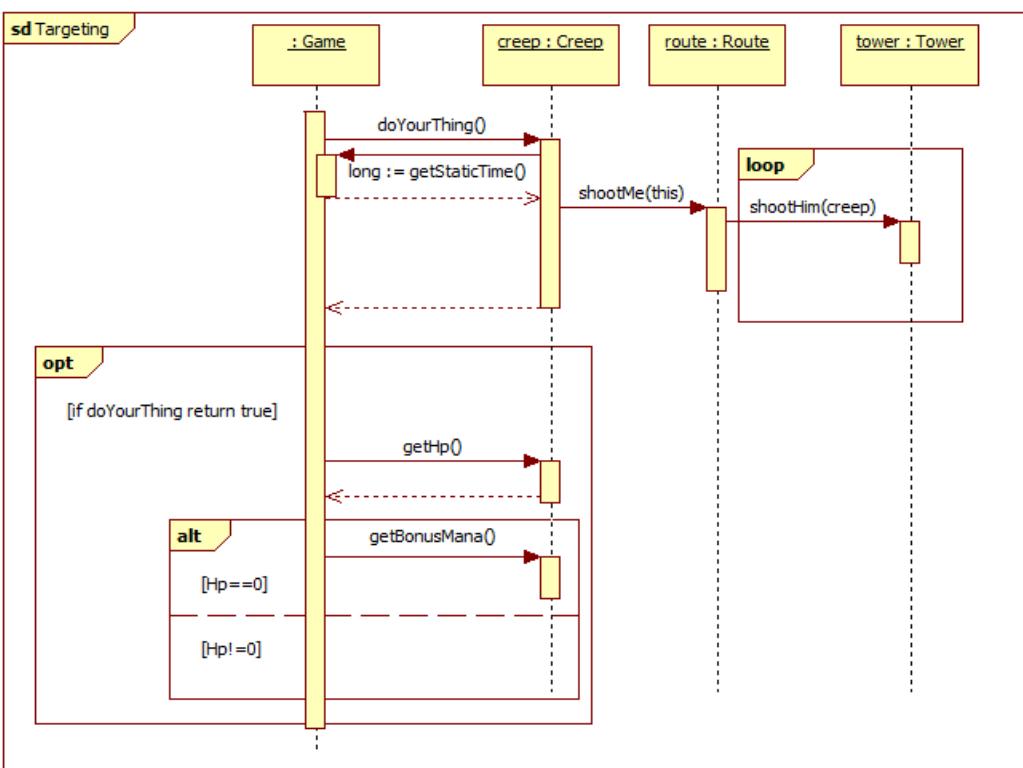
4.4.5 Toronyfejlesztés



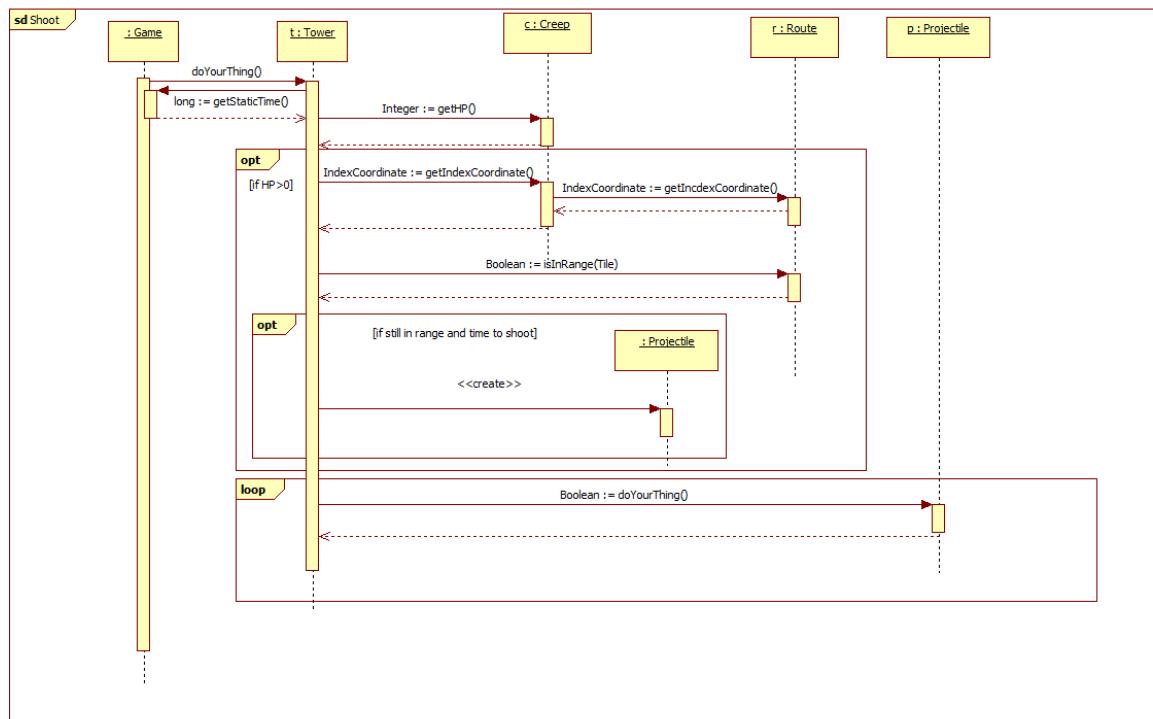
4.4.6 Akadályfejlesztés



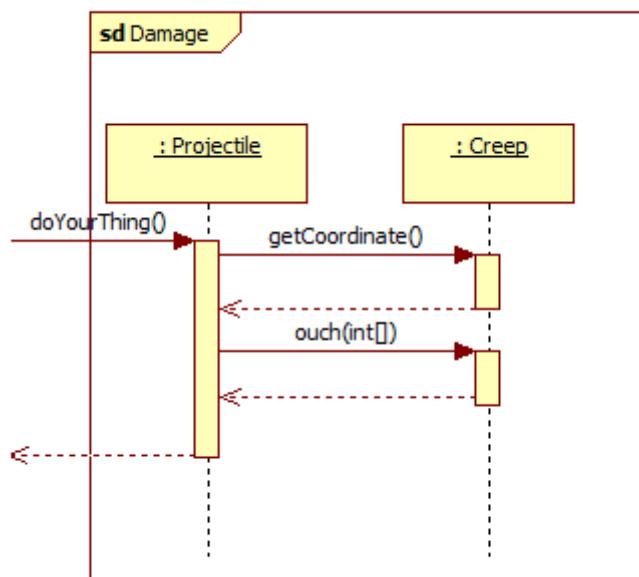
4.4.7 Célzás



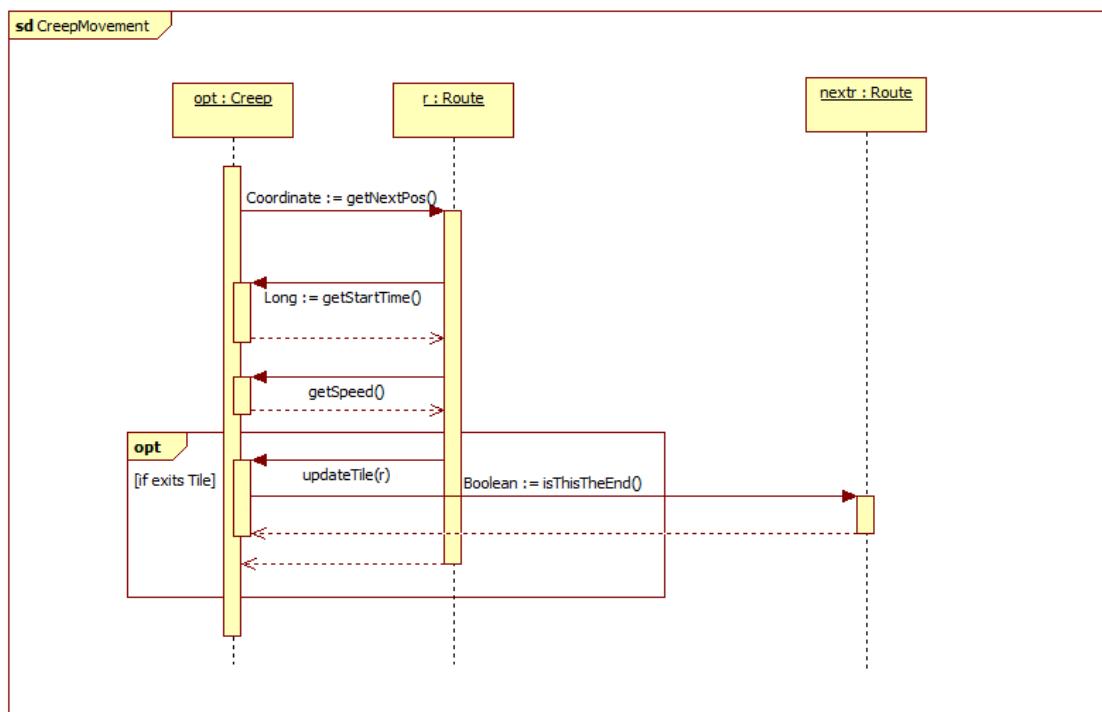
4.4.8 Lövés



4.4.9 Sebzés

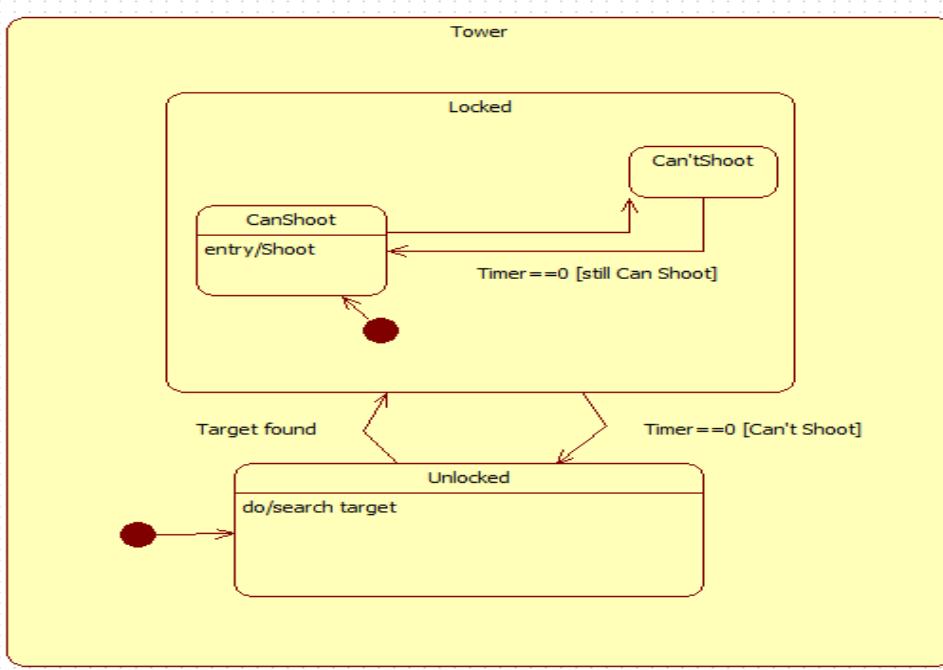


4.4.10 Szövetséges mozgatása

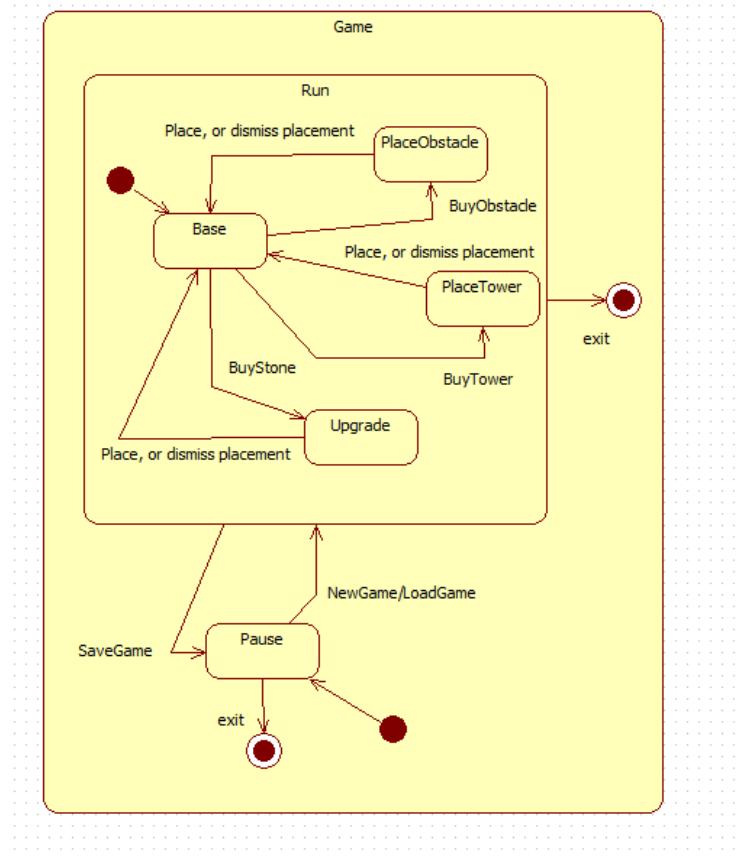


3.5 State-chartok

4.5.1 Torony állapota



4.5.2 Game állapot



3.6 Napló

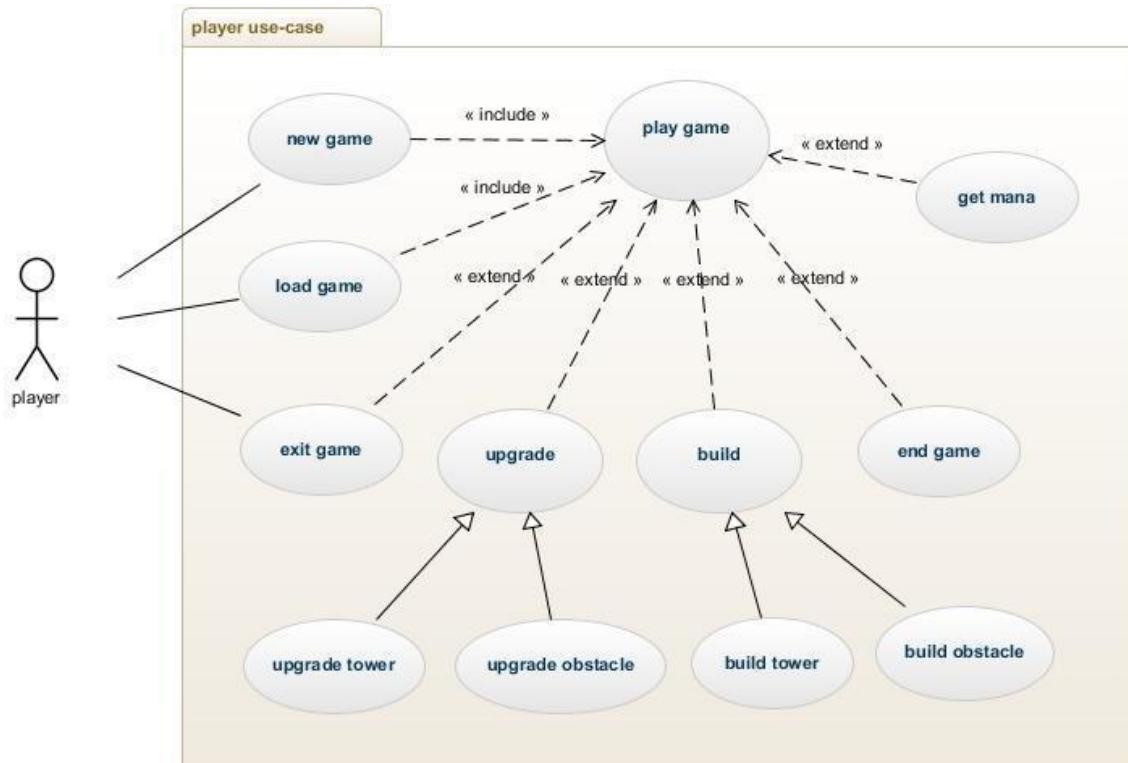
Kezdet	Időtartam	Résztvevők	Leírás
2014.03.05. 8:00	1,5 óra	Iklodi Molnár Németh Srajner	Konzultáció a konzulenssel, analízis modell hibáinak feljegyzése
2014.03.07. 13:00	2 óra	Iklodi Molnár Németh Srajner	A felmerülő hibák egyeztetése, kijavítása
2014. 03.08. 20:00	1,5 óra	Iklodi Radnai Srajner	Hiányos részek pótlása, szekvencia és osztálydiagramok átnézése
2014. 03.09. 15:00	0,5	Molnár Németh	Dokumentáció formázása, lezárása

5. Szkeleton tervezése

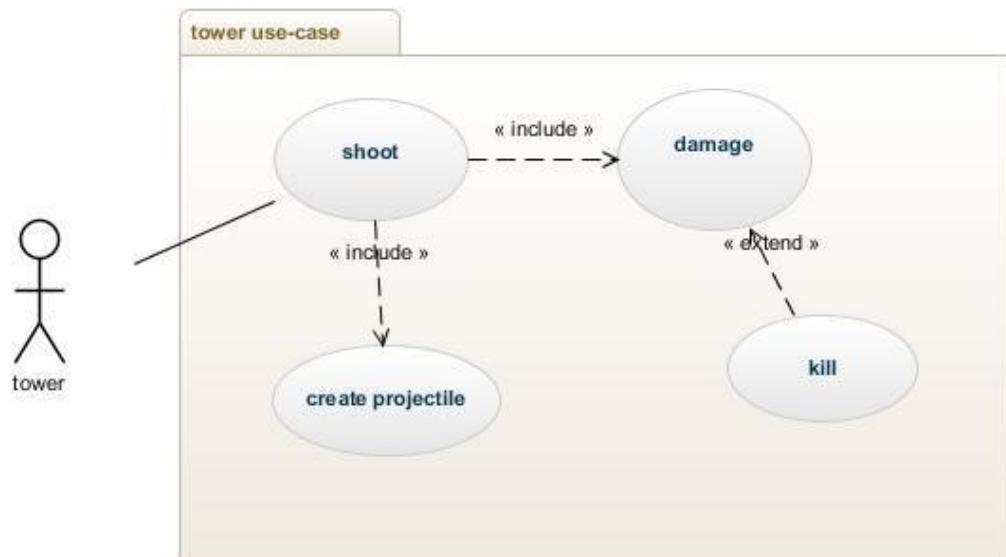
5.1 A szkeleton modell valóságos use-case-ai

5.1.1 Use-case diagram

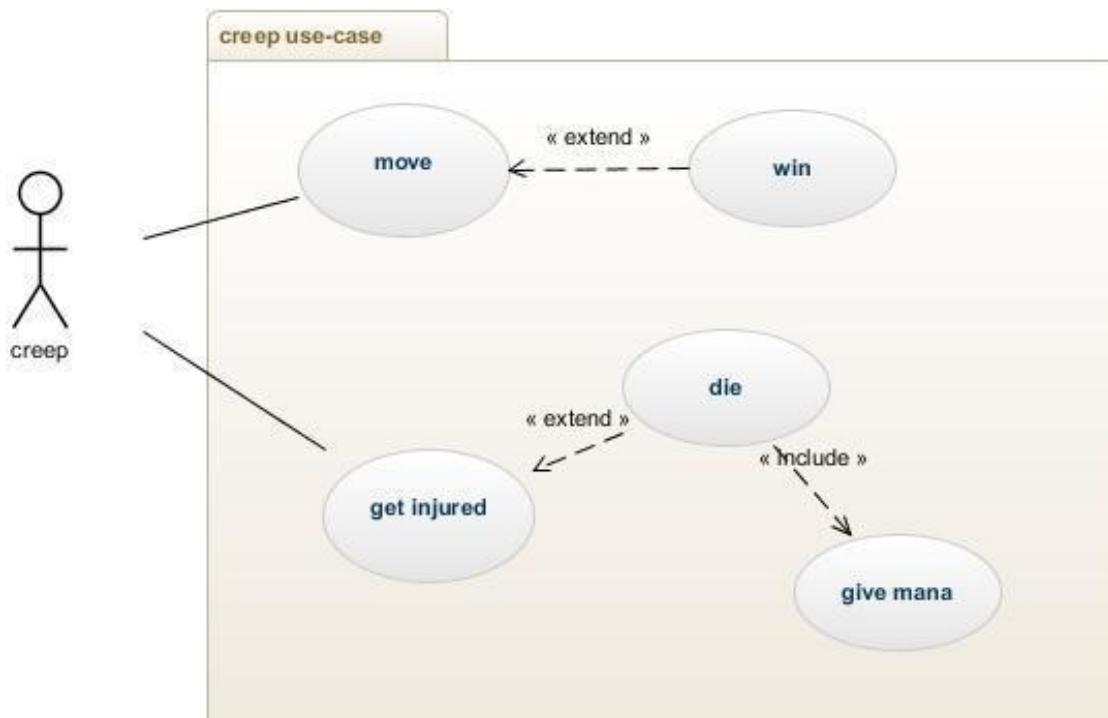
5.1.1.1 A Player-hez tartozó use-case diagram:



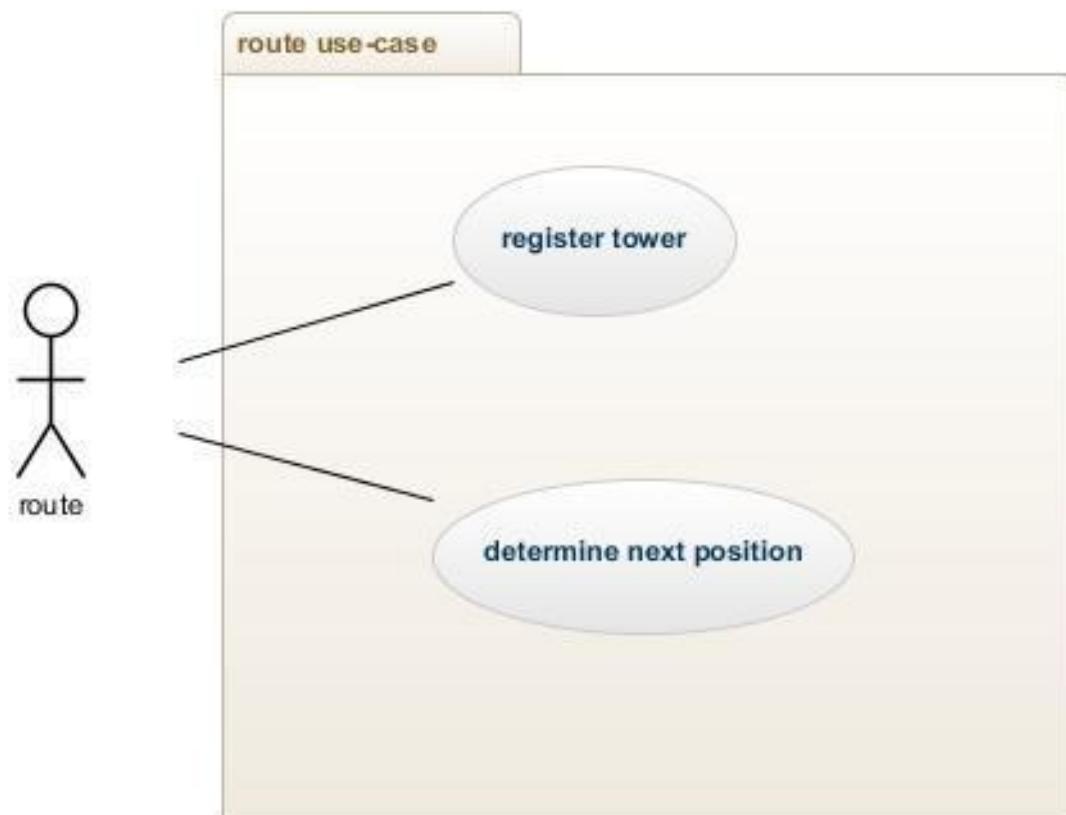
5.1.1.2 A Tower-hoz tartozó use-case diagram:



5.1.1.3 A Creep-hez tartozó use-case diagram:



5.1.1.4 A Route-hoz tartozó use-case diagram:



5.1.2 Use-case leírások

5.1.2.1 A Player-hez tartozó use-case-ek:

Use-case neve	New Game
Rövid leírás	Új játék kezdése
Aktorok	player
Forgatókönyv	A játékos új játékot indít. Itt négy fajta varázskövet, egy tornyot, egy embert, és három utat inicializálunk.

Use-case neve	Load Game
Rövid leírás	Egy korábban elmentett játékállás betöltése
Aktorok	player
Forgatókönyv	A játékos egy korábbi játék állást tölt be és folytat. Itt már csak két fajta varázskövet, ellenben két tornyot, négy creepet, és három utat inicializálunk.

Use-case neve	Exit Game
Rövid leírás	A játék bezárása
Aktorok	player
Forgatókönyv	A játékos ezzel kilép az egész alkalmazásból.

Use-case neve	Build Tower
Rövid leírás	Új torony építése a pályán
Aktorok	player
Forgatókönyv	A játékos új tornyot helyezhet, adhat hozzá a játékhoz, továbbá megadja, hogy melyik út essen a torony hatótávolságába.

Use-case neve	Build Obstacle
Rövid leírás	Új akadály elhelyezése a pályán
Aktorok	player
Forgatókönyv	A játékos új akadályt adhat hozzá a játékhoz.

Use-case neve	Upgrade Tower
Rövid leírás	Egy torony fejlesztése
Aktorok	player
Forgatókönyv	A játékos kiválasztja a tornyot és a magic stone-t, amivel fejleszteni szeretne.

Use-case neve	Upgrade Obstacle
Rövid leírás	Egy akadály fejlesztése
Aktorok	player
Forgatókönyv	A játékos kiválasztja, hogy melyik utat szeretné fejleszteni, és hogy melyik varázskővel.

Use-case neve	Get Mana
Rövid leírás	Megölt ellenség után manát kap
Aktorok	player
Forgatókönyv	A játékos megmondja, hogy melyik ellenséget pusztította el, s ilyenkor lekérdezzük az ezért a játékosért járó mana-t.

Use-case neve	Save Game
Rövid leírás	Aktuális játékállás elmentése
Aktorok	player
Forgatókönyv	A játékos elmentheti a játékmenet aktuális állását. Valódi mentés nem történik, csak azért szerepel, mert a végső verzióban lesz majd ilyen use-case.

Use-case neve	End Game
Rövid leírás	Játékmenet befejezése
Aktorok	player
Forgatókönyv	A játékos megadja, hogy melyik creep ért célba, és ekkor minden objektumnak, amelyet a new game/load game parancsban hoztunk létre, lefut a destruktora.

5.1.2.2 A Tower-hez tartozó use-case:

Use-case neve	Shoot
Rövid leírás	Lövés egy ellenséges egységre
Aktorok	tower
Forgatókönyv	A játékos megadja, hogy melyik tower lőjön. Mivel még az objektumok attribútumai inicializálatlanok, így a tower létrehoz maga egy creepet, és őt fogja lőni.

Use-case neve	Kill
Rövid leírás	Ellenséges egység elpusztítása
Aktorok	tower
Forgatókönyv	A játékos megadja, hogy melyik creep-et öljük meg, s ilyenkor töröljük a játékból.

5.1.2.3. A Creep-hez tartozó use-case-ek:

Use-case neve	Move
Rövid leírás	Előre haladás a cél irányába
Aktorok	creep
Forgatókönyv	A játékos megadja, hogy melyik creep-et mozgassuk, majd a program megkérdezi, hogy lépjene a creep a következő route-ra. A creep a játékos válasza szerint cselekszik. A route-okat a creep hozza létre. Ha továbblépett egy következő útra, megkérdezzük, a játékos, hogy ez a mező-e az utolsó. Amennyiben igen, úgy az alábbi win use-case teljesül.

Use-case neve	Win
Rövid leírás	Játék megnyerése
Aktorok	creep
Forgatókönyv	Lásd Move.

Use-case neve	Get Injured
Rövid leírás	Sérülés, életerő vesztése
Aktorok	creep
Forgatókönyv	A játékos megadja, hogy melyik creep sebződjön. Az itt lefutó függvény már szerepel a shoot-ban is, azért szerepel itt is, hogy külön demonstráljuk a creep-nek is ezt a use-case-ét.

Use-case neve	Die
Rövid leírás	Halál, egység elpusztulása
Aktorok	creep
Forgatókönyv	A játékos megadja, hogy melyik creep pusztuljon el, és még mielőtt töröljük, elkérjük az érte járó mana-t.

5.1.2.4. Route-hoz tartozó use-case-e:

Use-case neve	Register Tower
Rövid leírás	Az út eltárolja, hogy melyik tornyoknak esik a hatósugarába.
Aktorok	route
Forgatókönyv	A játékos megadja, hogy melyik úthoz, és hogy melyik tornyot regisztráljuk be.

Use-case neve	Determine Next Position
Rövid leírás	Következő pozíció megadása
Aktorok	route
Forgatókönyv	A játékos megadja, hogy melyik út adja meg a következő pozíciót. Az út egy creep-nek tudja megadni a következő pozíóját, így az út maga létrehoz egy creep-et, és ennek hívja meg a megfelelő függvényeit.

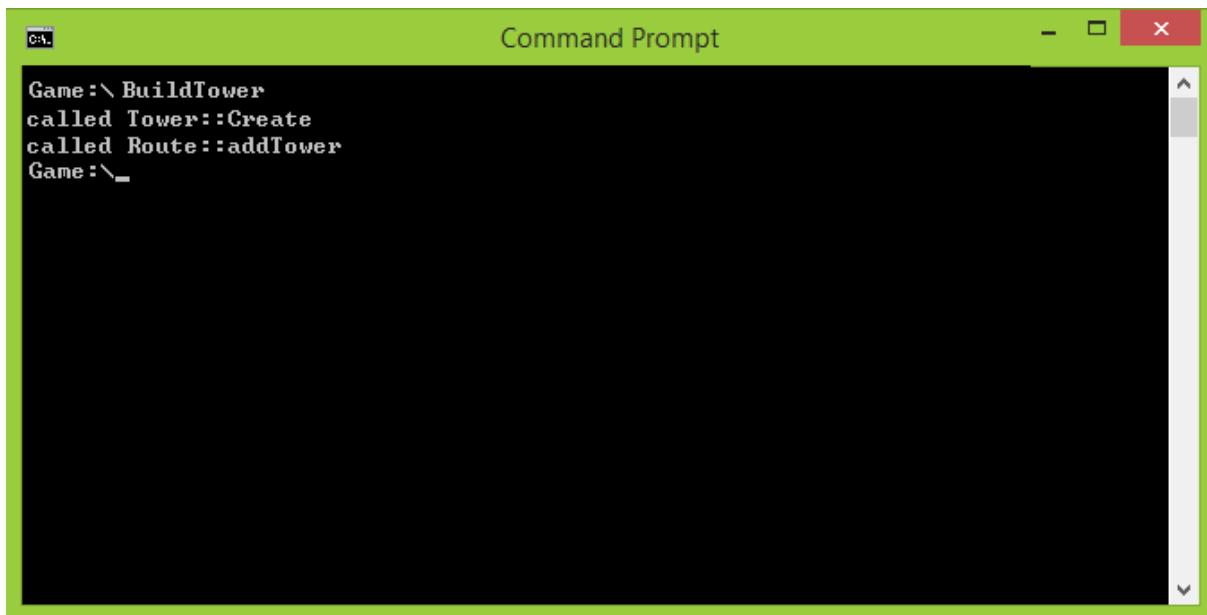
5.2 A szkeleton kezelői felületének terve, dialógusok

A szkeleton program egy konzolos alkalmazás, amely lehetővé teszi a program főbb szekvenciáinak a tesztelését. A szkeletonban minden business objektum megtalálható, azonban csak az interfészük kerül megvalósításra.

A könnyebb megvalósítás érdekében nem hozunk létre Game class-t. A Game class a program futásáért felel, illetve ő tárolja el az objektumokat. Mivel a szkeletonban előre megadott környezetet tesztelünk, így nincs szükségünk erre az osztályra. Helyette létrehozunk egy Dummy osztályt, amelyet speciálisan a szkeletonra tervezünk. Ez az osztály tárolja az összes létrehozott objektumot, illetve végzi a parancsfeldolgozást.

A szkeleton kezelőfelülete egy parancsfeldolgozó. Egy parancs lényegében egy use-case-t valósít meg. (Kivételt képeznek ez alól az extended use-casek, mint például a move-win. Ott, amennyiben a felhasználó a "ez az utolsó útdarab?" kérdésre igennel felel, úgy már a mozgásnak az a kiterjesztett esete valósul meg, amikor az ellenfél nyert.) Továbbá az exit game függvényhez nem tartozik külön szekvencia diagram. Ez az eset azt valósítja meg, amikor x-re kattintva kilépünk az egész alkalmazásból, de hogy ne kelljen bezárni a konzolablakot elég ha beütjük az exit game parancsot. A parancsokat a help nevű parancccsal lehet majd kilistázni. A szkeleton indítása után először a new game, a load game, az exit game, és a help tartozik pusztán az elérhető parancsok közé. A new game és load game között az a különböző, hogy másként inicializálja a játékot Ezek után elérhető csak el a többi parancs. Néhány parancs futása közben a program bemenetet kér a felhasználótól. Amennyiben a felhasználónak egy objektumlistából kell választania, úgy a program felsorolja a lehetőségeket.

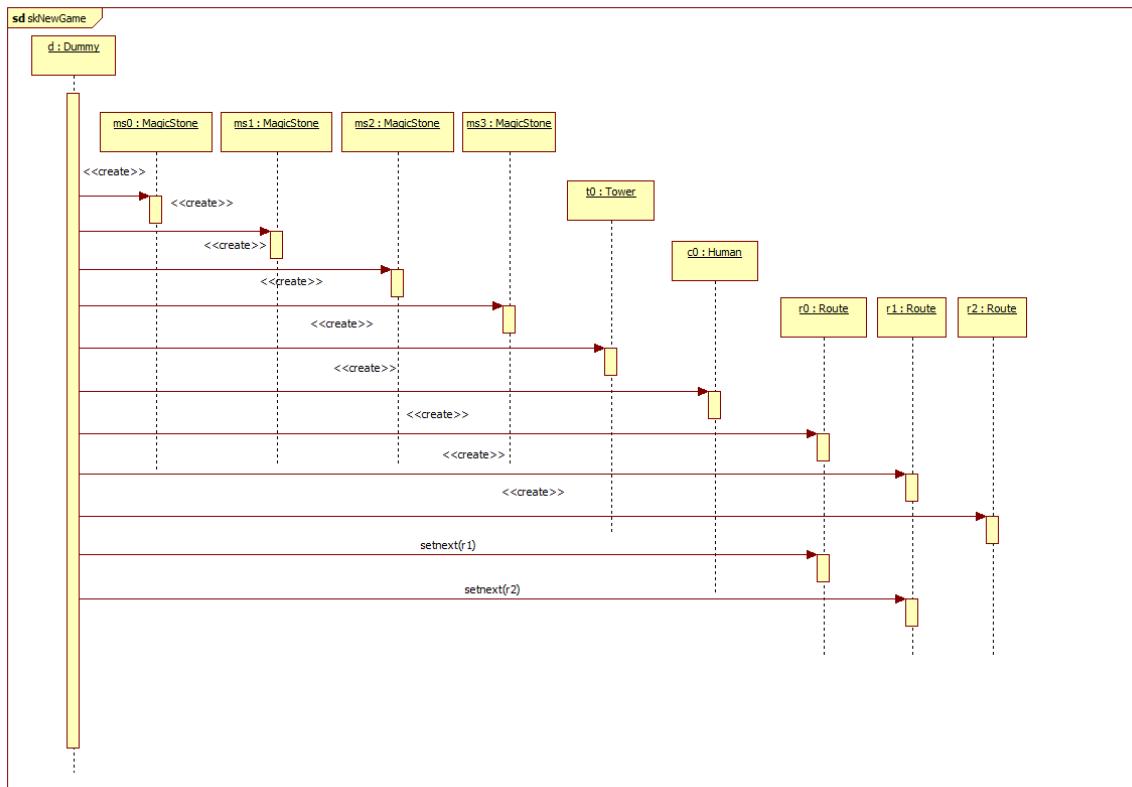
A program mindenki írja az éppen meghívott metódus nevét. Ezzel nyomon követhető a szekvenciák működése. A metódus neve előtt c++-os namespace szintaktikához hasonlóan írjuk ki, hogy mely osztályhoz tartozik. Az alábbi kép demonstrálja a kezelőfelületet:



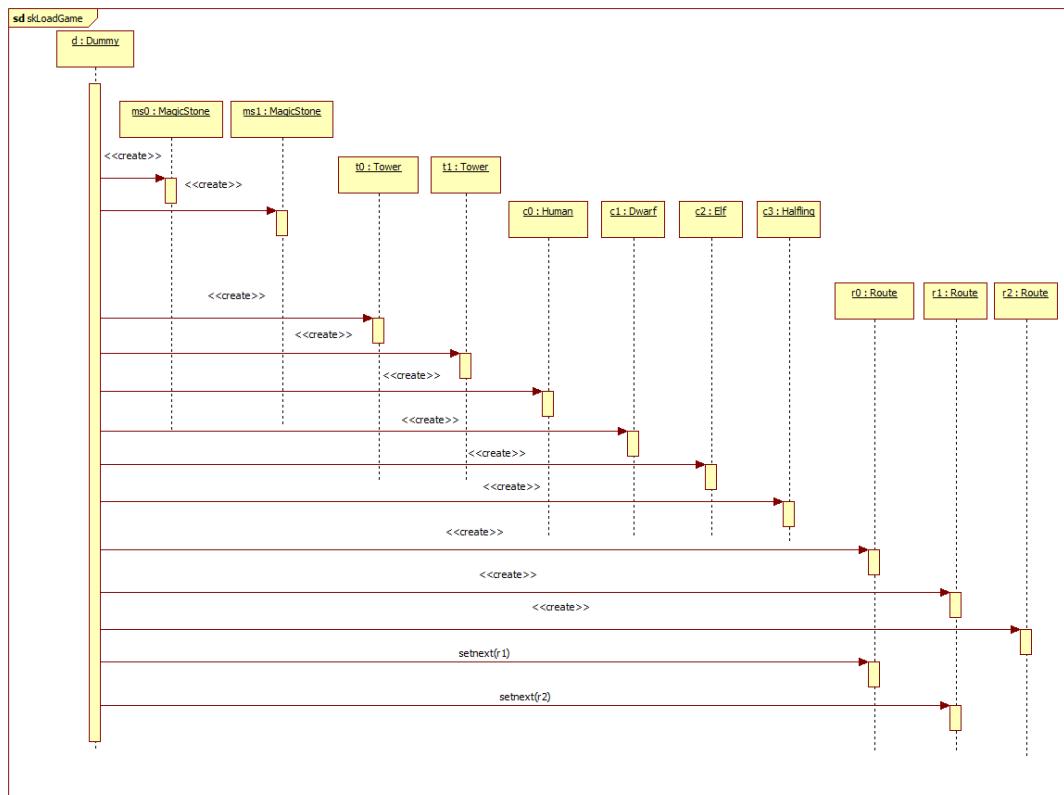
```
Game:\nBuildTower\ncalled Tower::Create\ncalled Route::addTower\nGame:\n
```

5.3 Szekvencia diagramok a belső működésre

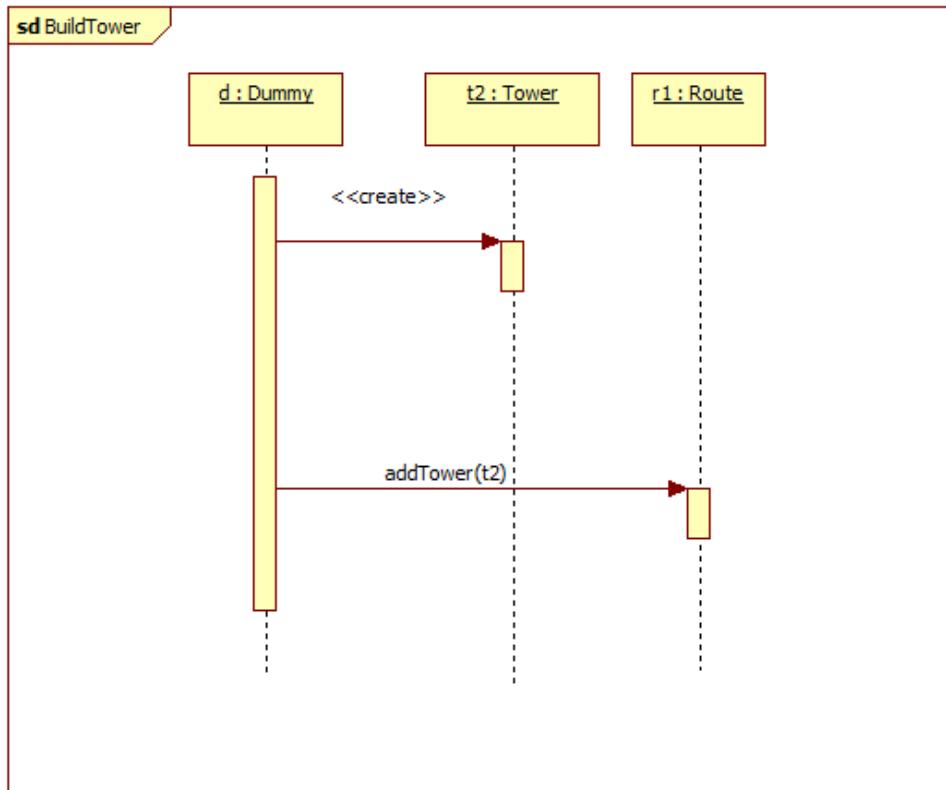
5.3.1 New Game



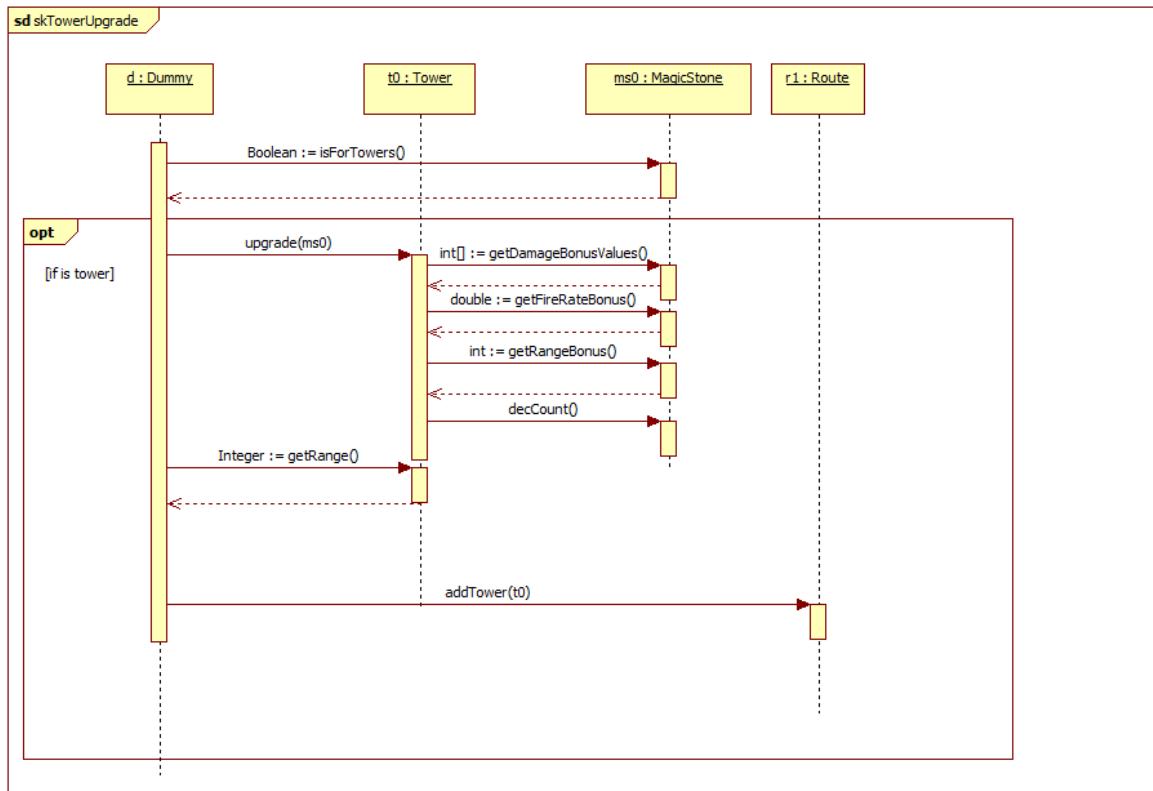
5.3.2 Load Game



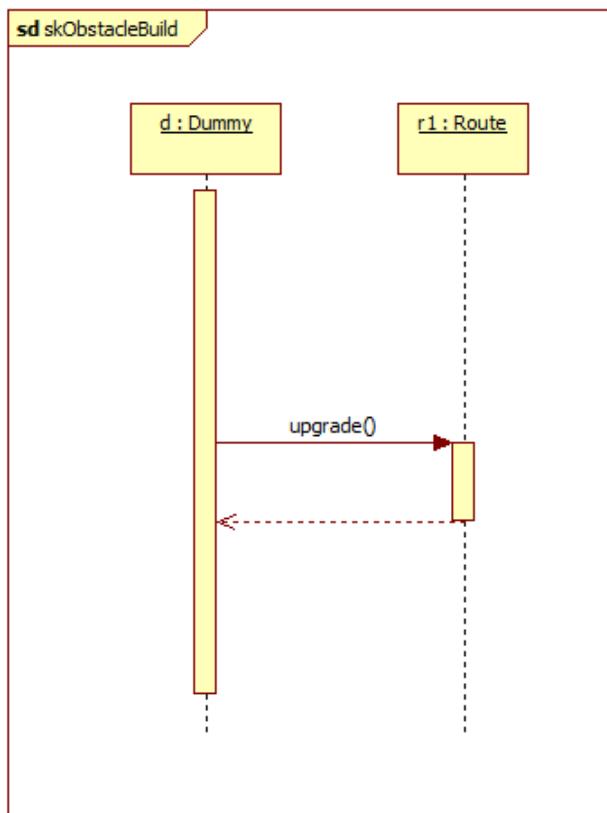
5.3.3 Build Tower



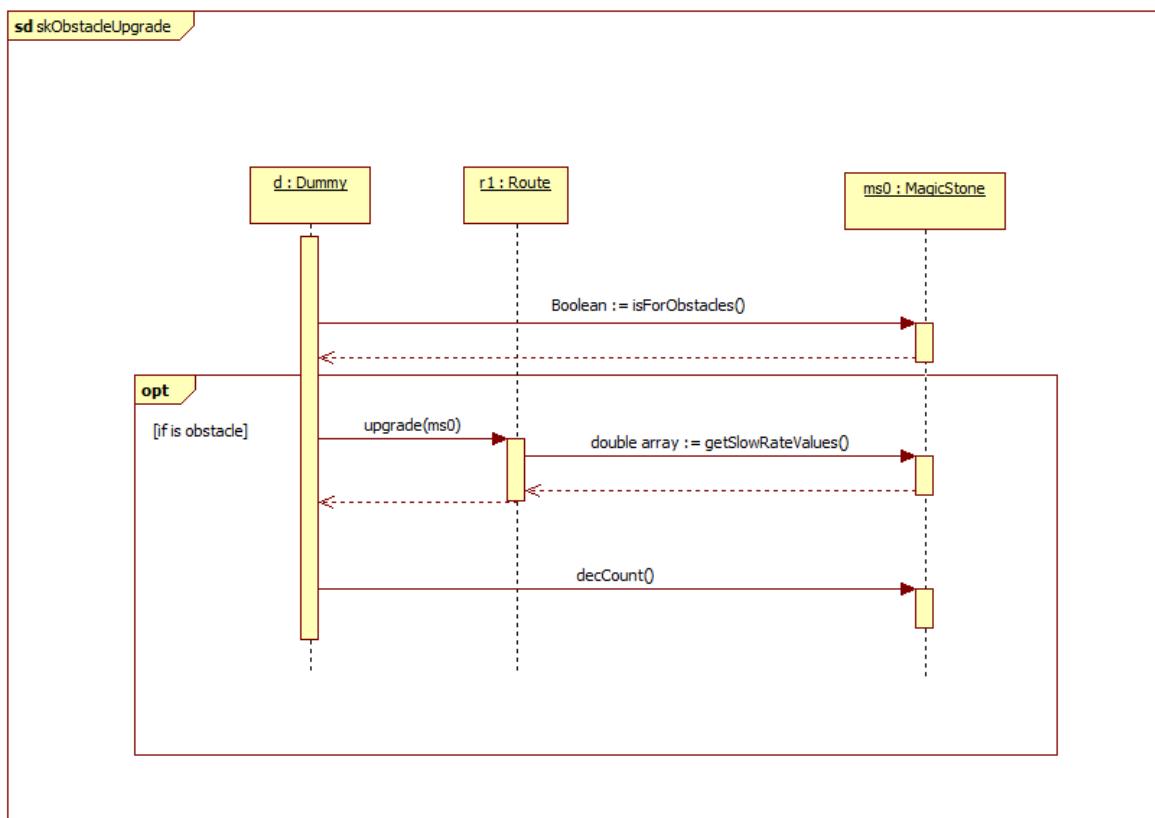
5.3.4 Upgrade Tower



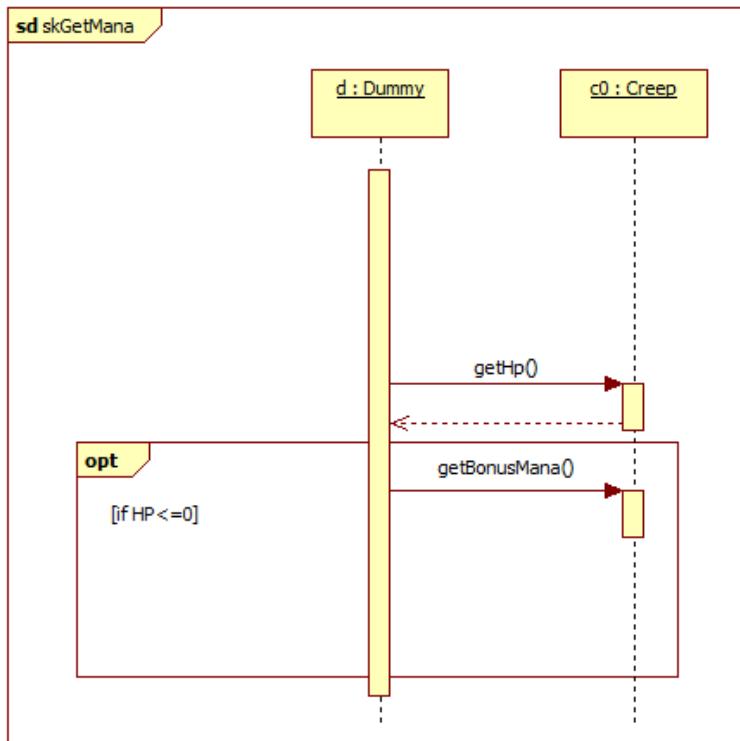
5.3.5 Build Obstacle



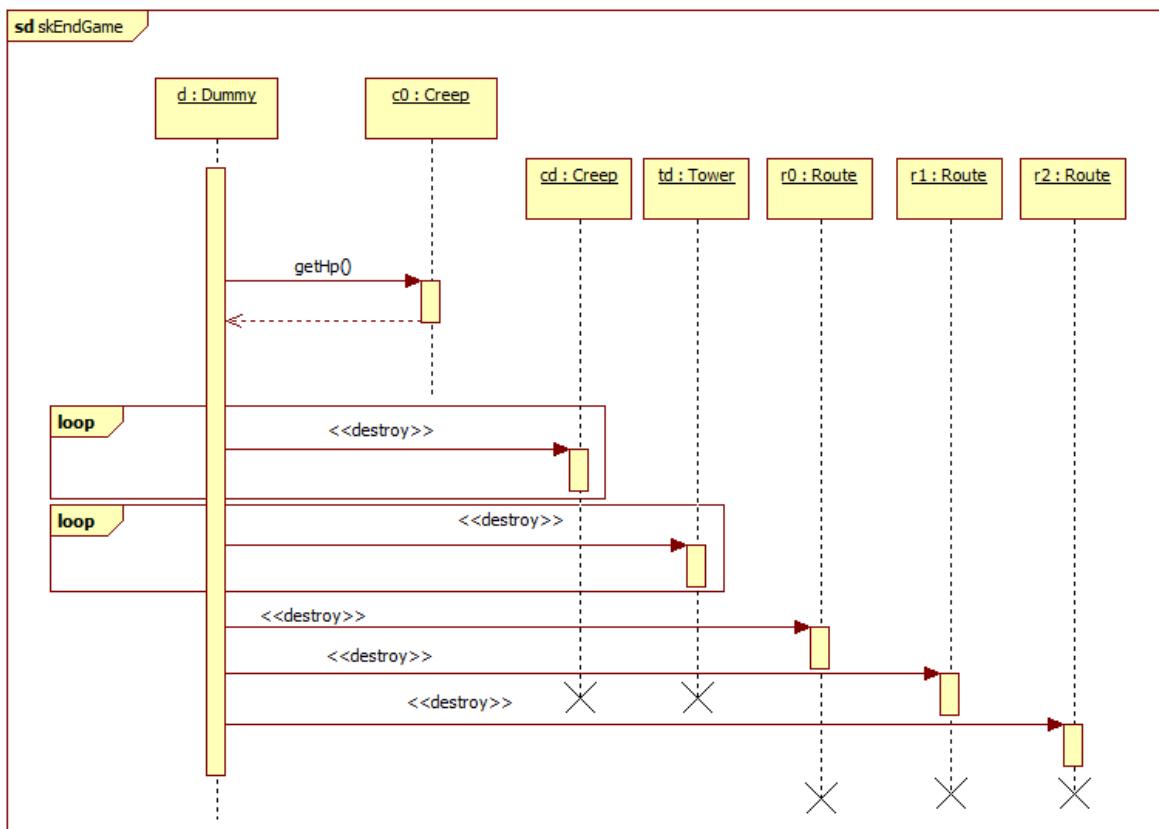
5.3.6 Upgrade Obstacle



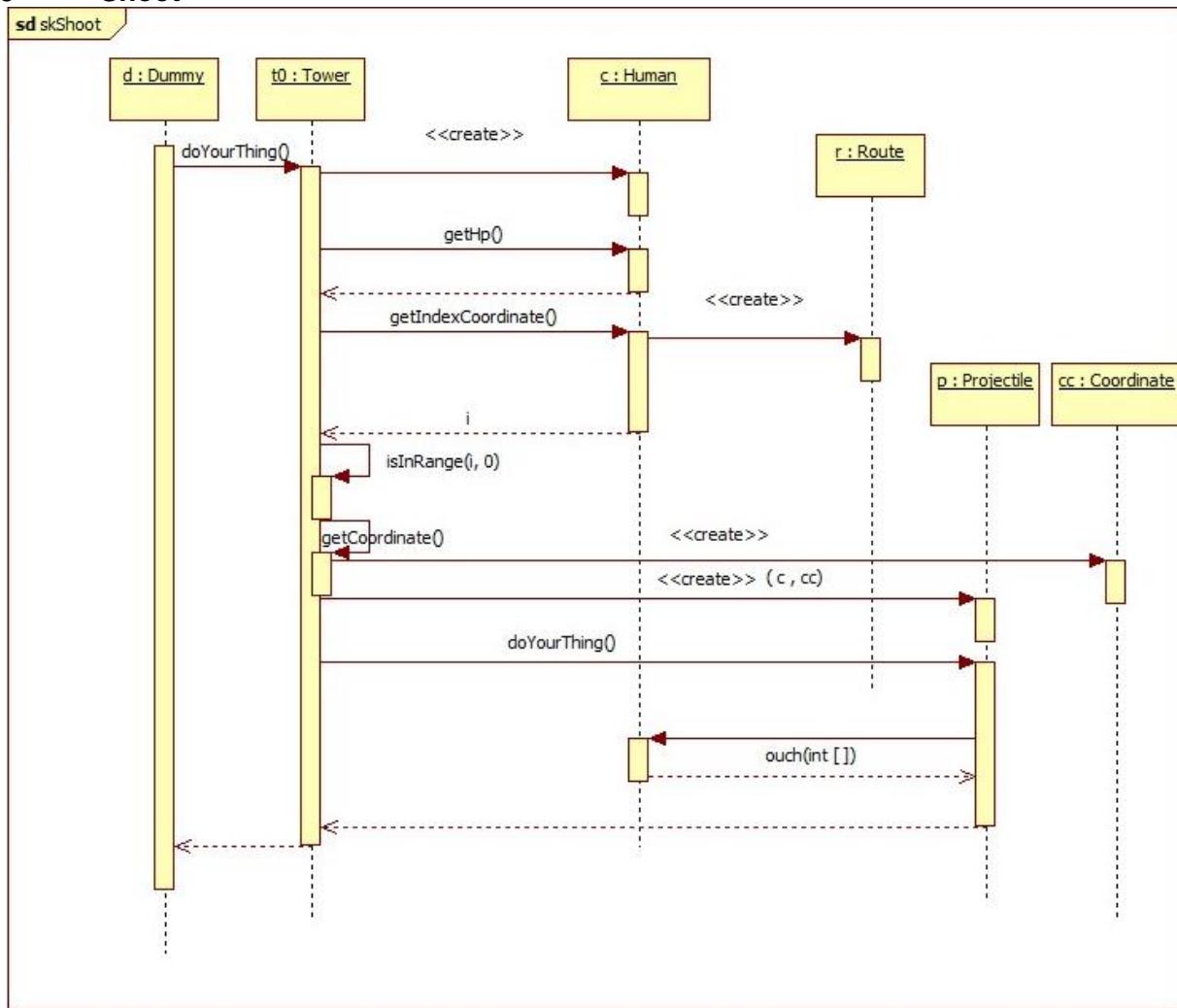
5.3.7 Get Mana



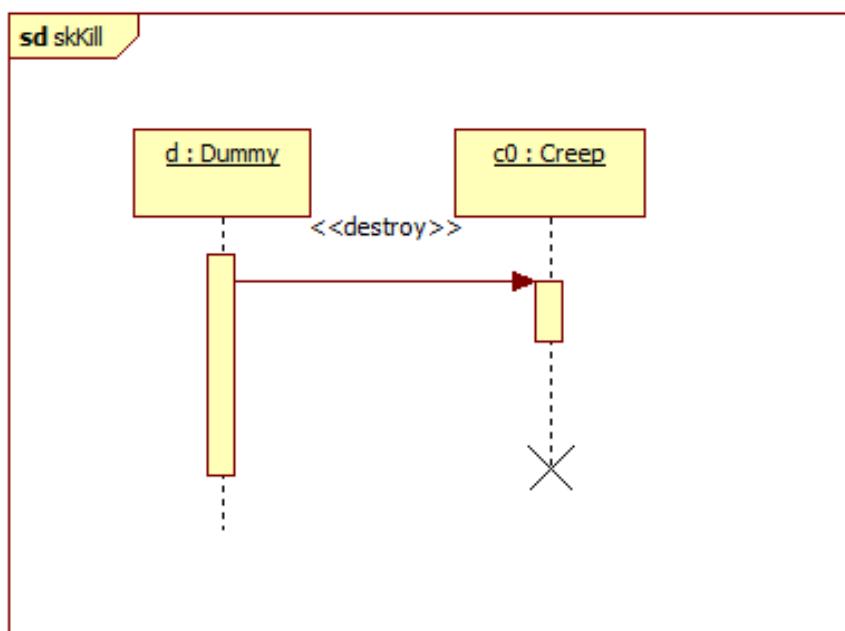
5.3.8 End Game



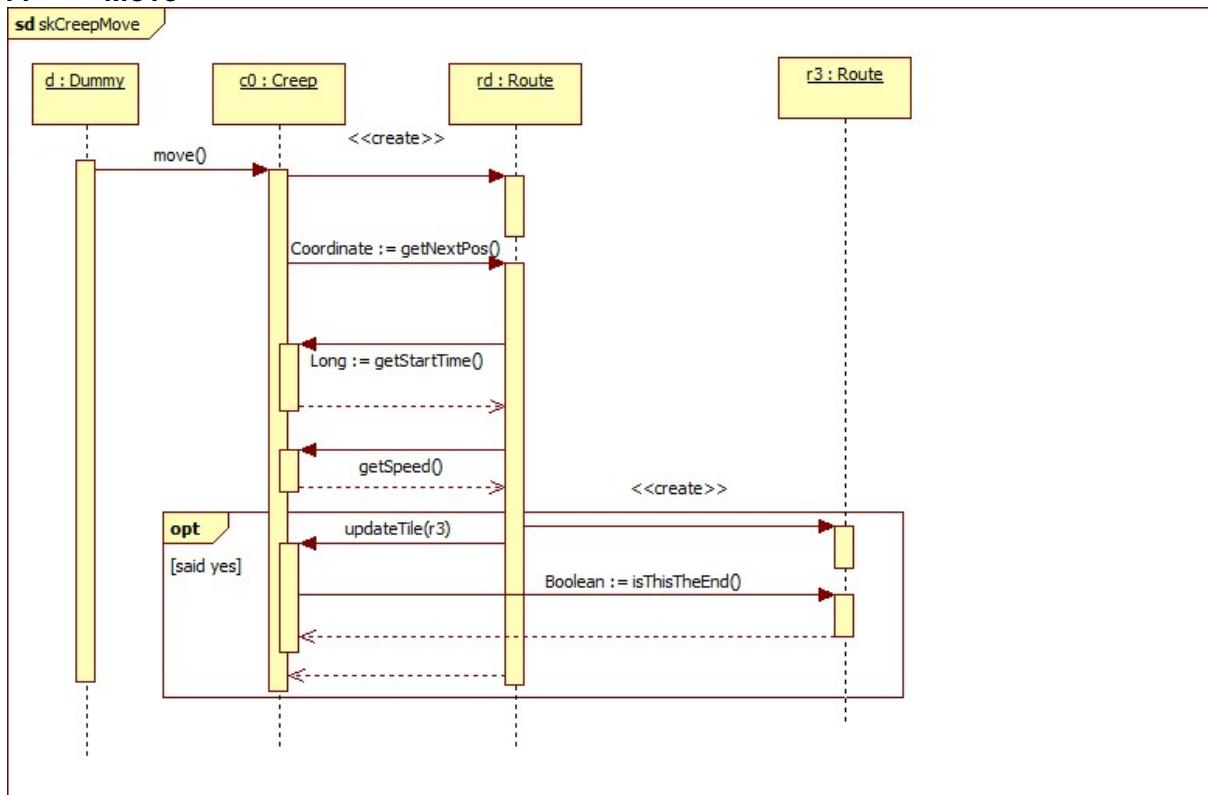
5.3.9 Shoot



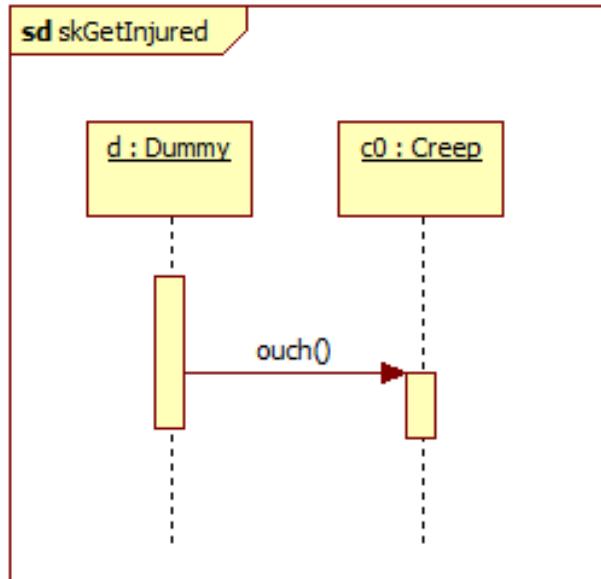
5.3.10 Kill



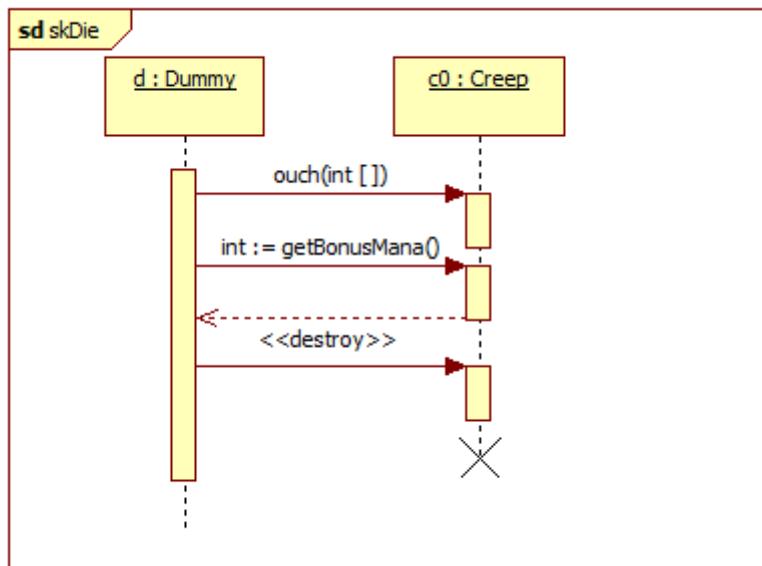
5.3.11 Move



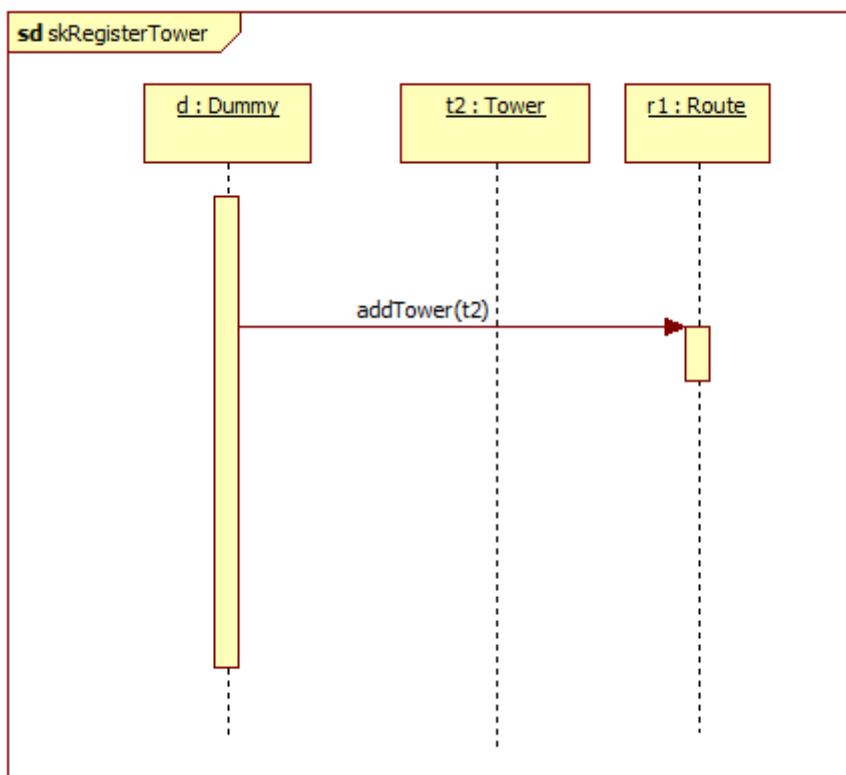
5.3.12 Get Injured



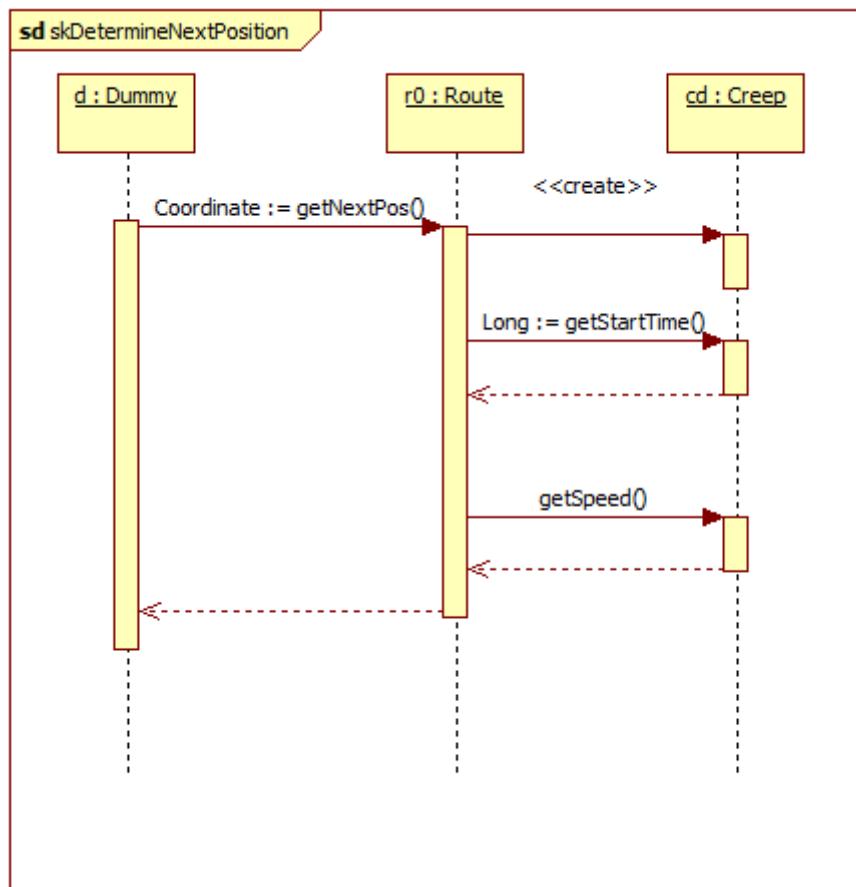
5.3.13 Die



5.3.14 Register Tower

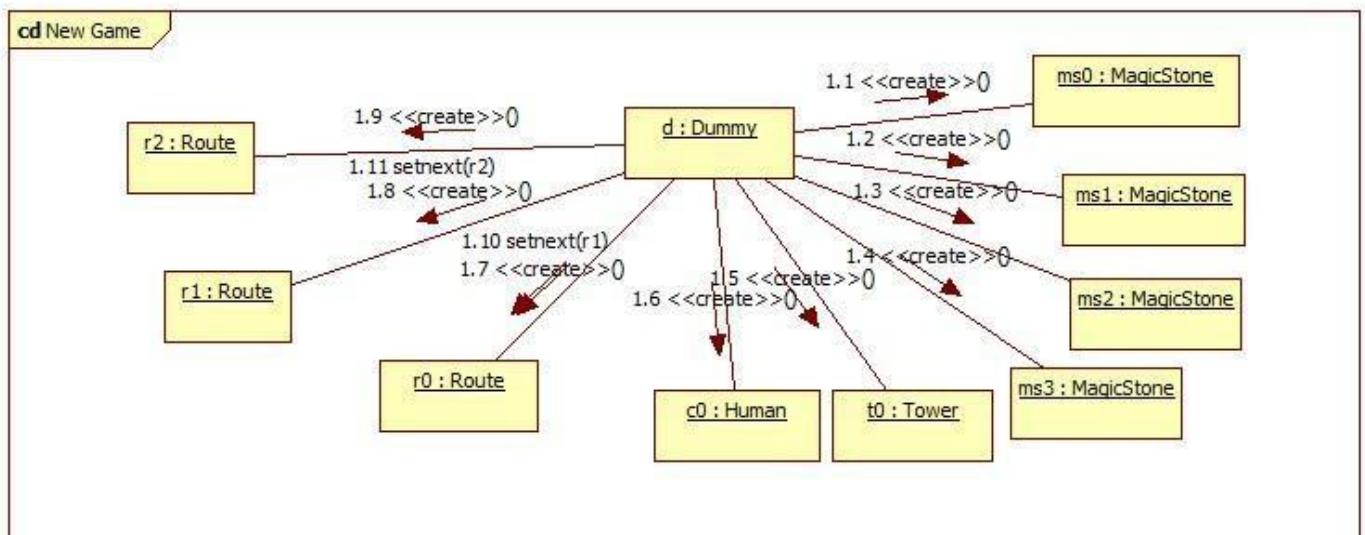


5.3.15 Determine Next Position

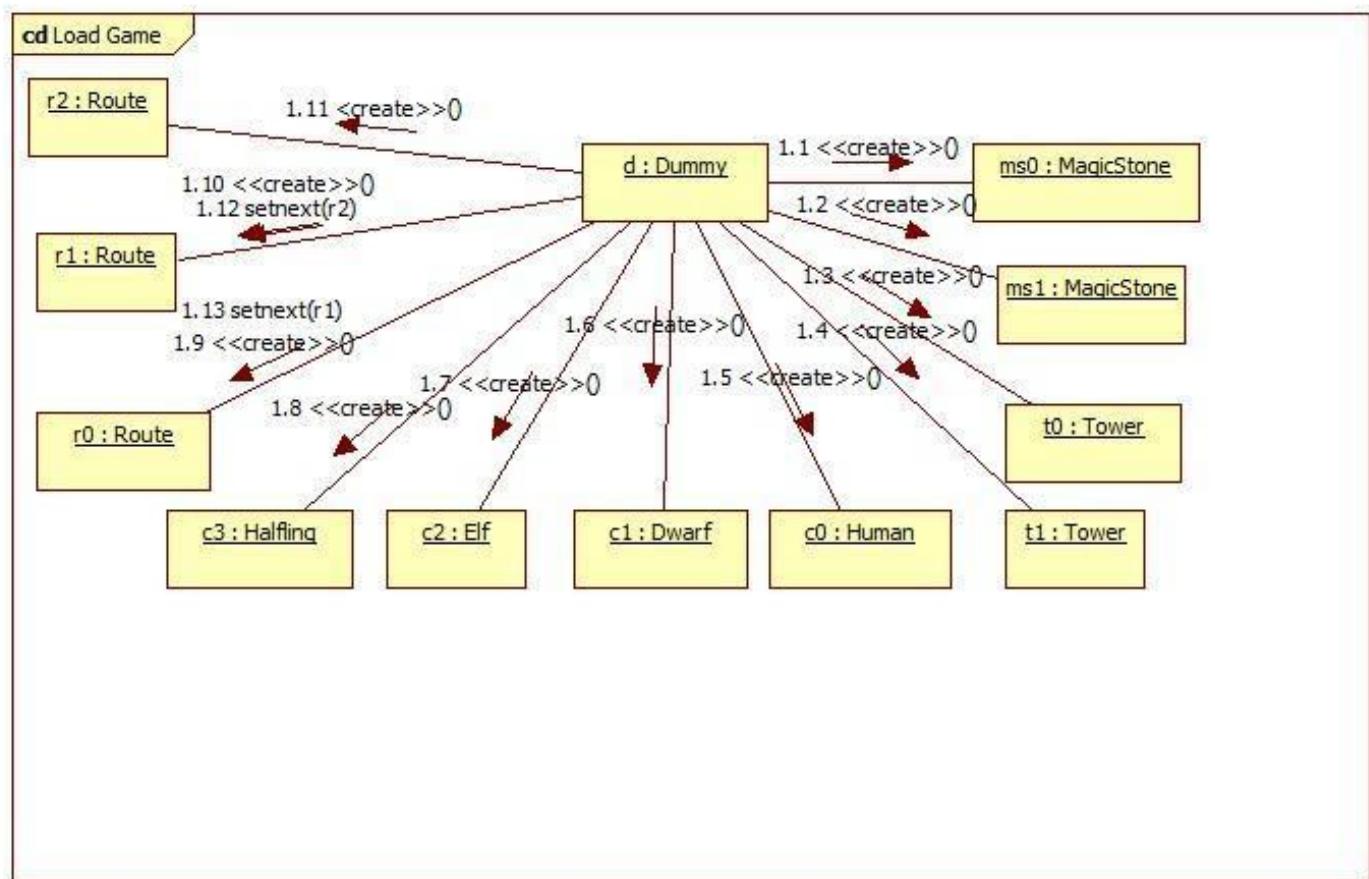


5.4 Kommunikációs diagramok

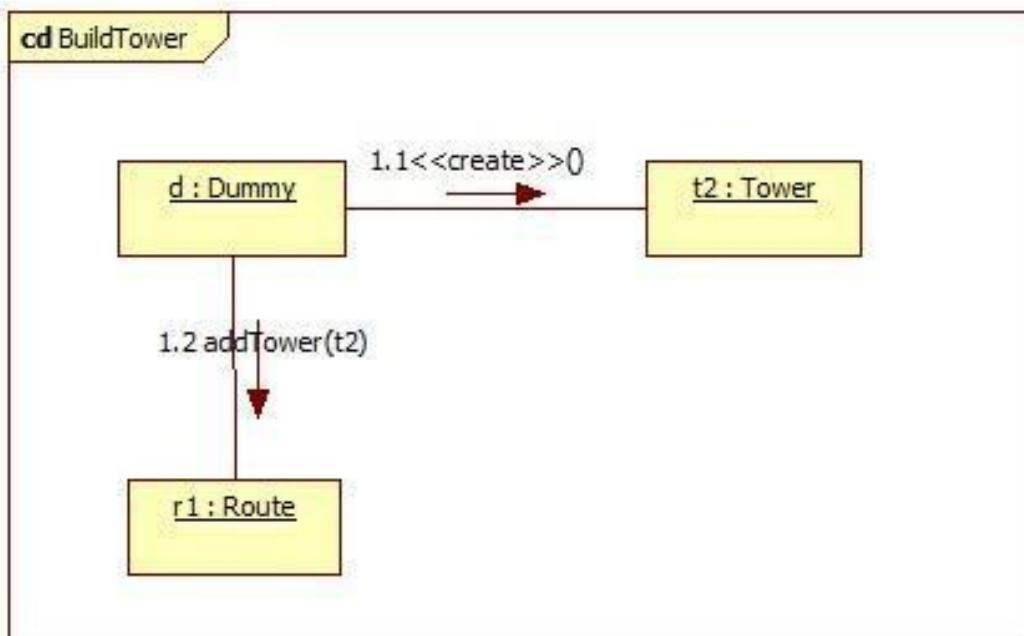
5.4.1 New Game



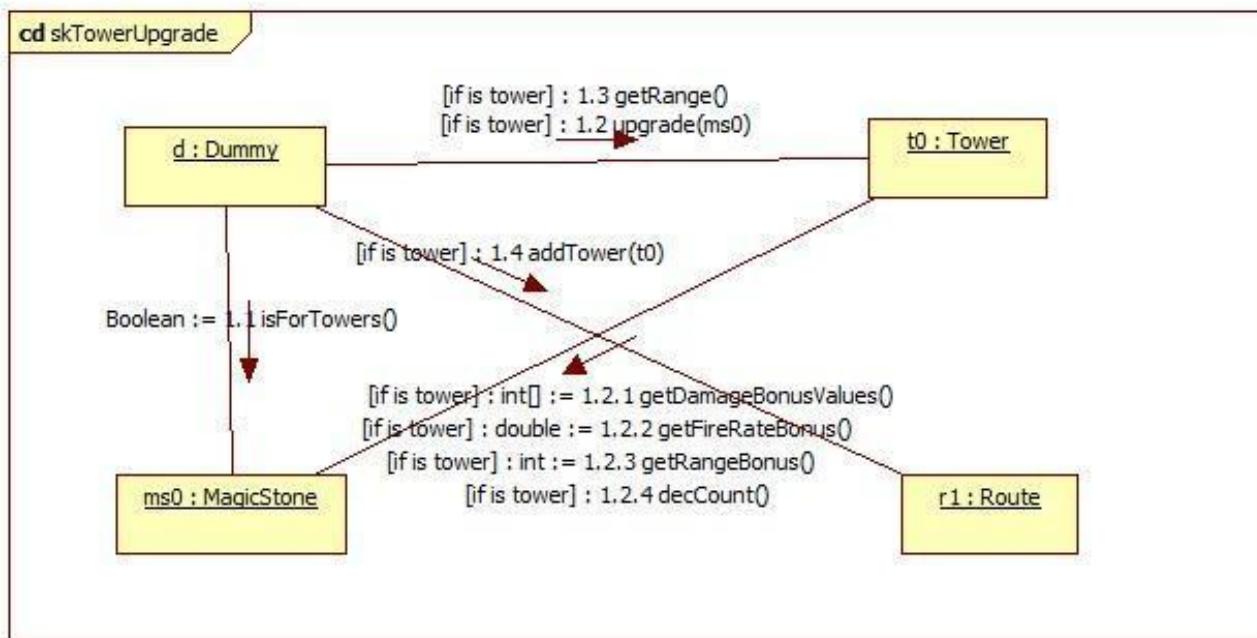
5.4.2 Load Game



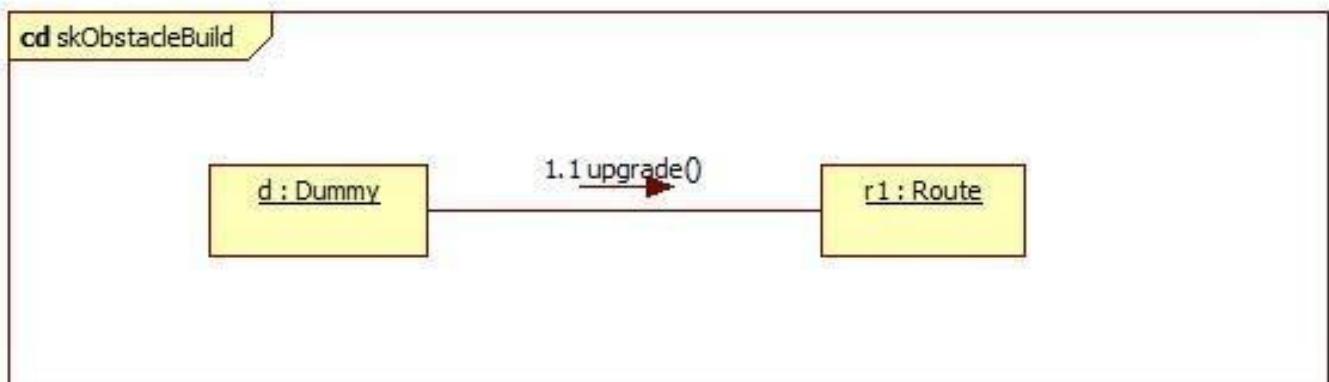
5.4.3 Build Tower



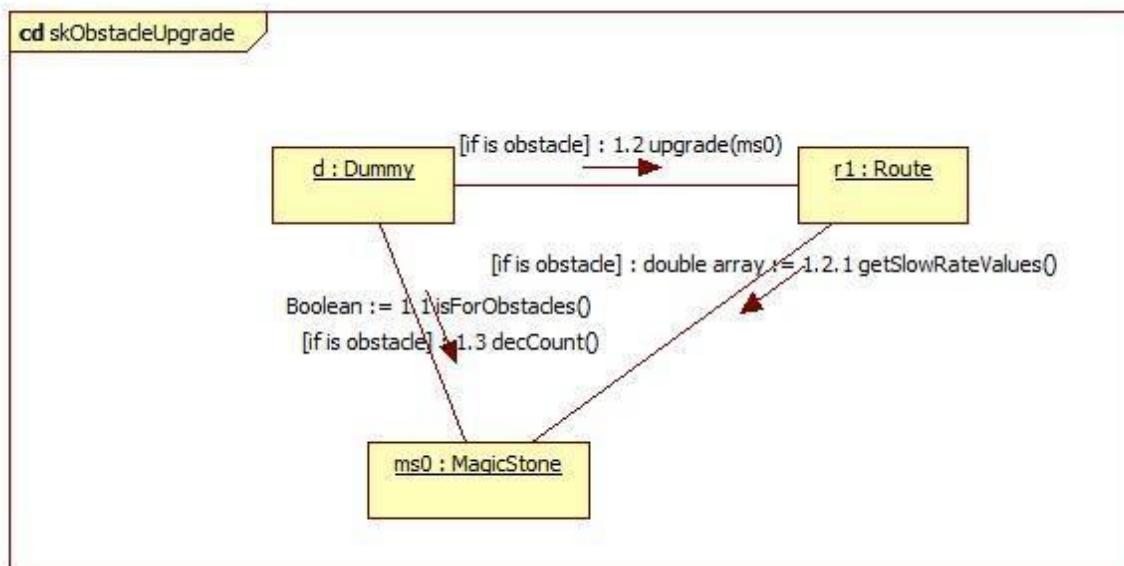
5.4.4 Upgrade Tower



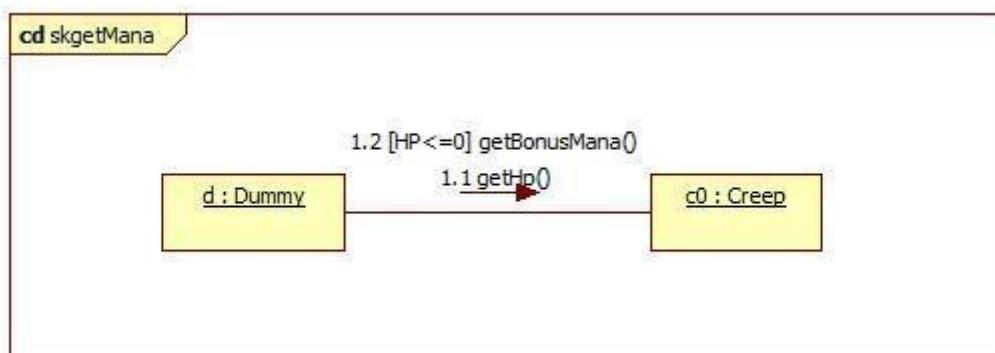
5.4.5 Build Obstacle



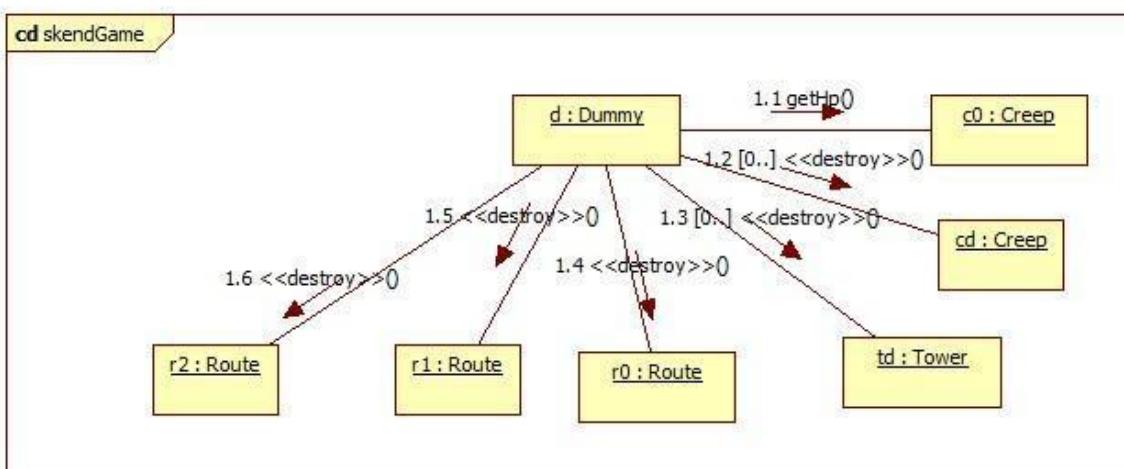
5.4.6 Upgrade Obstacle



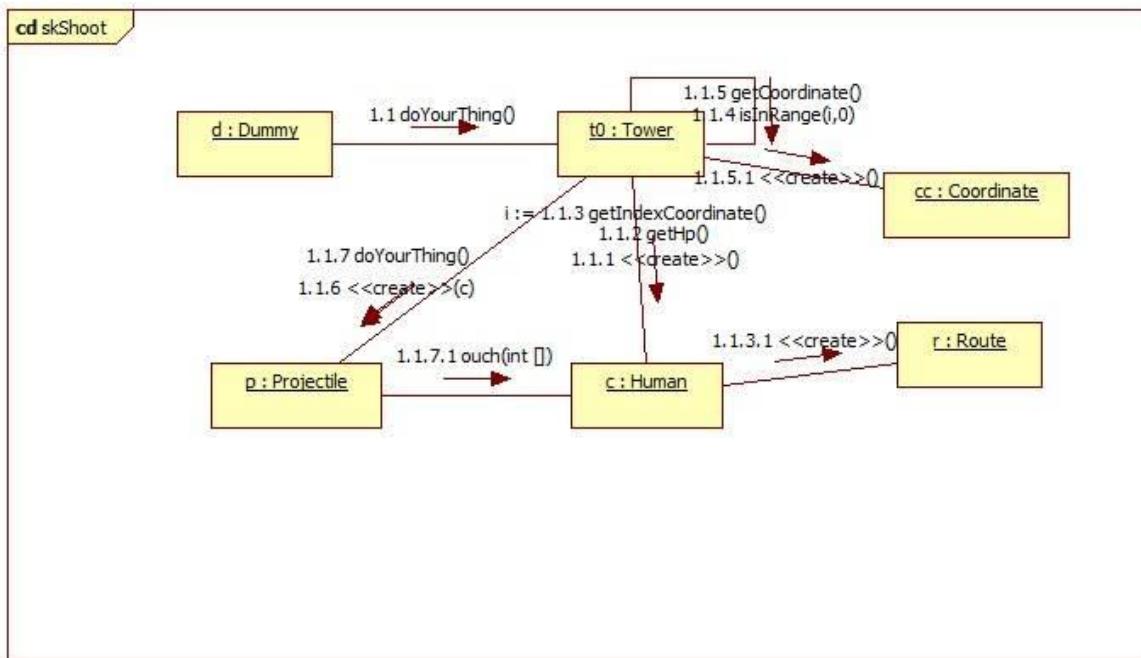
5.4.7 Get Mana



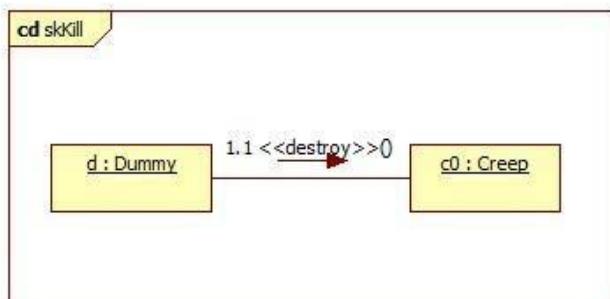
5.4.8 End Game



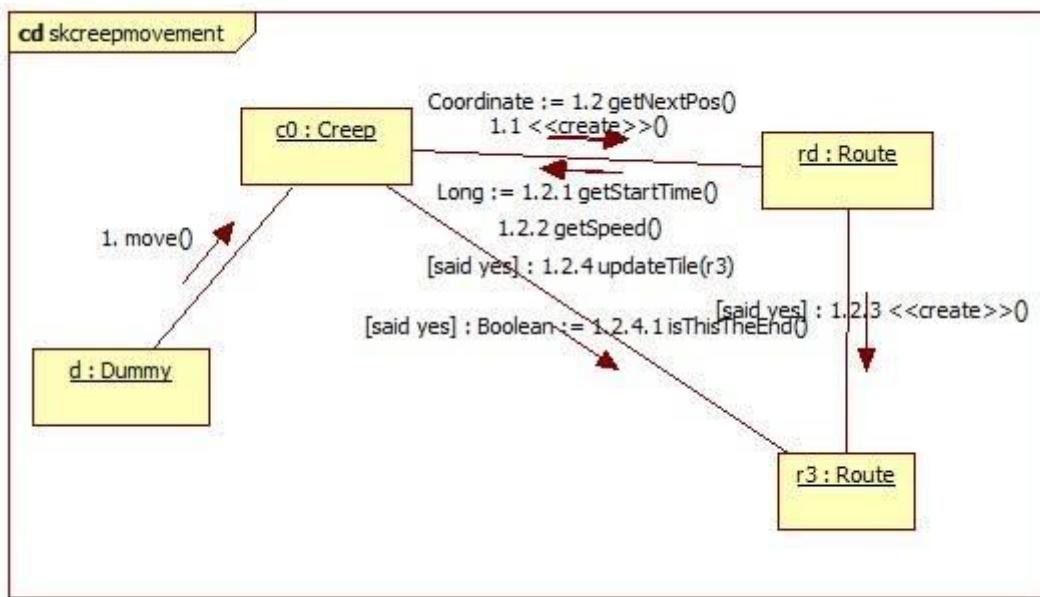
5.4.9 Shoot



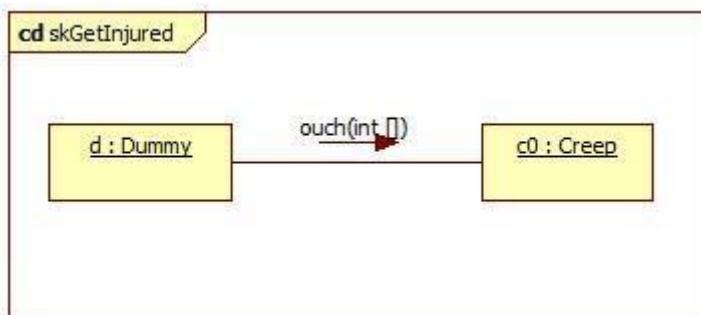
5.4.10 Kill



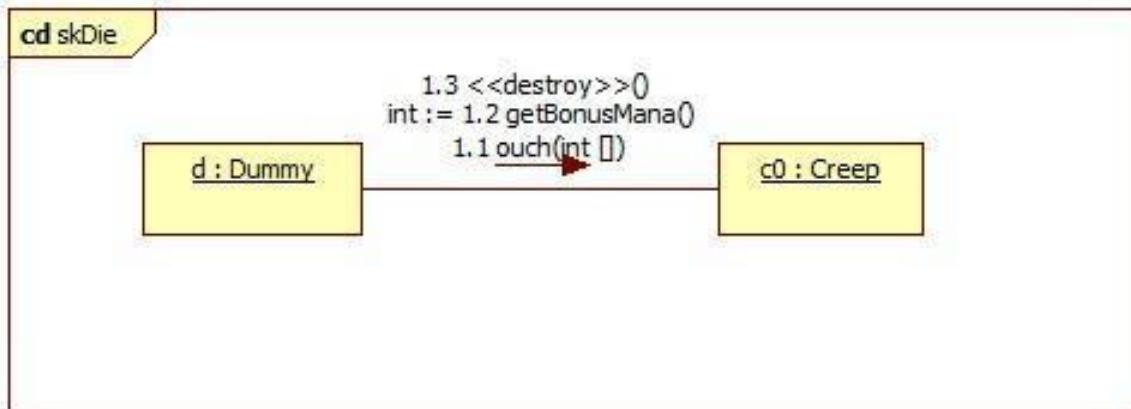
5.4.11 Move



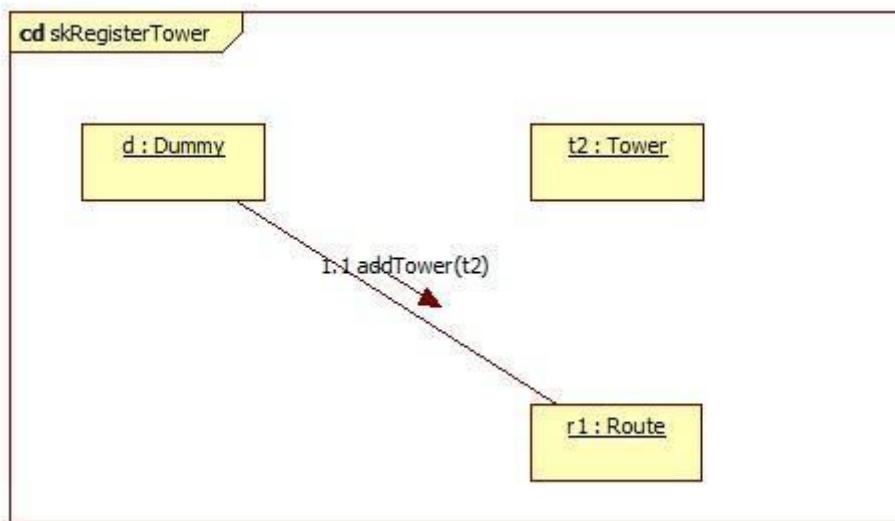
5.4.12 Get Injured



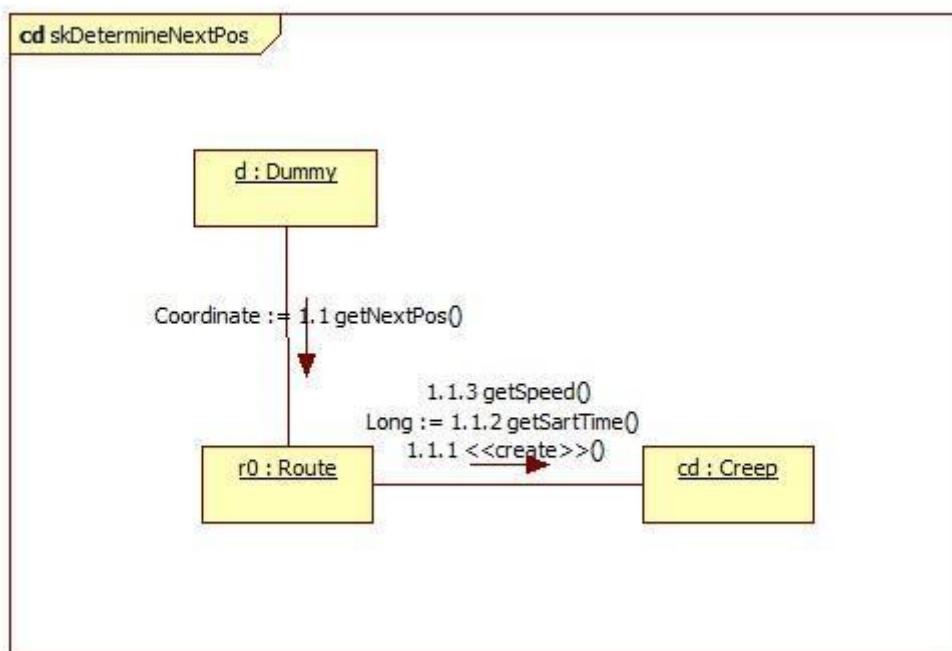
5.4.13 Die



5.4.14 Register Tower



5.4.15 Determine Next Position



5.5 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2014.03.14. 18:30	1 óra	Iklódi	use-case diagramok elkészítése
2014. 03.14. 21:15	1 óra	Molnár	use-case leírások elkészítése
2014. 03.15. 16:00	6 óra	Iklódi	szekvenciadiagramok megtervezése, forgatókönyv, szkeleton dialógus megírása, koordinálás
2014.03.15. 16:00	3 óra	Srajner	szekvenciadiagramok megszerkesztése
2014.03.15 20:00	3 óra	Németh	kommunikációs diagramok megszerkesztése

6. Szkeleton beadás

6.1 Fordítási és futtatási útmutató

6.1.1 Fájllista

Fájl neve	Méret	Keletkezés ideje	Tartalom
coordinate.java	1 KB	2014.03.23 10:57	Coordinate class és függvényei
creep.java	2 KB	2014.03.23 10:57	Creep class és függvényei
dummy.java	12 KB	2014.03.23 10:57	A skeleton kezelő class-ja
dwarf.java	1 KB	2014.03.23 10:57	Dwarf class és függvényei
elf.java	1 KB	2014.03.23 10:57	Elf class és függvényei
halfling.java	1 KB	2014.03.23 10:57	Halfling class és függvényei
human.java	1 KB	2014.03.23 10:57	Human class és függvényei
indexcoordinate.java	1 KB	2014.03.23 10:57	IndexCoordinate class és függvényei
magicstone.java	2 KB	2014.03.23 10:57	MagicStone class és függvényei
main.java	1 KB	2014.03.23 10:57	Main class
projectile.java	1 KB	2014.03.23 10:57	Projectile class és függvényei
route.java	2 KB	2014.03.23 10:57	Route class és függvényei
tile.java	1 KB	2014.03.23 10:57	Tile class és függvényei
tower.java	2 KB	2014.03.23 10:57	Tower class és függvényei

6.1.2 Fordítás

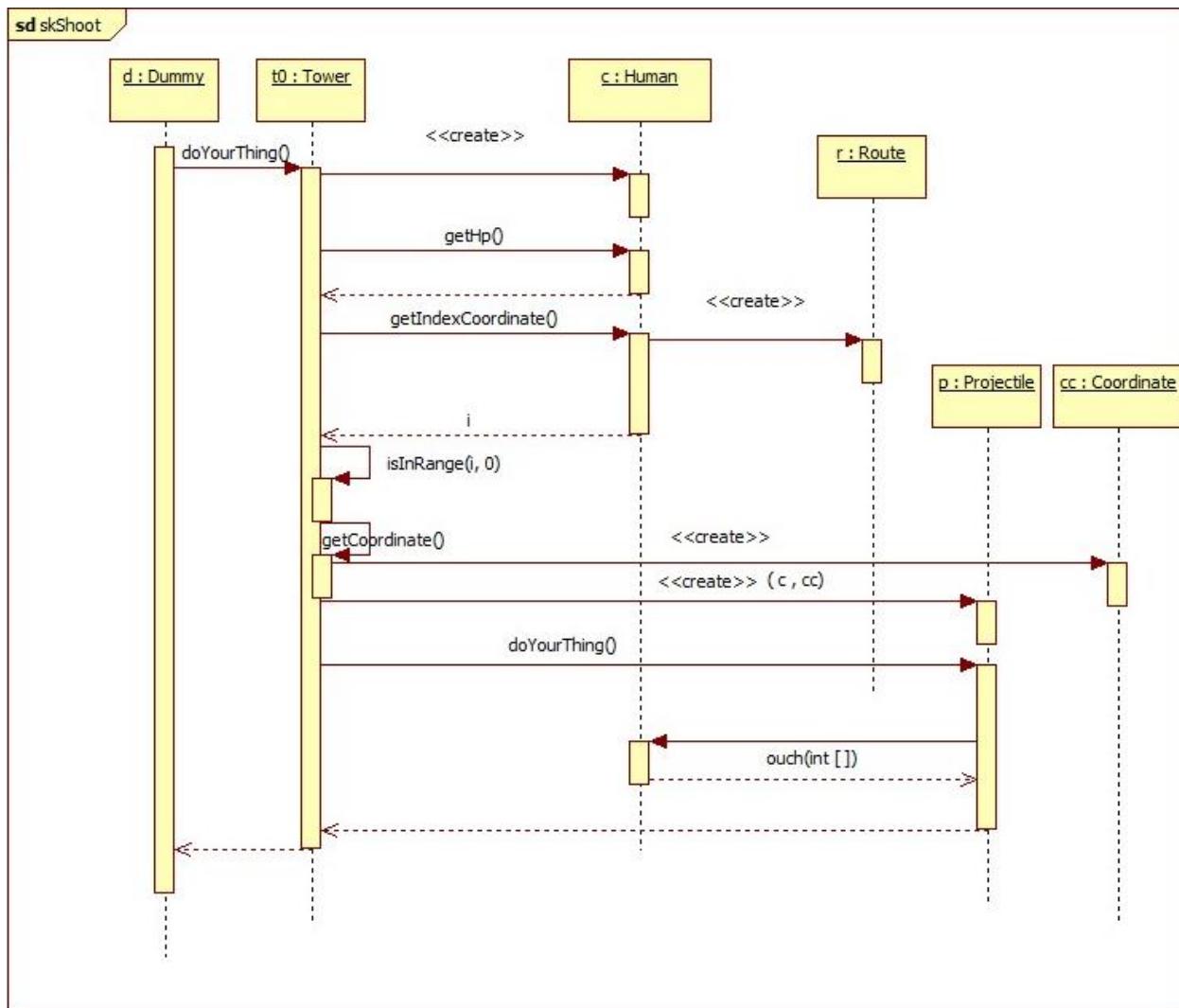
Az Eclipse fejlesztői környezet megnyitását követően új java projekt létrehozását követően be kell importálni a programhoz tartozó összes osztály *.java fájlait és ezt követően a “run” parancsra kattintva az alkalmazás elindul és a lenti konzol ablakban innentől vezérelhető tovább.

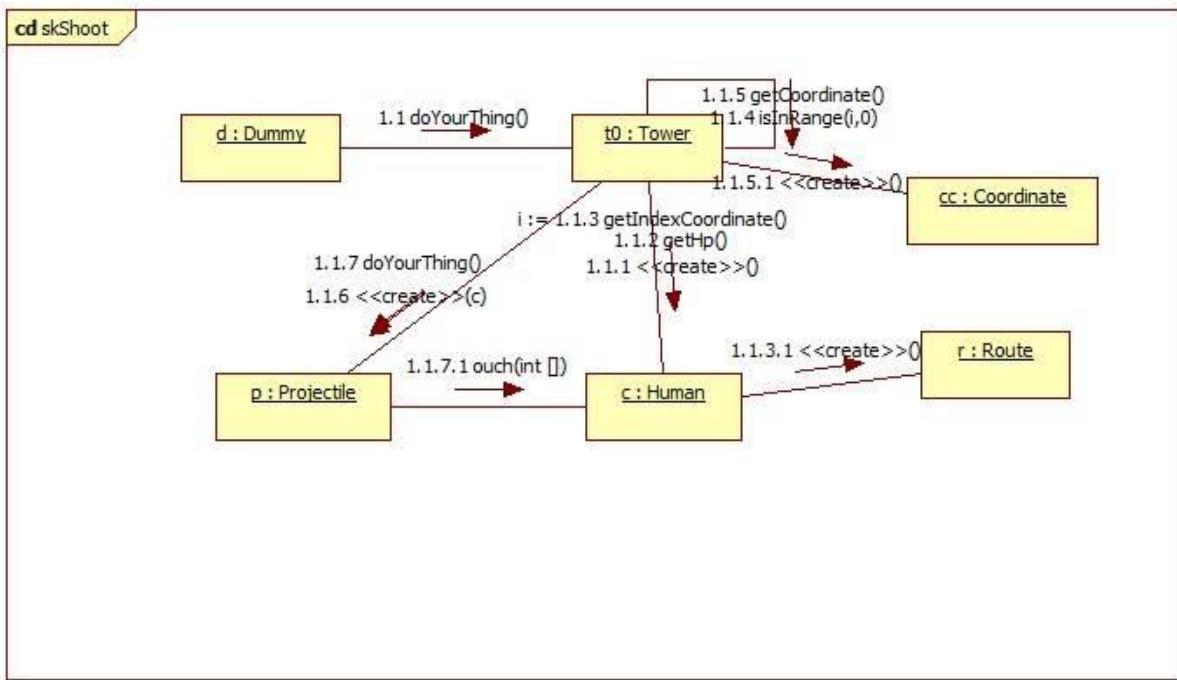
6.1.3 Futtatás

Az Eclipse alsó konzol menűjébe beírt parancsokkal lehet a programot vezérelni. Az egyes funkciókhoz a “help” parancs megadása ad segítséget.

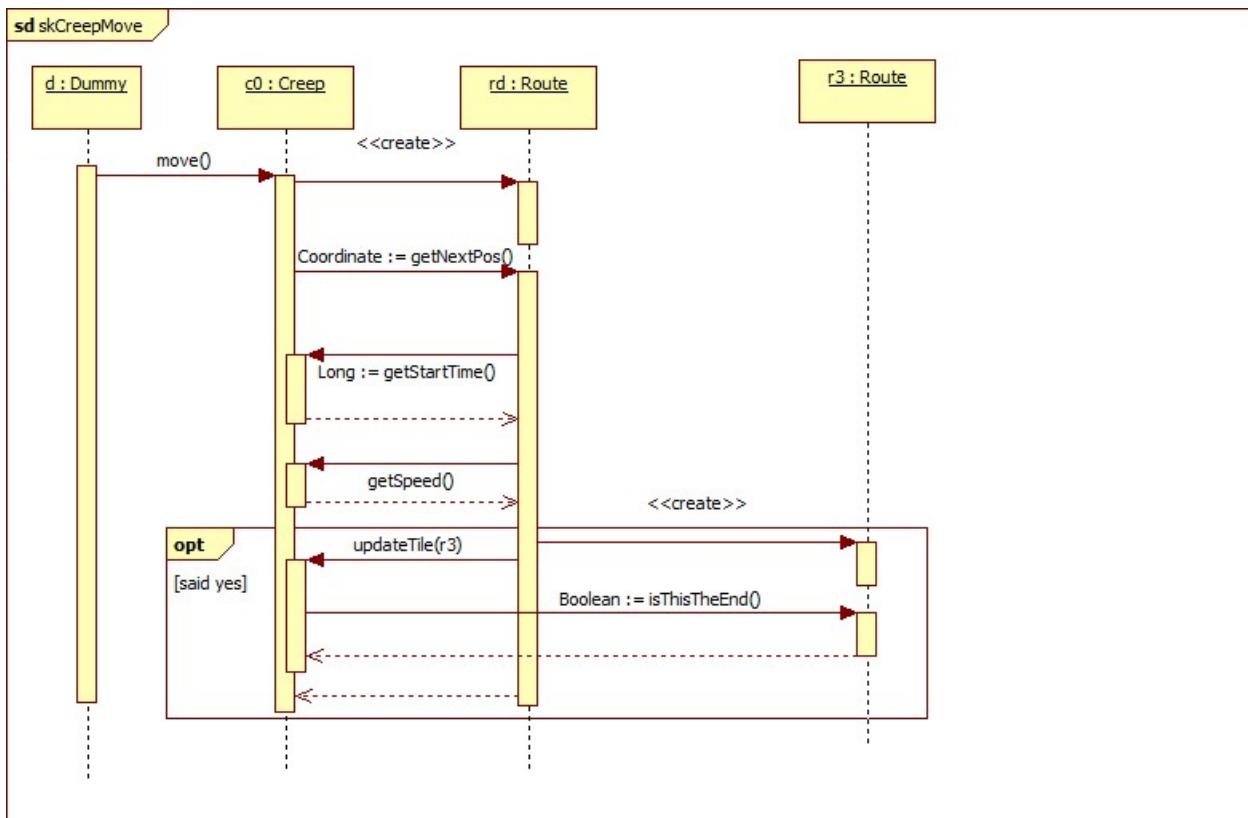
6.1.4 Ezt most ideírom: módosítások:

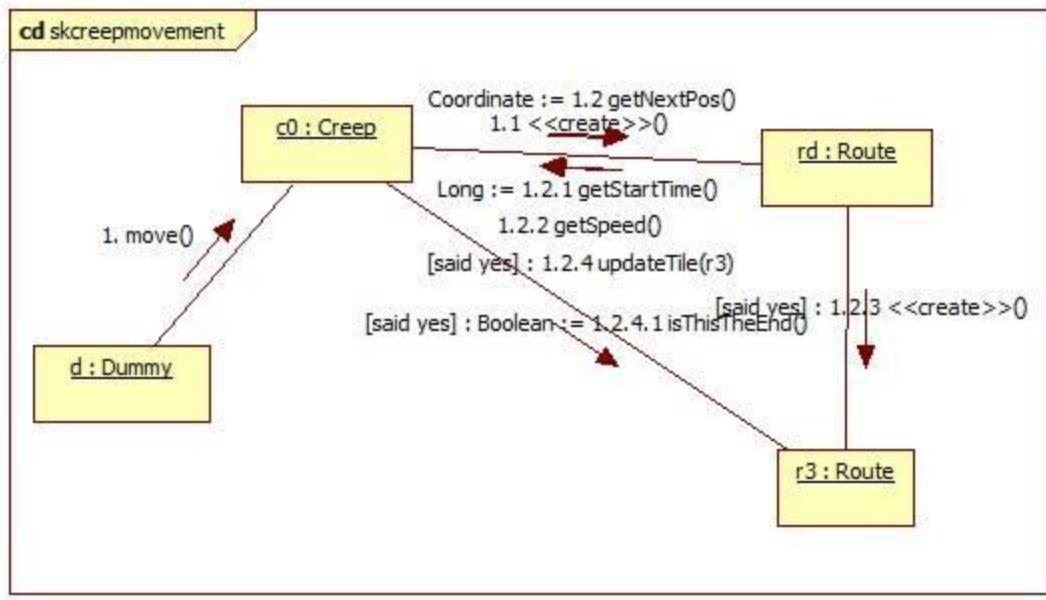
Shoot szekvencia:





Move szekvencia:





6.2 Értékelés

Tag neve	Munka százalékban
Iklódi Eszter	25,92 %
Molnár Bence	18,30 %
Németh Zsolt	17,96 %
Radnai Balázs	18,71 %
Srajner Ferenc	19,11 %

6.3 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2014.03.19. 08:15	2 óra	Iklódi Srajner Németh Molnár	szkeleton terv átbeszélése
2014.03.19. 10:20	1 óra	Iklódi Srajner Radnai	a szkelton program felületének lekódolása
2014.03.22. 12:00	6 óra	Radnai	a program megírása, tesztelése
2014.03.22. 14:00	1 óra	Molnár	kódolás közben felmerülő kérdések pontosítása
2014.03.23. 10.00	1 óra	Németh	Kommunikációs diagramok újrarendezése
2014.03.23. 10:00	2 óra	Radnai	program vélegesítése, szekvencia diagrammok módosítása

7. Prototípus koncepciója

7.1 Prototípus interface-definíciója

7.1.1 Az interfész általános leírása

A determinisztikusság érdekében a prototípus két eltérő állapotra bontható:

- **Alap állapotban** parancsokra vár. Amennyiben a parancs további bemeneteket vár, **segédleti kimenet** kerül a konzolra. A parancs végrehajtása után **log kimenetet** ír a a program a konzolra. Ezek bekerülnek egy log file-ba is. A következő állapotba a **Play** parancssal lehet eljutni, mely paraméterként egy időtartamot kap.
- **Futás állapotban** minden aktor a szerepe szerint végzi a műveleteit. Futás állapotban csak **log kimenet** van. A megadott idő letelte után a program automatikusan az alap állapotba kerül.

Input:

File bemenet:

1. Parancsok beadása:

.txt kiterjesztésű szöveges file, amiket **New Game** esetén lehet kiválasztani.

2. Pálya:

.dat kiterjesztésű file. **New Game** esetén az előre definiált .dat file-t tölti be, **Load Game** választása esetén a korábbi játékmenetek során mentett állapotot tölti vissza.

3. Config betöltése:

.ini kiterjesztésű file, előre definiált beállítások betöltésének lehetőségére

4. Wave betöltése:

.wav kiterjesztésű file, előre definiált hullámokat tartalmaz

Konzolos parancsok:

Az interfész tervezésekor figyelembe kell venni a leendő kész program bemeneteit, és ez alapján kialakítani a prototípus lehetséges bemeneteit.

- A program indításakor a felhasználónak egy menüből kell kiválasztania, hogy új, vagy már megkezdett játékot akar-e indítani. Ezen felül a kilépésre is módja van.
- A felhasználói parancsok vakon próbálják szimulálni, hogy a felhasználó mire lesz képes.
 1. Main Screen:
Eben az ablakban látjuk a játék pályáját (nem része a prototípusnak), valamint itt lehetséges a különböző tornyok, akadályok kiválasztása, és új védelmi eszköz lerakása esetén is itt választhatjuk ki a kívánt helyet.
 2. Stone List (shop):
Miután az egyes számú ablakban kiválasztottunk egy tile-t, ebben a listában soroljuk fel, a hozzá elérhető fejlesztő köveket.
 3. Show properties Text:
Amennyiben kiválasztottunk egy követ a listából, itt láthatók a kő tulajdonságai.
A Stone List kiegészítésére szolgál.
 4. Buy:
A kiválasztott követ hozzáadja a kijelölt védelmi egységhez. Ha ez megtörtént, levonódik a megfelelő mennyiségű varázserő, és az egység tulajdonságai a leírtak alapján módosulnak.

5. Build Tower:
Új torony lerakás, miután lenyomtuk a gombot, megadhatunk a Main Screen-ben a torony koordinátait. Amennyiben helyes koordinátát adtunk meg, levonódik a megfelelő mennyiségű varázserő, és torny épül az adott helyre.
 6. Build Obstacle:
Hasonló mint a Build Tower, csak akadály lehelyezésére.
 7. Quit:
Lehetőséget ad a kilépésre, és megkérdezi, hogy mentse-e az eddigi játékot.
- Ezekben a parancsokon felül, a tesztelés megkönnyítése végett úgynevezett debug parancsokat is definiálunk. Ezek segítségével tudjuk az összes lehetséges kimenetet tesztelni.

Mivel a prototípusban csak konzolos parancsok megadása lehetséges, ezért egyszerű karakteres bemenetre képezzük le a fent megadott bemeneteket (lást a bemeneti nyelvnél).

Output:

File kimenet:

1. Log-ok mentése:
.log kiterjesztésű file, a parancsok során keletkezett log kimeneteket ide is kiírja a program.
2. Játék mentése:
.dat kiterjesztésű file, a játék folytatásához készített mentés, a **Quit** parancs esetén a save opción választásakor keletkezik a file.

7.1.2 Bemeneti nyelv

A konzol a következő parancsokra reagál:

Menü parancsok:

New Game

Leírás: Új játék kezdése

Opciók: A következő információkat kell megadni a program számára:

- olvasson-e parancsokat file-ból, ha igen, melyik file-ból
- meg kell neki adni, hogy melyik wave file-t töltse be

Load Game

Leírás: A már megkezdett pálya betöltése.

Opciók: Meg lehet adni, hogy melyik mentést töltök be.

Exit

Leírás: Kilép a programból

Opciók: Nincs.

Felhasználói parancsok:**addTower**

Leírás: Új torony lehelyezése a pályán

Opciók: meg kell adni egy érvényes koordinátát. (mező, torony nélkül)

addObstacle

Leírás: Új akadály létrehozása.

Opciók: Meg kell adni egy érvényes koordinátát. (út, akadály nélkül)

selectTile

Leírás: A *Main Screen* egér bal gomb lenyomására való választ szimulálja. Kiválaszt egy pontot a pályán, majd felsorolja a lehetséges opciókat (a **showStone**, és **buyStone** parancsokhoz).

Opciók: Meg kell adni a tile pontos koordinátáját, ha a felhasználó ezt vissza szeretné vonni, ismét be kell írnia az előző koordinátákat

showStone

Leírás: A **selectTile** után használható parancs, a választott kő tulajdonságait adja meg. A *Stone List* köré kattintását szimulálja.

Opciók: Egy érvényes kő kiválasztása.

buyStone

Leírás: A **selectStone** után használható parancs, megvásárolja a követ.

Opciók: Egy érvényes kő kiválasztása.

Quit

Leírás: Kilép a megkezdett játékból, vissza a főmenübe.

Opciók: El kell dönten, hogy mentsük-e a játékot

Debug parancsok:**help**

Leírás: Kiírja a lehetséges parancsok nevét

Opciók: Nincs.

debugFog

Leírás: Állítja a köd állapotot.

Opciók: Ki lehet választani, hogy legyen-e köd a pályán, valamint vissza lehet állítani az alap játék-beli állapotba. (3 állapot)

debugSpecialHit

Leírás: Állítja a lövedékek SpecialHit állapotát.

Opciók: Ki lehet választani, hogy a lött lövedékek kettéválasszák-e az ellenfelet, vagy ne, valamint vissza lehet állítani alap állapotba. (3 állapot)

debugSelectRoute

Leírás: Állítja az elágazásoknál levő következő választott utat. (az összes elágazást módosítja)

Opciók: Hárrom lehetőség: az alap állapot, ahol véletlenszerűen halad tovább az ellenfél, valamint a fix esetek, ekkor vagy az első, vagy a második úton halad tovább.

debugGodMode

Leírás: Elegendő varázserőt ad bármelyik forgatókönyv lefuttatásához.

Opciók: Nincs.

debugSetMana

Leírás: A kért varázserőt ad a játékos számára.

Opciók: A felhasználónak be kell írnia a kért mennyiséget.

Play

Leírás: Elindítja a játékot.

Opciók: Meg kell adni másodpercen, hogy a játék meddig menjen.

Az opciókkal rendelkező parancsok beírása esetén a program további információt vár el a felhasználótól. A várt információt a **segédleti kimenettel** határozza meg, és felsorolja a lehetőségeket sorszámozva. Ez esetben a parancs érvényesítéséhez meg kell adni egy érvényes választ (opciók esetén minden esetben egész szám).

7.1.3 Kimeneti nyelv

A prototípusban két féle kimenetet definiálunk:

-A **log kimenet** a program parancsaira való válaszát adja, valamint a program belső aktorok állapotát részletezi futási állapotban. **A kimenet a konzolon kívül egy file-ba is íródik.**

-A **segédleti kimenet** a felhasználót segíti a program irányításában. Opcióval rendelkező parancsok esetén a konzolra írja a lehetséges opciók értékét, ami alapján a felhasználó választhat. **A kimenet csak a konzolra íródik.**

A kommunikáció a következő képpen folyik a programmal:

bemenet konzolra: Parancs

kimenet konzolra: Opció esetén a lehetőségek felsorolása konzolra. A lehetőségeket egész számok reprezentálják, ezt kell bemenetként adnia a felhasználónak a következő parancsra. (Három parancs esetén két számot (koordinátát) kell megadni, ezek: az **addTower**, **addObstacle**, **selectTile**)

bemenet konzolra: Opció esetén a megfelelő lehetőség száma.

kimenet konzolra: Rossz opció esetén a probléma részletezése, amennyiben egyértelműen nem végezhető el a parancs új parancsra vár. (például nincs elég varázserő az építéshez). Alapértelmezett esetben, ha nem megfelelő bemenetet kapott a parancs visszalép a bemenet konzolra állapothoz.

kimenet konzolra és file-ba: A parancs által történt változtatásokat látható formában dokumentálja. A kimenetet hozzáfüzi a megfelelő kimeneti file-okhoz a játék jelenlegi idejének feltüntetésével. A könnyebb átláthatóság érdekében az egyes aktoroknak saját log file-ja is van, így könnyebben le lehet tesztelni az egyes osztályok működését. A következő log file-okat definiáljuk:**main.log** (ide minden log kimenet kiíródik, továbbá ennek a tartalmát látjuk a konzolon is), **program.log** (játékfüggetlen adatok), **route.log**, **saruman.log**, **tower.log**, **projectile.log**, **creep.log**.

A **Play** parancs máshogyan viselkedik:

bemenet konzolra: Play

kimenet konzolra: Hány másodpercig menjen a játék?

bemenet konzolra: idő megadása.

kimenet konzolra és file-ba: Ekkor a játék főciklusa fut, így a belső aktorok a játék mechanikája szerint végzik a dolgukat. minden eseményhez hozzáfüzi a program az eddig eltelt időt a játék megkezdése óta.

kimenet konzolra: Ha letelt a megadott idő, a program azt jelzi a felhasználónak a konzolon, valamint visszalép alapállapotba.

Parancsok részletezése:

Parancs	New Game
segédleti kimenet	-Melyik file-ból olvasson bemenetet? -Melyik wave-et töltse be?

Parancs	Load Game
segédleti kimenet	-Hányas mentést töltsem be? - lehetőségek felsorolása

Parancs	Exit
segédleti kimenet	-

Parancs	addTower
segédleti kimenet	-Adj meg egy 2D-s koordinátát (szóközzel elválasztva) <i>Helyes koordináta esetén:</i> Elegendő varázserővel: -Torony került a (szám) (szám) koordinátára. <i>Kevés varázserő esetén:</i> -Nincs elég varázserő az építéshez. <i>Helytelen koordináta esetén:</i> -Érvénytelen koordináta.

Parancs	addObstacle
segédleti kimenet	-Adj meg egy 2D-s koordinátát (szóközzel elválasztva) <i>Helyes koordináta esetén:</i> Elegendő varázserővel: -Akadály került a (szám) (szám) koordinátára. <i>Kevés varázserő esetén:</i> -Nincs elég varázserő az építéshez. <i>Helytelen koordináta esetén:</i> -Érvénytelen koordináta.

Parancs	selectTile
segédleti kimenet	<p>-Adj meg egy 2D-s koordinátát (szóközzel elválasztva)</p> <p><i>Helyes koordináta esetén:</i></p> <ul style="list-style-type: none"> -<i>Ez egy (torony akadály) a következőket teheted vele:</i> - (<i>Felsorolja a lehetséges köveket</i>) <p><i>Ha az előzővel azonos a koordináta:</i></p> <ul style="list-style-type: none"> -<i>Jelenleg nincs kiválasztott tile.</i> <p><i>Helytelen koordináta esetén:</i></p> <ul style="list-style-type: none"> -<i>Érvénytelen koordináta.</i>

Parancs	showStone
segédleti kimenet	<p>-A kövek tulajdonságai:</p> <p><i>tulajdonságok</i></p>

Parancs	buyStone
segédleti kimenet	<p>-Add meg hányas sorszámu követ szeretnéd megvásárolni</p> <p><i>Helyes válasz esetén:</i></p> <ul style="list-style-type: none"> -A (i) kő meglett vásárolva <p><i>Helytelen válasz esetén:</i></p> <ul style="list-style-type: none"> -Ilyen (i) kő nem vásárolható

Parancs	Quit
segédleti kimenet	<p>-Kívánja menteni a játékot?</p> <p><i>igen esetén:</i></p> <ul style="list-style-type: none"> -Melyik slotra mentsem? <p><i>(slotok fesorolása)</i></p> <ul style="list-style-type: none"> -A mentés sikeresült -Főmenü: <p><i>(főmenü elemeinek kiírása)</i></p>

Parancs	help
segédleti kimenet	<p>-A következő elérhető parancsok vannak a számodra:</p> <p><i>parancsok</i></p>

Parancs	debugFog
segédleti kimenet	<ul style="list-style-type: none"> -Milyen állapotba szeretnéd állítani a ködöt? -1: van köd; -1: nincs köd; 0: automatikus <p>Ha érvényes állapotot választott:</p> <ul style="list-style-type: none"> -(i) állapot be lett állítva <p>Ha érvénytelen állapotot választott:</p> <ul style="list-style-type: none"> -(i) állapot nem létezik

Parancs	debugSpecialHit
segédleti kimenet	<ul style="list-style-type: none"> -Melyik lövedék állapotot választja? (1,0,-1) 0, Alap 1, minden lövedék kettéosztja az eltalált ellenfelet -1, Egyik lövedék sem oszt ketté

Parancs	debugSelectRoute
segédleti kimenet	<ul style="list-style-type: none"> -Melyik útvonalon szeretnél tovább haladni út kereszteződésben? -0: random; 1: jobbra; -1:balra <p>Ha helyes utat választ:</p> <ul style="list-style-type: none"> -A kereszteződésből (i) irányba haladsz tovább <p>Ha helytelen utat választ:</p> <ul style="list-style-type: none"> -Nem létezik ilyen út a kereszteződésben

Parancs	debugGodMode
segédleti kimenet	-A játékos varázsereje 9999-re nőtt.

Parancs	debugSetMana
segédleti kimenet	<ul style="list-style-type: none"> -Mennyi manát kérsz? <p>Érvényes érték esetén:</p> <p>Okés.</p> <p>Érvénytelen érték esetén:</p> <p>Érvénytelen adat.</p>

Ezek alapján a lehetséges log kimenetek konkrét definiálása:

Parancsok részére:

New Game:

"Game initialized" wavefilename
 routeid "created" idxcoord, prev
 routeid "next route added" routeid
 "magicstone" magicstoneid "created"
 "program started" configfilename
 "Game started"

Load Game:

A New Game-hez kiegészítve:
 "Game loaded" savefilename
 "-"

Exit:

"Game exited"

addTower:

(relative time) towerid "created" idxcoord
 Ha nincs elég varázserő:
 (relative time) "not enough mana"

addObstacle:

(relative time) routeid "obstacle built"
 Ha nincs elég varázserő:
 (relative time) "not enough mana"

selectTile:

"Tile selected" idxcoord

showStone:

value "stone shown"

buyStone:

value "stone bought"
 Ha nincs elég varázserő:
 (relative time) "not enough mana"

Quit:

"Left game"
 ha mentett:
 "Game saved" savefilename

debugFog:

(debug-value) "set to" value

debugSpecialHit:

(debug-value) "set to" value

debugSelectRoute:
 (debug-value) "set to" value

debugGodMode:
 (debug-value) "set to" value

debugSetMana:
 (debug-value) "set to" value

Play:
 "Game started for" value "seconds"
 majd ha visszalép az alapállapotba:
 "Game paused"

Belső aktorok részére, a különböző log file-okban a következő commandok keletkezhetnek:

creep.log:
 (relative time) creepid "created" routeid, lvl, race
 (relative time) creepid "moved to" routeid, slowrate
 (relative time) creepid "injured" projectileid,damage
 (relative time) creepid "died" projectileid, mana
 (relative time) creepid "slowrate changed" routeid, slowrate

tower.log:
 (relative time) towerid "created" idxcoord
 (relative time) towerid "projectile created:" projectileid
 (relative time) towerid "new target:" creepid
 (relative time) towerid "target lost" died/got away
 (relative time) towerid "upgraded" MSid changed_values

route.log:
 (relative time) routeid "obstacle built"
 (relative time) routeid "upgraded" MSid changed_values

projectile.log:
 (relative time) projectileid "created" creepid dealt damage special/notspecial
 (relative time) projectileid "i've finally arrived"

saruman.log:
 (relative time) "i lost" creepid
 (relative time) "got" value "mana" //ez csak akkor, amikor meghal egy creep, nem?
 (relative time) "i won" yeeeeea
 (relative time) "not enough mana"
 (relative time) "the fog came for us"

program.log:
 routeid "created" idxcoord, prev
 routeid "next route added" routeid
 "program started" configfilename
 "Game loaded" savefilename
 "Game initialized" wavefilename
 "Game started"
 "Game saved" savefilename
 "Tile selected" idxcoord
 (debug-value) "set to" value

7.2 Összes részletes use-case

Use-case neve	new game
Rövid leírás	Új játék kezdése a menüből.
Aktorok	Game
Forgatókönyv	A felhasználó a New Game parancs megadása után opcionálisan megadhatja, hogy előre megírt parancs file-t akar-e használni. Amennyiben igen, ki kell választania, a lehetőségek közül. Ezután elindul a játék.

Use-case neve	load game
Rövid leírás	Korábbi játék betöltése
Aktorok	Game
Forgatókönyv	Load Game parancs, a felhasználó választ egy mentést a lehetséges mentett játékok közül. (slot-ok) A parancs megadása után automatikusan betölt az elmentett játék.

Use-case neve	add tower
Rövid leírás	Torony lerakása a mező egy pontjára.
Aktorok	Game, Tower, Route-ok
Forgatókönyv	addTower parancs, a felhasználónak meg kell adnia egy megfelelő koordinátát, ahova letelepíthető az új torony. Utána levonódik a megfelelő mennyiségű varázserő Szarumántól. (megj. csak elegendő varázserő esetén használható a parancs)

Use-case neve	add Obstacle
Rövid leírás	Akadály lerakása még üres útra.
Aktorok	Game, Route
Forgatókönyv	addObstacle parancs, a felhasználónak meg kell adnia egy megfelelő koordinátát, ahova letelepíthető az új akadály. Utána levonódik a megfelelő mennyiségű varázserő Szarumántól. (megj. csak elegendő varázserő esetén használható a parancs)

Use-case neve	select tile
Rövid leírás	A pálya egy pontjának kiválasztása.
Aktorok	Game
Forgatókönyv	<p>selectTile parancs, a felhasználó megadja a kívánt koordinátákat, három lehetséges kimenet van:</p> <ul style="list-style-type: none"> - üres a tile - akadály van a ponton - torony van a ponton <p>Ha torony, vagy akadály van a ponton, akkor kiírja a lehetséges köveket fejlesztéshez. (segédleti kimenet)</p>

Use-case neve	upgrade
Rövid leírás	A kiválasztott védelmi egység fejlesztése.
Aktorok	Tower vagy Route, Game (varázserő)
Forgatókönyv	a selectTile által kiválasztott védelmi egységet fejlesztheti a felhasználó. A showStone parancs kiírja a választott kő tulajdonságait, a buyStone parancs módosítja az egység értékeit, valamint leveszi a megfelelő mennyiségű varázserőt.

Use-case neve	quit
Rövid leírás	Kilép a megkezdett játékból
Aktorok	Game
Forgatókönyv	a Quit parancs használata után, a program megkérdezi a felhasználót, hogy szeretné-e menteni a játékot.

Use-case neve	debug
Rövid leírás	A debug parancsokkal tudjuk irányítani a játék reakciót.
Aktorok	Game, Projectile, Route,
Forgatókönyv	<p>A debugFog, debugSpecialHit és debugSelectRoute parancsok esetén három opción közül lehet választani. Az első kettőnél: igen, nem és játékbeli.</p> <p>a debugSelectRoute esetén: jobb, bal és játékbeli.</p> <p>A debugGodMode nem rendelkezik opcionálisan, minden össze növeli a rendelkezésre álló varázskövet és a varázserőt.</p> <p>A help parancs kilistázza a lehetséges parancsokat.</p>

7.3 Tesztelési terv

Teszt-eset neve	Új játék kezdése
Rövid leírás	A new game parancssal új játékot tudunk kezdeni, ekkor kezdetben se akadályunk se tornyunk nincs a pályán.
Teszt célja	Ellenőrizni, hogy tényleg megfelelően inicializálódtak-e az egyes objektumok a játékkezdéshez.

Teszt-eset neve	Játék lementése
Rövid leírás	A quit parancs a mentés opcióját választjuk.
Teszt célja	Fontos use-case, hogy a játékot megszakíthatjuk, és adott állapotban lementhetjük.

Teszt-eset neve	Játék betöltése
Rövid leírás	New game helyett load game parancsot fogunk megadni, s ekkor előre megadott állásból kell folytatódnia a játéknak.
Teszt célja	Fontos use-case, hogy korábban lementett játékot lementhetünk.

Teszt-eset neve	Kattintás a pálya egy mezéjére
Rövid leírás	A felhasználó tetszőleges koordináta megadásával megkapja, hogy a pályának melyik IndexCoordinátáját adta meg. (melyik Tile-t választotta ki) A funkció a selectTile parancssal próbálható ki, és kimenetnél megkapjuk, hogy mezőre/toronyra/útra esett a választásunk.
Teszt célja	A játékban fontos kommunikációs forma lesz az egérkattintás. Így valósítjuk meg a torony és akadályfejlesztéseket, illetve az építésnél is fontos szerepet kap. Amennyiben a játékos fejleszteni szeretne egy védelmi egységet, úgy rákattint majd az adott helyre, és amennyiben fejleszthető terület jelölt ki, úgy jobb oldalon lista jelenik meg a felhasználható varázskövekkel.

Teszt-eset neve	Sikeres toronyépítés
Rövid leírás	A toronyépítés helyes működésének vizsgálatához először új játékot kell kezdenünk. Ez után az addTower parancs megadásával tudunk tornyot letenni. Ez a parancs bekéri a koordinátákat, ahová le szeretnénk tenni a tornyot. A forgatókönyv szerinti koordináták ebben az esetben mező koordináták lesznek, továbbá a játékosnak kellő mennyiségű varázsereje is biztosítva lesz, vagyis a toronynak fel kell épülnie.
Teszt célja	Ellenőrizni, hogy tényleg letettünk-e egy tornyot, és hogy tényleg az általunk megadott helyre lett-e letéve.

Teszt-eset neve	Toronyépítési kísérlet útra
Rövid leírás	Az addTower parancsnak a pálya egy olyan koordinátáját adjuk meg, ahol út van. Ilyenkor a játék érzékeli a Tile-választást, azonban a tornyot nem teszi le, így a kimeneti file-ban nem szabad plusz torony megjelenését kapnunk.
Teszt célja	Miután rákattintottunk a toronyvásárlás gombra, a torony pozíciójának kijelölése szintén kattintással történik. Fontos leellenőrzní, hogy amennyiben útra kattintunk, úgy a torony ne kerüljön lehelyezésre.

Teszt-eset neve	Toronyépítési kísérlet kevés varázserővel
Rövid leírás	A játékosnak nem lesz elég varázsereje, és ekkor fogja meghívni az addTower parancsot. A parancsnak fel kell ismerni, hogy nincs elég mana, és ilyenkor már a koordinátákat se fogja bekérni.
Teszt célja	Tornyot csak megfelelő mennyiségű varázserővel lehet. Amennyiben a játékosnak nem áll elegendő a rendelkezésére, úgy a toronyépítés gombnak inaktív állapotban kell lennie.

Teszt-eset neve	Sikeres akadályépítés
Rövid leírás	Az akadályépítés helyes működésének vizsgálatához először új játékot kell kezdenünk. Ez után az addObstacle parancs megadásával tudunk akadályt letenni. Ez a parancs bekéri a koordinátákat, ahová le szeretnénk tenni az akadályt. A forgatókönyv szerinti koordináták ebben az esetben út koordináták lesznek, továbbá a játékosnak kellő mennyiségű varázsereje is biztosítva lesz, vagyis az akadálynak létrejön.
Teszt célja	Ellenőrizni, hogy tényleg letettünk-e egy akadályt, és hogy tényleg az általunk megadott helyre lett-e letéve.

Teszt-eset neve	Akadályépítési kísérlet mezőn
Rövid leírás	Az addObstacle parancsnak a pálya egy olyan koordinátáját adjuk meg, ahol mező vagy torony van. Ilyenkor a játék érzékeli a Tile-választást, azonban az akadályt nem teszi le, így a kimeneti file-ban nem szabad plusz akadály megjelenését kapnunk.
Teszt célja	Miután rákattintottunk az akadályvásárlás gombra, az akadály pozíciójának kijelölése szintén kattintással történik. Fontos leellenőrizni, hogy amennyiben nem útra kattintunk, úgy az akadály ne kerüljön lehelyezésre.

Teszt-eset neve	Akadályépítési kísérlet kevés varázserővel
Rövid leírás	A játékosnak nem lesz elég varázsereje, és ekkor fogja meghívni az addObstacle parancsot. A parancsnak fel kell ismerni, hogy nincs elég mana, és ilyenkor már a koordinátákat se fogja bekérni.
Teszt célja	Akadályt csak megfelelő mennyiségű varázserővel lehet. Amennyiben a játékosnak nem áll elegendő a rendelkezésére, úgy az akadályépítés gombnak inaktív állapotban kell lennie.

Teszt-eset neve	Varázskövétek toronyra
Rövid leírás	A játékban korábban már rendelkezésre állnak tornyaink és van kellő varázserőnk is. Ekkor a selectTile parancssal kiválasztunk egy tornyot tartalmazó Tile-t. Ekkor segédleti kimenetként megkapjuk a felhasználható varázskövek listáját. Ebből a listából a selectStone parancssal választhatunk ki egy követ, illetve kaphatunk róla részletesebb leírást. Vétel a buyStone parancs beadásával történik. Amennyiben mégsem szeretnénk a kiválasztott követ megvenni, úgy a cancel parancssal visszaléphetünk a listázáshoz.
Teszt célja	A toronyfejlesztés folyamatának tesztelése.

Teszt-eset neve	Toronyfejlesztési kísérlet kevés varázserővel.
Rövid leírás	A játékosnak lesz már tornya, és a store-ban is lesz még kő, azonban varázsereje nem lesz elegendő. Így amikor a fent megadott módon elkezdené fejleszteni egy tornyát, a buyStone parancsnak inaktivnak kell lennie.
Teszt célja	Ha nincs elég varázserőnk a nem fejleszthetünk.

Teszt-eset neve	Varázskővétel akadályra
Rövid leírás	A játékban korábban már rendelkezésre állnak akadályok és van kellő varázserőnk is. Ekkor a selectTile parancssal kiválasztunk egy akadályt tartalmazó Tile-t. Ekkor segédleti kimenetként megkapjuk a felhasználható varázskövek listáját. Ebből a listából a showStone parancssal választhatunk ki egy követ, illetve kaphatunk róla részletesebb leírást. Vétel a buyStone parancs beadásával történik. Amennyiben mégsem szeretnénk a kiválasztott követ megvenni, úgy a ugyanazt a Tile-t megadva berekesztjük a vásárlást.
Teszt célja	Az akadályfejlesztés folyamatának tesztelése.

Teszt-eset neve	Akadályfejlesztés kevés varázserővel
Rövid leírás	A játékosnak lesz már akadálya, és a store-ban is lesz még kő, azonban varázsereje nem lesz elegendő. Így amikor a fent megadott módon elkezdené fejleszteni egy akadályt, a buyStone parancsnak inaktinak kell lennie.
Teszt célja	Ha nincs elég varázserőnk a nem fejleszthetünk.

Teszt-eset neve	Varázskő elfogyása
Rövid leírás	A játékban már megfelelő számú védelmi egységünk és elegendő varázserőnk lesz. Ekkor egymás után többször fogunk fejleszteni egy megadott létesítményre a fent leírt parancsok segítségével (selectTile , selectStone , buyStone), egészen addig, amíg el nem fog egy adott varázskő. A megfelelő számú lépés vásárlás után a következő listázásra már nem szabad megjelennie.
Teszt célja	A varázskő készlet véges, így fontos, hogy amennyiben a játékos már felhasználta a rendelkezésre álló kvótáját, úgy többé ne tudjon venni belőle.

Teszt-eset neve	Lövés teszt
Rövid leírás	Mivel a célpontkijelölés automatikus, így a játékosnak a lövés teszthez pusztán egy tornyot kell leraknia.
Teszt célja	Torony lövése megsebzi az ellenfelet. Fontos, hogy a célpont hatósugarán belül helyezkedjen el.

Teszt-eset neve	Creep elpusztulása
Rövid leírás	Szintén autómatikus, csak megfelelő számú toronynak kell lennie.
Teszt célja	A creep elpusztulásával megnő a varázserőnk, illetve a creepnek el kell tűnnie a creep listából.

Teszt-eset neve	Creep elhagyja a tüzelő torony hatósugarát
Rövid leírás	Szintén autómatikus, csak megfelelő számú toronynak kell lennie, és a creepnek elegendő életerővel kell rendelkeznie.
Teszt célja	A toronyok csak addig lőhetik a creepet, amíg a hatósugarukon belül vannak. Ha továbbment, új célpont kell.

Teszt-eset neve	Fejlesztett torony nagyobb hatótávolságának ellenőrzése
Rövid leírás	A toronynak olyan messzebbre is kell tudnia lőni, mint fejlesztés előtt.
Teszt célja	Fontos, hogy a fejlesztésnek tényleg legyen hatása.

Teszt-eset neve	Fejlesztett torony nagyobb sebzésének ellenőrzése
Rövid leírás	A toronynak több életerőt kell elvenni adott típusú ellenfélől mint fejlesztés előtt. A kimeneti fájlban látható lesz a sebzés mértéke, ez alapján könnyen ellenőrizhető lesz.
Teszt célja	Fontos, hogy a fejlesztésnek tényleg legyen hatása.

Teszt-eset neve	Fejlesztett torony nagyobb tüzelési gyakoriságának ellenőrzése
Rövid leírás	A toronynak adott időn belül több lövést kell leadnia, mint fejlesztés előtt.
Teszt célja	Fontos, hogy a fejlesztésnek tényleg legyen hatása.

Teszt-eset neve	Creep sebességének ellenőrzése
Rövid leírás	Ellenőrizzük, hogy a creep tényleg a megadott idő alatt halad-e át egy Tile-on.
Teszt célja	Fontos, hogy a creep tényleg a saját sebességevel haladjon.

Teszt-eset neve	Fejlesztett akadály lassításának ellenőrzése
Rövid leírás	Ellenőrizzük, hogy a creep tényleg lassabban jut-e át egy Tile-on.
Teszt célja	Fontos, hogy a fejlesztésnek tényleg legyen hatása.

Teszt-eset neve	Speciális lövedékek sebzésének ellenőrzése
Rövid leírás	A lövedék típusa kézzel is állítható lesz, így ennek beállítás után kettéhasadt ellenfeleket kell kapnunk.
Teszt célja	Néhány lövedék megkettőzi az eltalált ellenfelet, életerejüket felére csökkentve.

Teszt-eset neve	Köd teszt
Rövid leírás	A köd eljövetele szintén kézzel állítható lesz a prototípusban, így le lehet tesztelni, hogy valóban csökkenti-e a hatótávolságot.
Teszt célja	Amennyiben leszáll a köd, a tornyok vesztenek a hatótávolságukból.

Teszt-eset neve	Útelágazás teszt
Rövid leírás	Útelágazásnál az ellenfelek véletlenszerűen választanak irányt, azonban a prototípus determinisztikus működése végett az irányválasztás manuálisan állítható a megfelelő debug parancsal.
Teszt célja	Fontos, hogy az útelágazások ne okozzanak zavart a játék menetében.

Teszt-eset neve	Win teszt
Rövid leírás	Egy olyan parancssorozat eredménye amelyben a felhasználónak sikerül minden ellenfelet elpusztítania.
Teszt célja	A játék helyesen kezelje a játék végének a lehetséges kimeneteit.

Teszt-eset neve	Lose teszt
Rövid leírás	Egy olyan parancssorozat eredménye amelyben sikerül a szövetségeseknek eljutni a Végzet Hegyéhez.
Teszt célja	A játék helyesen kezelje a játék végének a lehetséges kimeneteit.

7.4 Tesztelést támogató segéd- és fordítóprogramok specifikálása

Notepad++, Eclipse, noobcomp

7.5 Módosítások

1. A tornyokra időnként köd ereszkedik, aminek következtében a látás erősen lecsökken. Ez hatással van a lövésre.

Ezt egy új, statikus boolean beiktatásával oldotta meg a csapat, ami a Tower osztály része. Valamint egy új függvénye a setFog(), amit a Game osztály wave() függvénye fog meghívni, ha szükséges.

2. A játékosok által járható útvonalon lehetnek elágazások és becsatlakozások. Az elágazásokon az egyes játékosok véletlenszerűen mennek a különböző irányokba.

Egy új osztályt hoztunk létre, ami a Route-ból örököl. Nem egy, hanem két következő utat tárolhat, valamint felüldefiniálja a Route getNextPosition()-ját. Ezen kívül a Game osztály init() függvényét kell módosítani, hogy értelmezze az elágazásokat mint érvényes tile.

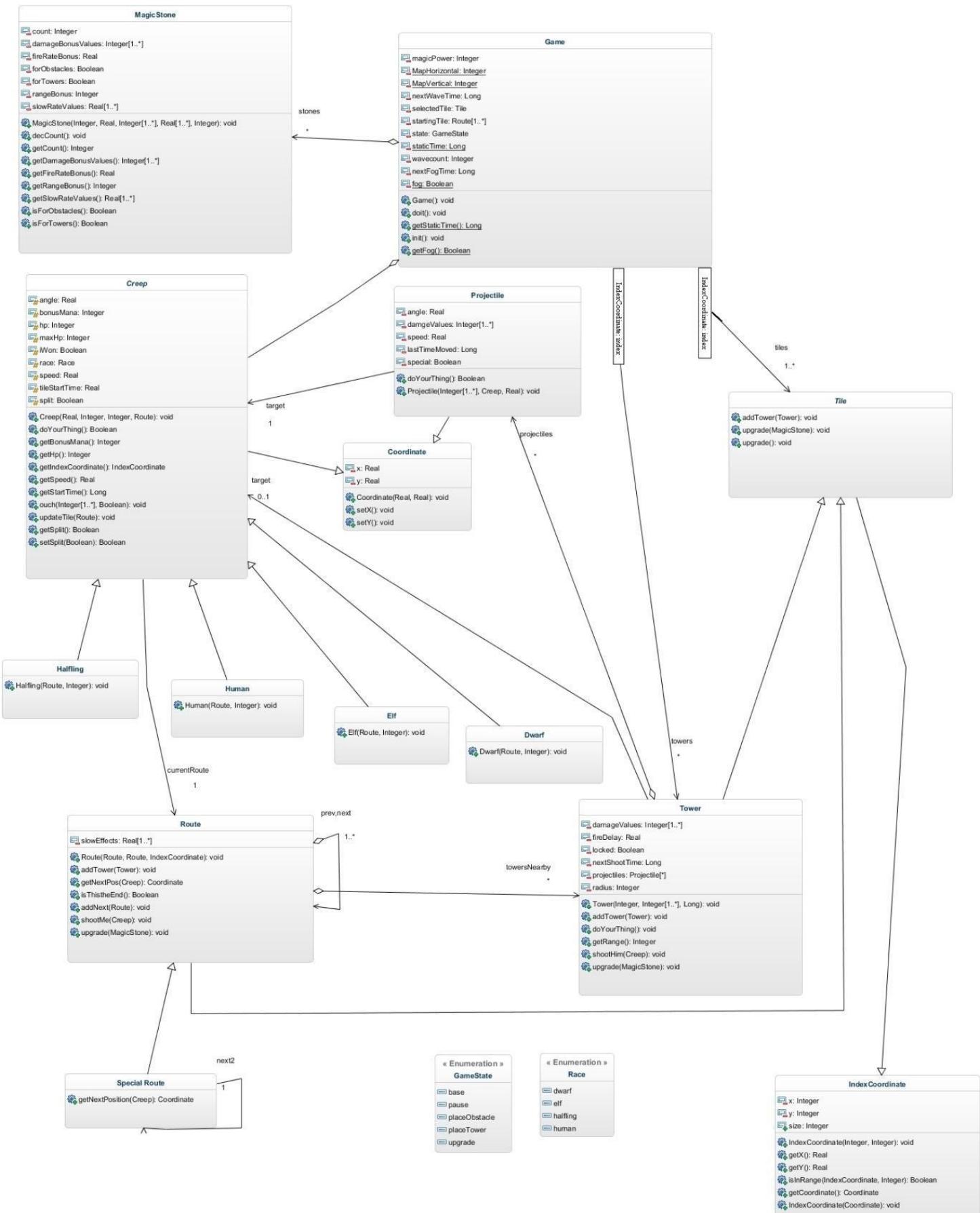
3. A tornyokban elvétve lehetnek olyan lövedékek, amelyek az eltalált játékost kettőbe vágják. A két játékos egymástól függetlenül él tovább, csökkentett életerővel.

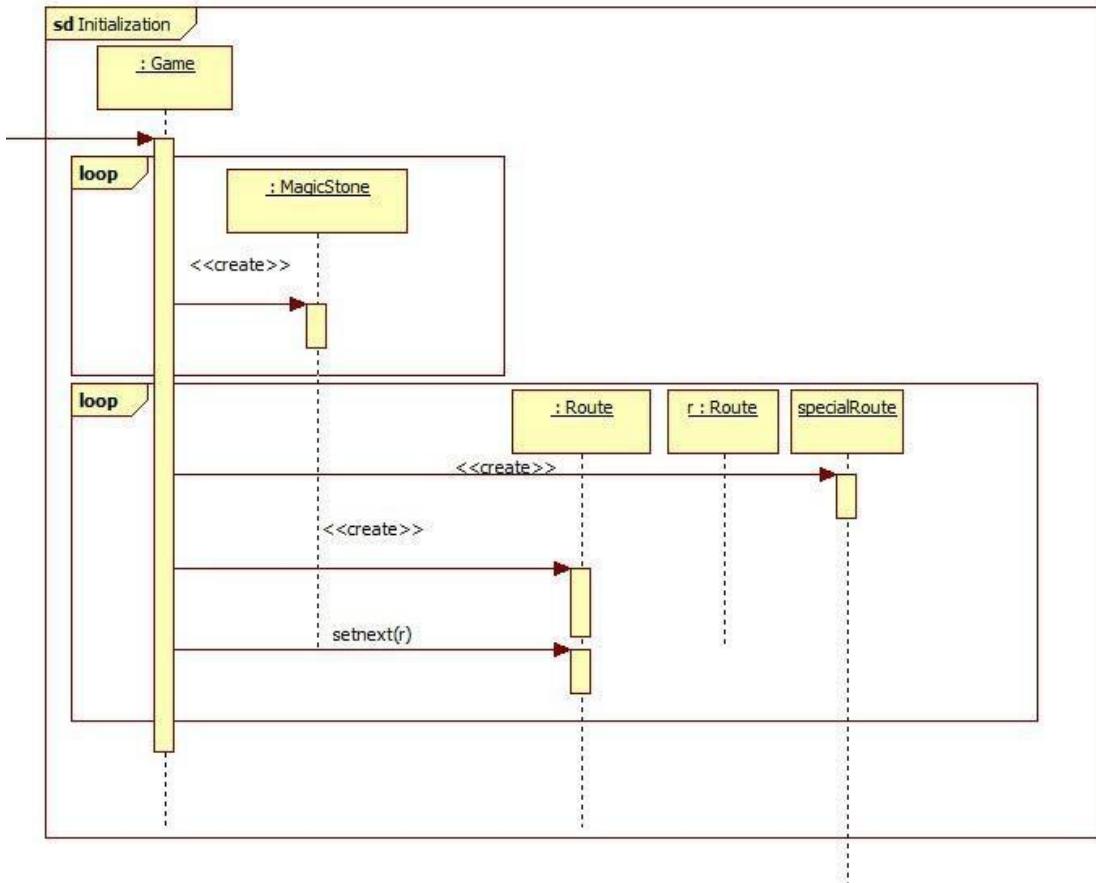
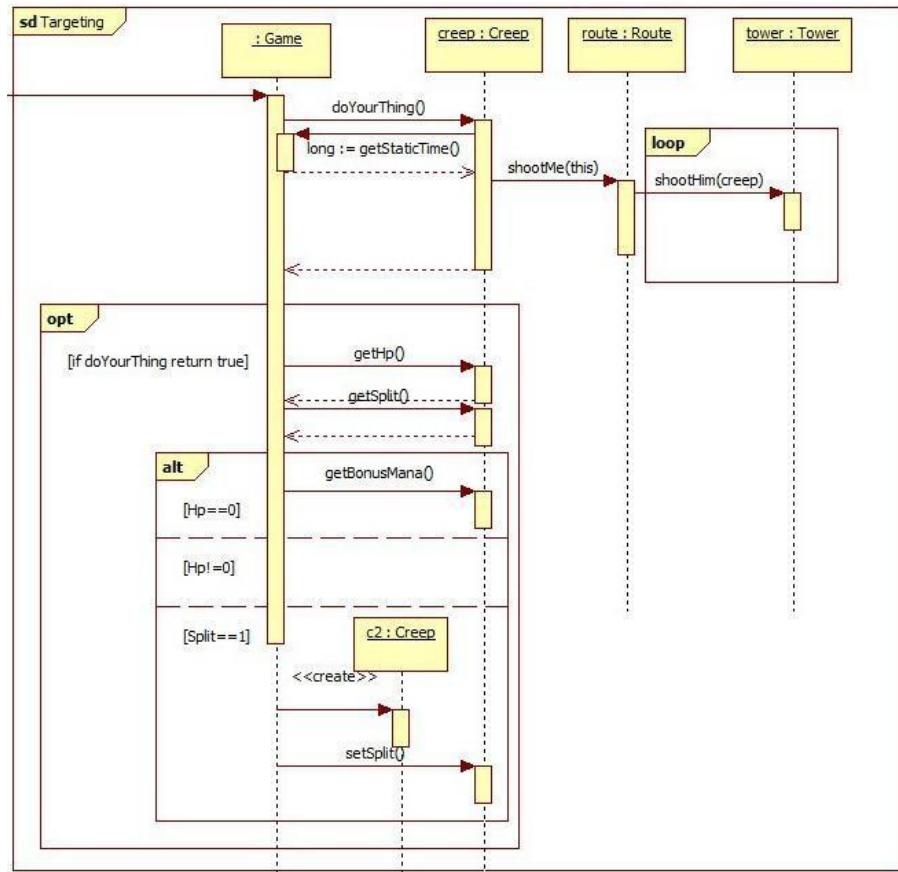
Azt hogy a lövedék rendelkezik-e ezekkel a tulajdonságokkal, a Projectile magának állítja be a konstruktőr ában. Ezek a lövedékek úgy sebeznek, hogy az életerőt felére csökkentik. A creepekhez hozzáadtunk egy új attribútumot. (List split) Ha a Creep-et elérte egy ilyen lövedék, elmentjük a listába az akkori életerőnek a felét. A Game pedig amikor végigiterál az összes Creepen lekérdezi tőlük, hogy van-e benne elem. Ha igaz, akkor létrehozza az új Creep-eket, és törli a listát.

7. Prototípus koncepciója

[NoobTeam]

A változások miatt módosult osztálydiagram és szekvencia diagramok:





7.6 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2014.03.28. 18:00	2,5 óra	Iklódi Németh Molnár Srajner	kupaktanács
2014.03.29. 3:00	4 óra	Srajner	prototípus dokumentációnak prototípusának elkészítése
2014.03.29. 13:30	4,5 óra	Radnai	a játék program írása, kezdetleges grafikus megjelenítéssel, hogy írás közben könnyen tesztelhető legyen. ebből fog majd készülni a prototípus
2014.03.30. 00:30	1,5 óra	Iklódi	Tesztesetek megírása
2014.03.30. 19:15	3 óra	Radnai	Log eljárások tervezése, dokumentálása
2014.03.30. 21:00	1 óra	Molnár	Kimeneti,bemeneti nyelv szerkesztése
2014.03.30. 23:00	1,5 óra	Iklódi	Dokumentum letisztázása
2014.03.31. 10:00	3 óra	Iklódi Molnár Németh Srajner	Dokumentum hiányosságának pótlása, végső módosítások véglegesítése
2014.04.06. 01:00	1 óra	Srajner	utómódosítások gui szó kiszedése, split módosítása, stb.

8. Részletes tervezés

8.1 Osztályok és metódusok tervezése.

8.1.1 Main

- **Felelősség**

A program konzolos felületét kezelő osztály. A prototípusban helyettesíti a végleges be és kimeneti egységeket.

- **Ősosztályok**

- **Interfészek**

- **Attribútumok**

- **- pause: boolean**: A pause állapothoz változó.
- **- CONFIG_FN: String**: A config file címe.
- **- logDate: String**: A logolás időpontja.
- **- sdf: SimpleDateFormat**: a futás pontos dátuma.
- **- g: Game**: A játék maga.
- **- stIn: Scanner**: A konzol bemenethez Scanner. (parancsok)
- **- makIn: Scanner**: A konzol bemenethez Scanner. (paranancsok pontosítása)

- **Metódusok**

- **+ void log(String fn, String s)**: Irányítja a logolást a különböző kimenetekre, .log file-okba.
- **- void pause(boolean p)**: A pause állapot megvalósítása.
- **- void commandSelectTile()**: A névben szereplő parancs megvalósítása.
- **- void commandPause()**: A névben szereplő parancs megvalósítása.
- **- void commandDebugSelectRoute()**: A névben szereplő parancs megvalósítása.
- **- void commandDebugSpecialHit()**: A névben szereplő parancs megvalósítása.
- **- void commandMana()**: A névben szereplő parancs megvalósítása.
- **- void commandFog()**: A névben szereplő parancs megvalósítása.
- **- void commandGod()**: A névben szereplő parancs megvalósítása.
- **- void commandLoad()**: A névben szereplő parancs megvalósítása.
- **- void commandQuit()**: A névben szereplő parancs megvalósítása.
- **- void commandPlay()**: A névben szereplő parancs megvalósítása.
- **- void commandAddObstacle()**: A névben szereplő parancs megvalósítása.
- **- void commandAddTower()**: A névben szereplő parancs megvalósítása.
- **- void commandNewGame()**: A névben szereplő parancs megvalósítása.
- **- void commandShowStone()**: A névben szereplő parancs megvalósítása.
- **- void commandBuyStone()**: A névben szereplő parancs megvalósítása.
- **- void commandHelp()**: A névben szereplő parancs megvalósítása.
- **- void handleCommand(String command, boolean config)**: Meghívja a megfelelő parancs függvényét.
- **- void game()**: A parancsok feldolgozásához használt függvény.
- **- void menu()**: A főmenü váza.
- **- void config()**: A config file beolvasásához használt függvény.
- **- void main(String[] args)**: A program fő metódusa.

8.1.2 Game

- **Felelősség**

Irányítja az egész programot, valamint reprezentálja a felhasználót. Tartalmazza a játékbeli elemeket, mint például az eltelt időt.

- **Ősosztályok**

- **Interfészek**

Serializable interface

- **Attribútumok**

- - **staticTime: long**: Az indítás pontos idejét tárolja.
- - **TimeSave: long**: Mentésnél és betöltésnél használjuk a staticTime szerizálására
- - **state: GameState**: A játék jelenlegi állapota.
- - **selectedTile: Tile**: Tárolja az éppen kiválasztott Tile-t. Innen tudja a program, hogy a játékos mit választott utoljára.
- - **magicPower: int**: Szarumán (a játékos) jelenlegi varázserejének mennyiségét tárolja.
- - **nextWaveTime: long**: A következő ellenség-hullámig tartó idő.
- - **waveCount: int**: A jelenlegi wave sorszámát adja meg.
- - **creeps: ArrayList<Creep>**: Tárolja a játékban levő ellenségeket.
- - **towers: HashMap<IndexCoordinate, Tower>**: Hashmap, ami tárolja a játékos által lerakott tornyokat.
- - **tiles: HashMap<IndexCoordinate, Tile>**: A pálya építőelemeit, a Tile-okat tárolja, szintén HashMap-ben.
- - **stones: ArrayList<MagicStones>**: Az elérhető köveket tárolja.
- - **startingTiles: ArrayList<Route>**: Erre a Tile-ra rakja fel a Creep-eket a game. A wave-ben ehhez ad sorszámot.
- - **pause: boolean**: Megmondja, hogy a játék épp áll-e.
- - **pausetime: long**: Tárolja az időpontot, amikor a program belépett pause állapotba.
- - **waveFn: String**: A megnyitásra kerülő wave file nevét tárolja.
- - **waveString: String**: Ebbe a változóba töltjük bele a wavefile-t.
- + **waves: Scanner**: A file-ból beolvasandó ellenség hullámok beolvasója.
- + **MapVertical: int**: Pálya vertikális hossza.
- + **MapHorizontal: int**: Pálya horizontális hossza.

- **Metódusok**

- **Game()**: Az osztály konstruktora, inicializálja a tárolt változókat.
- + **long getStaticTime()**: Visszatér a kezdési idővel.
- + **void init(String waveFn, String makroFn)**: Inicializálja a játékot. Meghívja a routelnit() és stonelnit() függvényeket, beállítja a kezdő időt, a következő ellenség-hullámig tartó időt, valamint beolvassa a wave-eket a file-ból.
- - **void routelnit()**: Inicializálja az utakat, azaz létrehozza a megfelelő kezdő utakat a pályán belül, és meghívja rájuk a makeRoute()-ot.
- - **void stonelnit()**: Inicializálja a köveket.
- - **Route makeRoute(IndexCoordinate[] idxw)**: Összefűzi az út részeit, hogy a teljes úthálózatot képezzék. A kapott tömb első eleme a gyökér, a többi elemet felfűzi, hogy utat képezzenek, majd visszatér a gyökérelemmel.
- + **void newCreep(int race, int lvl, Route spawnPoint)**: A kapott adatokból létrehoz egy új ellenséget, valamint hozzáfűzi a Hashmap-hez.

- **+ void newObstacle(IndexCoordinate idx):** Letesz egy új akadályt. Amennyiben nem megfelelő helyet kap, már más akadály által foglalt, vagy nincs elegendő varázserő, akkor nem teszi le. Ha minden kritériumnak megfelel, akkor levonja a megfelelő mennyiségi varázserőt, és meghívja a Tile upgrade() függvényét.
- **+ void newTower(IndexCoordinate idx):** Letesz egy új tornyot az indexkoordinátára, ha még nincs rajta más, és megfelelő varázserővel rendelkezik Szarumán. Ezen kívül levonja a megfelelő mennyiségi varázserőt, és regisztrálja az új tornyot a hashmap-be. Végül meghívja a registerTowerstoRoutes() metódust.
- **- void registerTowerstoRoutes(IndexCoordinate idx):** Amikor letesz a felhasználó egy tornyot, a környékén levő utaknak meg kell adni, hogy a torony területén vannak, így tudja a Creep, hogy kinek kell lónie őt az adott úton. Azaz a torony végigmegy a hatókörébe eső Tile-okon, és beregisztrálja magát.
- **+ void doit():** A program főciklusa. Meghívja a generateWaves() függvényt, Meghívja a Creep-ek, és a tornyok doYourThing() metódusát, ezen keresztül kommunikálnak egymással a belső aktorok. Ha egy Creep-nek nem üres a special lövedékkor keletkezett listája, létrehozza az új Creepeket és törli a listát.
- **- void generateWaves():** A beolvasott wave file szerint generálja a megfelelő ellenséghullámokat. A doit() hívja meg, ha a waves attribútumban a következő szám nem -1, akkor új Creep lép be a játékba. Beolvassa az adatait, majd a kapott adatokra meghívja a newCreep() függvényt.
- **+ void buyStone(int id):** Ha a selectedTile attribútum érvényes helyre mutat, akkor elhasználja a kapott követ a választott Tile-ra.
- **+ void showStone(int id):** Az értékként kapott kónek leírja atulajdonságait.
- **+ void pause(boolean p):** A kapott érték alapján szünetelteti a játékot. Ha szünetel, akkor nem szabad tovább számolnia az eltelt időt. Viszont ha újra elindítják a programot, akkor folytatni kell a megfelelő időtől a számlálást.
- **+ void newGame(String waveFn):** Új játék kezdése, csak a kapott wave-et állítja be.
- **+ HashMap<IndexCoordinate, Tile> getTiles():** Visszaadja a tiles változót.
- **+ ArrayList<MagicStone> getStones():** Visszatér a stones attribútummal.
- **+ void setSelectedTile(Indexcoordinate idx):** Beállítja a selectedTile változót a kapott értékre.
- **+ void setMagicPower(int magicPower):** Beállítja a varázserőt a kapott értékre.
- **+ void saveStaticTime():** Statikus változó révén a mentéshez szükséges metódus.
- **+ void loadStaticTime():** Statikus változó révén a mentéshez szükséges metódus.
- **+ void newTowerStone(int cost, String name, int radius, double fireRate, int[] damages):** A kapott értékekkel létrehoz egy új követ a stones attribútumban. (A követ toronyként hozza létre.)
- **+ void newObstacleStone(int cost, String name, double[] slow):** A kapott értékekkel létrehoz egy új követ a stones attribútumban. (A követ akadályként hozza létre.)
- **- void endGame():** A játék végét jelzi, kilép a játékból.

8.1.3 Route

- **Felelősség**

Ez az osztály tartalmazza és kezeli az út és akadály tulajdonságait. Csak ezen az osztályon, és leszármazottain tudnak mozogni az ellenségek.

- **Ősosztályok**

IndexCoordinate osztály, Tile osztály

- **Interfészek**

- **Attribútumok**

- **+ mid: int**: Logolást elősegítő azonosító.
- **+ cost: int**: Akadály fejlesztés esetén a fejlesztés ára.
- **- id: int**: Az út azonosítója.
- **- slowEffects: double[]**: Az akadály előkészítésére szolgál, tárolja, hogy melyik ellenséget mennyivel lassítja az adott útszakasz.
- **- next: Route**: A következő útszakaszt tárolja.
- **- prev: Route**: Az előző útszakaszt tárolja.
- **- obstacle: boolean**: Az út állapot jelzője, akadály vagy sem.
- **- towersNearby: ArrayList<Tower>**: Azokat a tornyokat tárolja, melyeknek az út a ható távolságában van.

- **Metódusok**

- **+ Route(Route prev, IndexCoordinate idx)**: A route osztály konstruktora, amely beállítja az alapvető attributumokat.
- **+ boolean isObstacle()**: Igaz, ha az út akadály lett.
- **+ int getId()**: Az id lekérdezésére szolgál.
- **+ void upgrade(MagicStone ms)**: Az ms varázskő lassítási rátájának megfelelően módosítja a slowEffects értékeit.
- **+ void upgrade()**: Az utat akadályá fejleszti, valamint beállítja a slowEffects minden értékét egy alapértelmezettben nagyobb értékre.
- **+ boolean isThisTheEnd()**: Igazzal tér vissza, ha itt található Orodruin, a Végzet hegye. Azaz ha eléri ezt a pontot egy ellenséges egység, vége a játéknak.
- **+ Coordinate getNextPos(Creep c)**: Visszatér a következő út koordinátájával.
- **+ void addTower(Tower t)**: Amennyiben a paraméterben kapott torony még nincs benne a towersNearby listában, úgy hozzáadja. (Override)
- **+ void shootMe(Creep it)**: minden, a towersNearby listában szereplő toronynak meghívja a shootHim metódusát a kapott paraméterrel.
- **+ void setNext(Route next)**: Beállítja a következő útszakaszt.

8.1.4 SpecialRoute

- **Felelősség**

Az útelágazások osztálya. Több következő pozíciót is tud tárolni, az őssével szemben.

- **Ősosztályok**

IndexCoordinate osztály, Tile osztály, Route osztály

- **Interfészek**

- **Attribútumok**

- **- nexts: ArrayList<Route>**: Tárolja a többi lehetséges utat.
- **- debugLockNext: int**: Tesztelési célokra.

- **Metódusok**

- **SpecialRoute(Route prev, IndexCoordinate idx)**: Az osztály konstruktora, inicializálja a példányt.
- **setNext(Route next)**: Beállítja a lehetséges következő utakat.
- **+ Coordinate getNextPosition(Creep)**: Felülírja a Route getNextPosition()

függvényét, több út közül választ egyet.

- **+ int getSize():** Visszatér a lista méretével.
- **+ int getDebugLockNext():** Tesztelési célokra.
- **+ int setDebugLockNext(int debugLockNext):** Tesztelési célokra.

8.1.5 Tower

- **Felelősség**

A tornyokért felelős osztály, tartalmazza a tornyok főbb tulajdonságait valamint felelős azok funkcióinak működéséért.

- **Ősosztályok**

IndexCoordinate osztály, Tile osztály

- **Interfészek**

- **Attribútumok**

- **+ debugFog: int:** Kód teszteléséhez.
- **- fog: boolean:** A kód állapot attribútuma.
- **+ cost: int:** A torony ára.
- **+ mid: int:** id-k meghatározásához szükséges.
- **- id: int:** Adott torony azonosítója, ennek a segítségével logolunk.
- **- radius: int:** Jelzi, hogy hány Tile távolságra tud lőni.
- **- damageValues: int[]:** 4 elemű int tömb, a 4 különböző ellenségnak mekkora sebzést okoz a torony.
- **- fireDelay: double:** A lövések késleltetését tárolja.
- **- locked: boolean:** a torony célpontjának meglétét jelzi, ha van célpont, akkor igaz, ha nincs akkor hamis.
- **- nextShotTime:long:** A következő lövés relatív időpontját tárolja, azaz ekkor lőhet.
- **- target: Creep:** Az aktuális célpontot tárolja.
- **- projectiles: ArrayList<Projectile> :** A torony által létrehozott, és még nem megszüntetett lövedékek listája.

- **Metódusok**

- **+ Tower(int radius, double firedelay, IndexCoordinate idx):** Az osztály konstruktora, amely létrehoz egy tornyot a megadott paramétereknek megfelelően az ősosztálytól lekért koordinátáakra.
- **+ void shootHim(Creep it):** Összehasonlítja a jelenlegi célpontot a kapott értékkel. Ha a kapott érték kedvezőbb helyen van, mint a korábban tárolt, és nincs lock állapotban a torony, akkor lecseréli az új elemre a target-et. Kód esetén megnézi, hogy a csökkentett hatótávba belefér-e az új ellenség.
- **+ void: doYourThing():** Ha letelt a következő lövésig tartó idő, és a célpont még hatótávolságon belül van, akkor létrehoz egy Projectile-t, ami a célpontot kapja meg értékül, valamint visszaállítja alapállapotba a következő lövésig tartó időt. Ha a célpont már nincs hatótávon belül, annak életéből eldönti, hogy megsemmisült, vagy továbbállt, és ezt lejegyzi a logba. Valamint gondoskodik az általa létrehozott Projectile-ok megsemmisítéséről, ha már célbataláltak.
- **+ void : upgrade(MagicStone ko):** A kapott varázskő megfelelő értékeivel növeli a damageValues értékeit, csökkenti a kő lövési késleltetésével arányosan a torony késleltetését, valamint növeli a torony hatótávolságát a kő ehhez tartozó attribútumával.
- **+ int : getRange():** Visszaadja a torony hatótávolságát.
- **+ int getId():** Visszaadja a torony id-jét, azaz azonosítóját.

8.1.6 Creep

- **Felelősség**

Az ellenséges egységek absztrakt ōsosztálya. Ebben vannak definiálva az alaptulajdonságok és a közös tagfüggvények.

- **Ósosztályok**

Coordinate osztály

- **Interfészek**

- **Attribútumok**

- **+ mid: int**: Logolást elősegítő azonosító.
- **# race: Race**: Szám, mely megadja az egység faját.
- **- id: int**: A Creep azonosítója, szintén a logoláshoz.
- **- speed: double**: Az egység mozgásí sebességét adja meg.
- **- hp: int**: Az egység életterejét adja meg.
- **- maxHp: int**: A kezdeti élet mennyisége amivel "születik".
- **- bonusMana: int**: Az ellenséges egységért járó varázserő mennyisége.
- **- tileStartTime: long**: A Tile-ra való belépés ideje.
- **- level: int**: Az ellenséges egység fejlettségi szintje.
- **- currentRoute: Route**: Az aktuális helye az egységnek.
- **- iWon: boolean**: Játék végét jelző logikai attribútum.
- **- splits: ArrayList<Integer>**: A kettéválasztó lövedék módosításhoz szükséges attribútum.

- **Metódusok**

- **+ Creep(double speed,int maxHp,int bonusMana,int lvl,Route spawnPoint)**: Konstruktor mely létrehozza a Creep egy példányát és beállítja az alapértékeket a megfelelő attribútumoknak.
- **+ boolean doYourThing()**: Meghívja a move() metódust, valamint szól annak a jelenlegi koordinátának, hogy lővésre vár. Ha nyert, vagy elfogyott az élete, igazzal tér vissza.
- **+ void move()**: Elkéri a jelenlegi Route-tól, hogy a következő út merre van, és közeledik hozzá.
- **+ void ouch(int[] damageValues)**: Sebzés esetén csökkenti a hp-t.
- **+ void updateTile(Route t)**: Frissíti az aktuális Tile helyzetét. Amennyiben a jelenlegi koordinátája megegyezik az utolsóval, véget ért a játék, beállítja az IWon attribútumot.
- **+ int getID()**: Visszaadja az egység azonosítóját.
- **+ int getRace()**: Visszaadja az egységhez tartozó faját.
- **+ int getBonusMana()**: Visszaadja, hogy mennyi varázserő jár az egység megöléséért.
- **+ ArrayList<Integer> getSplits()**: Visszatér a getsplits változóval.
- **+ int getHp()**: Visszaadja az egység életterejét.
- **+ void setHp(int hp)**: Beállítja az életerőt a kapott értékre.
- **+ int getNext()**: Visszatér a next változóval
- **+ void setNext(int next)**: Beállítja a next változót.
- **+ int getLevel()**: Visszatér a level változóval.
- **+ double getSpeed()**: Visszaadja, hogy az egység mekkora sebességgel képes mozogni.
- **+ IndexCoordinate getIndexCoordinate()**: Megadja az aktuális helyzet koordinátáit.
- **+ long getStartTime()**: Viszaadja hogy mikor indult el a jelenlegi tile-járól.

- **+ void setStartTime(long l):** Beállítja a tile-ról való indulási időt

8.1.7 Human

- **Felelősség**

Az egyik ellenség típusa, az embereket reprezentáló osztály.

- **Ősosztályok**

Coordinate osztály, Creep osztály

- **Interfészek**
- **Attribútumok**
- **Metódusok**

- **+ Human(Route spawnPoint,int lvl):** Az osztály konstruktora. Megadja az űsének a kapott szintet, és, hogy hol kell létrejönnie. Valamint beállítja a faját Human-re.

8.1.8 Halfling

- **Felelősség**

Az egyik ellenség típusa, a hobbitokat reprezentáló osztály.

- **Ősosztályok**

Coordinate osztály, Creep osztály

- **Interfészek**
- **Attribútumok**
- **Metódusok**

- **+ Halfling(Route spawnPoint,int lvl):** Az osztály konstruktora. Megadja az űsének a kapott szintet, és, hogy hol kell létrejönnie. Valamint beállítja a faját Halfling-ra.

8.1.9 Dwarf

- **Felelősség**

Az egyik ellenség típusa, a törpöket reprezentáló osztály.

- **Ősosztályok**

Coordinate osztály, Creep osztály

- **Interfészek**
- **Attribútumok**
- **Metódusok**

- **+ Dwarf(Route spawnPoint,int lvl):** Az osztály konstruktora. Megadja az űsének a

kapott szintet, és, hogy hol kell létrejönnie. Valamint beállítja a faját Dwarf-ra.

8.1.10 Elf

- **Felelősség**

Az egyik ellenség típusa, a tündéket reprezentáló osztály.

- **Ősosztályok**

Coordinate osztály, Creep osztály

- **Interfészek**

- **Attribútumok**

- **Metódusok**

- **+ Elf(Route spawnPoint,int lvl):** Az osztály konstruktora. Megadja az ōsének a kapott szintet, és, hogy hol kell létrejönnie. Valamint beállítja a faját Elf-re.

8.1.11 Projectile

- **Felelősség**

A tornyok által lött lövedékek osztálya. Két féle típusa van, az ordinary, és a special.

- **Ősosztályok**

Coordinate osztály

- **Interfészek**

- **Attribútumok**

- **- speed: double:** A lövedék sebessége double-ben.
- **- damageValues: int[]:** Tartalmazza, hogy az egyes fajokat mekkora sebzés éri a becsapódáskor.
- **- lastMoveTime: long:** Az előző frame-kori idő, ez biztosítja a konzisztecnit, ha nem ugyan akkora időközönként futna a főciklus.
- **- special: boolean:** Megmondja, hogy special lövedék-e
- **- target: Creep:** A céltól ellenség referenciája.
- **- id: int:** A lövedék id-je.
- **+ mid: int:** logolást elősegítő azonosító.

- **Metódusok**

- **+ Projectile(Creep target, int[] damageValues, double speed, Coordinate c):** Az osztály konstruktora. Inicializálja az attribútumokat, valamint itt dölik el, hogy special lövedék-e (figyelembe veszi a debugspecialhit változót).
- **+ boolean doYourThing():** Kezdetben lekéri az időt, így ki tudja számolni, hoz az előző frame-hez képest pontosan mennyivel kell változtatnia a pozícióját. Valamint a koordinátát úgy változtatja, hogy az a célpont felé közeledjen, tehát minden alkalommal lekéri a célpont értékeit a pontos követéshez. Amennyiben a célpont koordinátája kellően közel van, meghívja annak az ouch() függvényét, a megfelelő damageValues-beli értékkel, és true-val tér vissza. Ha még nem ért oda, akkor

egyszerűen false-al tér vissza.

- **+ void setDebugSpecialHit(int debugSpecialHit)**: Tesztelés céljából beállítja a debugSpecialHit értékét.
- **+ getSpecial()**: Igazzal tér vissza, ha special lövedék.

8.1.12 MagicStone

- **Felelősség**

Ez az osztály tartalmazza a fejlesztéshez használható varázsköveket, melyek segítségével a tornyaink és akadályaink fejleszthetőek bizonyos tulajdonságokkal.

- **Ökosztályok**

- **Interfészek**

- **Attribútumok**

- **- mid: int**: Logolást elősegítő azonosító.
- **- rangeBonus: int**: Megadja, hogy mekkora mértékkel változik a vele fejlesztett torony hatósugara.
- **- fireRateBonus: double**: Megadja, hogy mekkora mértékben változik a vele felszerelt torony tüzelési gyakorisága.
- **- damageValues: int[]**: Megadja, hogy mekkora mértékben változik a vele felszerelt torony sebzése.
- **- slowRateValues: double[]**: Megadja, hogy mekkora mértékben változik a vele felszerelt akadály lassítása.
- **- count: int**: Egy adott kő darabszáma, ez alapján kerül meghatározásra, hogy még mennyi vásárolható belőle.
- **- forTowers: boolean**: Tárolja, hogy a kő torony fejlesztéséhez való-e.
- **- forObstacles: boolean**: Tárolja, hogy a kő akadály fejlesztéséhez való-e.
- **- id: int**: A kő id-je.
- **- cost: int**: A kő ára.
- **- name: String**: Az adott kő neve.

- **Metódusok**

- **+ MagicStone(int r,double f,int[] d,double[] s,int c,boolean t,boolean o)**: Az osztály konstruktora, létrehoz egy követ melynek attribútumait a kapott paramétereknek megfelelően állítja be.
- **+ String toString()**: Serializáláshoz override. Csak a névvel tér vissza.
- **+ int getCost()**: Visszatér a kő árával.
- **+ void decCount()**: Kő vásárlása esetén csökkenti a még vásárolható darabszámot.
- **+ int getCount()**: Visszaadja, hogy még hány követ tudunk vásárolni az adott fajtából.
- **+ int getRangeBonus()**: Visszaadja, hogy mekkora mértékben változtat a kő a hatósugaron.
- **+ double[] getSlowRateValues()**: Visszaadja, hogy mekkora mértékben változtat a kő az akadály lassításán.
- **+ double getFireRateBonus()**: Visszaadja, hogy mekkora mértékben változtat a kő a tüzelési gyakoriságon.
- **+ int[] getDamageBonusValues()**: Visszaadja, hogy mekkora mértékben változtat a kő a sebzés nagyságán.
- **+ boolean getForTowers()**: Megadja, hogy az adott kő toronyhoz felhasználható-e.
- **+ boolean getForObstacles()**: Megadja, hogy az adott kő akadályhoz használható-e.

8.1.13 Coordinate

- **Felelősség**

A játékbeli koordináták tárolására és lekérdezésére használható osztály.

- **Ősosztályok**

- **Interfészek**

- **Attribútumok**

- - **x: double**: A vízszintes tengelyen vett koordinátát tárolja.
- - **y: double**: A függőleges tengelyen vett koordinátát tárolja.

- **Metódusok**

- **+Coordinate(Coordinate pos)**: Az osztály konstruktora, a pos paraméter értékeire állítja az x és y értékét.
- **+Coordinate(double xx,double yy)**: Konstruktur, a kapott értékekre állítja x és y értékét.
- **+ double getX()**: Visszaadja az x értékét.
- **+ double getY()**: Visszaadja az y értékét.
- **+ void setX(double set)**: Beállítja x értékét.
- **+ void setY(double set)**: Beállítja y értékét.

8.1.14 IndexCoordinate

- **Felelősség**

Segítségével képezzük le a teljes pályát egy két dimenziós Tile tömbre. Valamint kiszámítja egy adott koordinátára, hogy pontosan melyik Tile részét képzik.

- **Ősosztályok**

- **Interfészek**

- **Attribútumok**

- - **x: int**: Az indexkoordináta x paramétere.
- - **y: int**: Az indexkoordináta y paramétere.
- **+ size: int**: Az indexkoordináta szélességének értéke, a számításokhoz szükséges.

- **Metódusok**

- **+ int getX()**: Visszaadja az x koordinátát.
- **+ int getY()**: Visszaadja az y koordinátát.
- **+ IndexCoordinate(int xx, int yy)**: Alap konstruktur, beállítja a koordinátákat a kapott értékekre.
- **+ IndexCoordinate(Coordinate c)**: A kapott koordinátából kiszámolja a Tile pontos helyét a pályán.
- **# double getWidth()**: visszaadja az indexkoordináta szélességét.
- **+ double getHeight()**: Visszaadja az indexkoordináta magasságát.
- **+ Coordinate getCoordinate()**: Visszatér az indexkoordináta koordinátájával.
- **+ boolean isInRange(IndexCoordinate idx, int range)**: Igazzal tér vissza, ha a kapott indexkoordináta a megadott range-en belül található.

- **+ boolean equals()**: Az equals override-ja, igazzal tér vissza, ha ugyan oda mutat a két indexkoordináta.
- **+ int hashCode()**: A hashCode override-ja, a hashtálbához szükséges.

8.1.15 Tile

- **Felelősség**

Egy absztrakt osztály, melynek feladata, hogy leszármazottainak előre definiálja a kötelezően megvalósítandó metódusokat.

- **Ősosztályok**

IndexCoordinate osztály

- **Interfészek**
- **Attribútumok**
- **Metódusok**

- **+ void upgrade(MagicStone ko)**: Előre deklarálja ezt a függvényt, mint megvalósítandót a leszármazottaknak
- **+ void addTower(Tower t)**: Előre deklarálja ezt a függvényt, mint megvalósítandót a leszármazottaknak
- **# Tile()**: Meghívja az őse konstruktörét.
- **# Tile(int x, int y)**: Meghívja az őse konstruktörét a kapott értékekkel.

Enumerációk

8.1.16 Race

- **Felelősség**

Az ellenségek lehetséges típusait tárolja.

- **Állapotok**

dwarf
elf
halfling
human

8.1.17 GameState

- **Felelősség**

A játék lehetséges állapotait tárolja.

- **Állapotok**

run
placeTower
placeObstacle
pause
upgrade

8.2 A tesztek részletes tervei, leírásuk a teszt nyelvén

A játék indítása után 3 parancs közül választhatunk: new game, load game, exit. A tesztesetek bementét new game parancs esete kétféle képpen lehet beadni: a „console” szó beírásával, amikor a játék a standard bemenetén várja a további parancsokat, illetve a megfelelő mako kiterjesztésű input file megadásával. A továbbiakban minden esetben megegyezik az console parancs alatt álló parancssorozattal.

8.2.1 Új játék kezdése, lementése és a lementett verzió elindítása

- **Leírás**

A new game parancssal új játékot tudunk kezdeni, ekkor kezdetben se akadályunk se tornyunk nincs a pályán. A wave1.wav file betölt egy wave-et, amely egy creep-et tartalmaz. A teszt célja ellenőrizni, hogy tényleg inicializálódott-e a játék, és a wav file tartalma. Ez után egyből ki is lépünk a játékból a quit parancssal, a mentés opciót választva, majd a load game-mel visszatölthük a játékot.

- **Ellenőrzött funkcionális, várható hibahelyek**

- **Bemenet**

```
new game
1.makro
1
load game
1
quit
1
1
exit
```

vagy

```
new game
console
wave1.wav
quit
1
1
load game
1
quit
1
1
exit
```

- **Elvárt kimenet**

```
program started config.ini
game started
route0 created | 0 0 | ----
route1 created |1 0 | route0
```

```
route0 next route added | route1
route2 created |2 1 | route1
route1 next route added | route2
route3 created |3 1 | route2
route2 next route added | route3
route4 created |4 2 | route3
route3 next route added | route4
route5 created |5 2 | route4
route4 next route added | route5
route6 created |6 3 | route5
route5 next route added | route6
route7 created |7 3 | route6
route6 next route added | route7
route8 created |2 2 | route2
route2 next route added | route8
route9 created |2 3 | route8
route8 next route added | route9
route10 created |3 3 | route9
route9 next route added | route10
route11 created |4 4 | route10
route10 next route added | route11
route12 created |5 4 | route11
route11 next route added | route12
route13 created |5 5 | route12
route12 next route added | route13
route14 created |6 5 | route12
route12 next route added | route14
route15 created |7 5 | route14
route14 next route added | route15
route16 created |8 3 | route7
route7 next route added | route16
route17 created |9 3 | route16
route16 next route added | route17
route18 created |5 6 | route13
route13 next route added | route18
magicstone0 created for towers 1 10 radius +1
magicstone1 created for towers 1 10 radius +3
magicstone2 created for towers 1 10 rate *2
magicstone3 created for towers 1 10 rate *3
magicstone4 created for towers 1 10 damage +10
magicstone5 created for towers 1 10 damage +30
magicstone6 created for obstacles 1 10 slow *2
program initialized waves/wave1.wav
program saved 1.sav
program quitted
program loaded 1.sav
program saved 1.sav
program quitted
program exited
```

8.2.2 Toronyépítés

- **Leírás**

Új játékot kezdünk majd leteszünk egy tornyot egy érvényes mezőre. Ez után le szeretnénk tenni még egy tornyot, azonban először útra, majd utána a másik toronyra próbáljuk ráhelyezni. Az utóbbi két esetben nem kerül új torony lehelyezésre, és a log file-ban láthatjuk is a sikertelen kísérlet nyomait is.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A tower0 created előtti szám időfüggő logkimenet. Ez a szám természetesen tesztenként eltérő lehet. Ezt azonban orvosolja a tesztelést támogató programunk. A továbbiakban pusztán az "időfüggés" szóval utalunk erre a jelenségre. Ez a makró futtatásával valamennyire kiküszöbölnihető, de kisebb eltérések így is lehetnek.

- **Bemenet**

```
new game
2.makro
exit
```

vagy

```
new game
console
wave1.wav
add tower
3 2
add tower
3 3
add tower
3 2
quit
1
2
exit
```

- **Elvárt kimenet**

```
program started config.ini
game started
route0 created | 0 0 | ----
route1 created |1 0 | route0
route0 next route added | route1
route2 created |2 1 | route1
route1 next route added | route2
route3 created |3 1 | route2
route2 next route added | route3
route4 created |4 2 | route3
route3 next route added | route4
route5 created |5 2 | route4
route4 next route added | route5
route6 created |6 3 | route5
route5 next route added | route6
```

```
route7 created |7 3 | route6
route6 next route added | route7
route8 created |2 2 | route2
route2 next route added | route8
route9 created |2 3 | route8
route8 next route added | route9
route10 created |3 3 | route9
route9 next route added | route10
route11 created |4 4 | route10
route10 next route added | route11
route12 created |5 4 | route11
route11 next route added | route12
route13 created |5 5 | route12
route12 next route added | route13
route14 created |6 5 | route12
route12 next route added | route14
route15 created |7 5 | route14
route14 next route added | route15
route16 created |8 3 | route7
route7 next route added | route16
route17 created |9 3 | route16
route16 next route added | route17
route18 created |5 6 | route13
route13 next route added | route18
magicstone0 created for towers 1 10 radius +1
magicstone1 created for towers 1 10 radius +3
magicstone2 created for towers 1 10 rate *2
magicstone3 created for towers 1 10 rate *3
magicstone4 created for towers 1 10 damage +10
magicstone5 created for towers 1 10 damage +30
magicstone6 created for obstacles 1 10 slow *2
program initialized waves/wave1.wav
1 tower0 created | 3 2
i used 2point(s) of magicpower to build tower0
route8: now tower0 is able to reach me
route9: now tower0 is able to reach me
route3: now tower0 is able to reach me
route10: now tower0 is able to reach me
route4: now tower0 is able to reach me
tower cannot be placed at 3 3
tower cannot be placed at 3 2
program saved 2.sav
program quitted
program exited
```

8.2.3 Akadályépítés

- Leírás

Először sikeresen lehelyezünk egy akadályt a 2 2 mezőre. Majd megpróbálunk lehelyezni még egyet erre az útra, ez után pedig egy akadályt mezőre. Az utóbbi két esetben nem helyezhetők le akadályok. A sikertelen lerakási kísérletek a log file-okban is megjelennek.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
időfüggés

- **Bemenet**

new game

3.makro

exit

vagy

new game

console

wave1.wav

add obstacle

2 2

add obstacle

2 2

add obstacle

3 2

quit

1

3

exit

- **Elvárt kimenet**

program started config.ini

game started

route0 created | 0 0 | ----

route1 created |1 0 | route0

route0 next route added | route1

route2 created |2 1 | route1

route1 next route added | route2

route3 created |3 1 | route2

route2 next route added | route3

route4 created |4 2 | route3

route3 next route added | route4

route5 created |5 2 | route4

route4 next route added | route5

route6 created |6 3 | route5

route5 next route added | route6

route7 created |7 3 | route6

route6 next route added | route7

route8 created |2 2 | route2

route2 next route added | route8

route9 created |2 3 | route8

route8 next route added | route9

route10 created |3 3 | route9

route9 next route added | route10

```

route11 created |4 4 | route10
route10 next route added | route11
route12 created |5 4 | route11
route11 next route added | route12
route13 created |5 5 | route12
route12 next route added | route13
route14 created |6 5 | route12
route12 next route added | route14
route15 created |7 5 | route14
route14 next route added | route15
route16 created |8 3 | route7
route7 next route added | route16
route17 created |9 3 | route16
route16 next route added | route17
route18 created |5 6 | route13
route13 next route added | route18
magicstone0 created for towers 1 10 radius +1
magicstone1 created for towers 1 10 radius +3
magicstone2 created for towers 1 10 rate *2
magicstone3 created for towers 1 10 rate *3
magicstone4 created for towers 1 10 damage +10
magicstone5 created for towers 1 10 damage +30
magicstone6 created for obstacles 1 10 slow *2
program initialized waves/wave1.wav
1 route8 obstacle built
i used 1point(s) of magicpower to build obstacle on route8
obstacle cannot be placed at 2 2
obstacle cannot be placed at 3 2
program saved 3.sav
program quitted
program exited

```

8.2.4 Építési kísérlet kevés varázserővel

- **Leírás**

Ha Szarumánnak nem áll rendelkezésére kellő mennyiságú varázserő, akkor nem tud se tornyot se akadályt építeni. Először elfogyasztjuk toronyépítéssel a mana-t, aztán további vásárlásra teszünk kísérletet, azonban ekkor a not enough mana figyelmeztet minket varázserőnk elfogyására.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
időfüggés

- **Bemenet**

new game
4.makro
exit

vagy

```

new game
console
wave1.wav
add tower
3 2
add tower
7 2
add tower
4 3
add tower
6 6
add tower
8 4
add tower
5 3
add obstacle
8 3
quit
1
4
exit

```

- **Elvárt kimenet**

```

program started config.ini
game started
route0 created | 0 0 | ----
route1 created |1 0 | route0
route0 next route added | route1
route2 created |2 1 | route1
route1 next route added | route2
route3 created |3 1 | route2
route2 next route added | route3
route4 created |4 2 | route3
route3 next route added | route4
route5 created |5 2 | route4
route4 next route added | route5
route6 created |6 3 | route5
route5 next route added | route6
route7 created |7 3 | route6
route6 next route added | route7
route8 created |2 2 | route2
route2 next route added | route8
route9 created |2 3 | route8
route8 next route added | route9
route10 created |3 3 | route9
route9 next route added | route10
route11 created |4 4 | route10
route10 next route added | route11
route12 created |5 4 | route11

```

```
route11 next route added | route12
route13 created |5 5 | route12
route12 next route added | route13
route14 created |6 5 | route12
route12 next route added | route14
route15 created |7 5 | route14
route14 next route added | route15
route16 created |8 3 | route7
route7 next route added | route16
route17 created |9 3 | route16
route16 next route added | route17
route18 created |5 6 | route13
route13 next route added | route18
magicstone0 created for towers 1 10 radius +1
magicstone1 created for towers 1 10 radius +3
magicstone2 created for towers 1 10 rate *2
magicstone3 created for towers 1 10 rate *3
magicstone4 created for towers 1 10 damage +10
magicstone5 created for towers 1 10 damage +30
magicstone6 created for obstacles 1 10 slow *2
program initialized waves/wave1.wav
1 tower0 created | 3 2
i used 2point(s) of magicpower to build tower0
route8: now tower0 is able to reach me
route9: now tower0 is able to reach me
route3: now tower0 is able to reach me
route10: now tower0 is able to reach me
route4: now tower0 is able to reach me
1 tower1 created | 7 2
i used 2point(s) of magicpower to build tower1
route6: now tower1 is able to reach me
route7: now tower1 is able to reach me
route16: now tower1 is able to reach me
1 tower2 created | 4 3
i used 2point(s) of magicpower to build tower2
route10: now tower2 is able to reach me
route4: now tower2 is able to reach me
route11: now tower2 is able to reach me
route5: now tower2 is able to reach me
1 tower3 created | 6 6
i used 2point(s) of magicpower to build tower3
route13: now tower3 is able to reach me
route18: now tower3 is able to reach me
route14: now tower3 is able to reach me
route15: now tower3 is able to reach me
1 tower4 created | 8 4
i used 2point(s) of magicpower to build tower4
route7: now tower4 is able to reach me
route16: now tower4 is able to reach me
route17: now tower4 is able to reach me
```

not enough mana
not enough mana
program saved 4.sav
program quitted
program exited

8.2.5 Varázskővétel (+fejlesztett torony nagyobb hatósugának ellenőrzése)

- **Leírás**

A toronyokat lehet fejleszteni. Először leteszünk egy tornyot, majd kiválasztjuk a torony mezejét. Ekkor megkapjuk, hogy milyen lehetőségeink vannak ezt a tile-t fejleszteni. A show stone parancssal kiiratjuk a kő tulajdonságát, a buy stone parancssal pedig megvesszük. Ugyanezt megtehetjük akadályra is. A 0-s kő hatására nő a torony hatósugara, és az újonnan elérhető utak elérhetővé válása megjelenik a log file-ban is.

- **Ellenőrzött funkcionális, várható hibahelyek**
időfüggés

- **Bemenet**

new game
5.makro
exit

vagy

new game
console
wave1.wav
add tower
3 2
select tile
3 2
show stone
0
buy stone
0
add obstacle
6 3
select tile
6 3
show stone
6
buy stone
6
quit
1
5
exit

- **Elvárt kimenet**

```
program started config.ini
game started
route0 created | 0 0 | ----
route1 created |1 0 | route0
route0 next route added | route1
route2 created |2 1 | route1
route1 next route added | route2
route3 created |3 1 | route2
route2 next route added | route3
route4 created |4 2 | route3
route3 next route added | route4
route5 created |5 2 | route4
route4 next route added | route5
route6 created |6 3 | route5
route5 next route added | route6
route7 created |7 3 | route6
route6 next route added | route7
route8 created |2 2 | route2
route2 next route added | route8
route9 created |2 3 | route8
route8 next route added | route9
route10 created |3 3 | route9
route9 next route added | route10
route11 created |4 4 | route10
route10 next route added | route11
route12 created |5 4 | route11
route11 next route added | route12
route13 created |5 5 | route12
route12 next route added | route13
route14 created |6 5 | route12
route12 next route added | route14
route15 created |7 5 | route14
route14 next route added | route15
route16 created |8 3 | route7
route7 next route added | route16
route17 created |9 3 | route16
route16 next route added | route17
route18 created |5 6 | route13
route13 next route added | route18
magicstone0 created for towers 1 10 radius +1
magicstone1 created for towers 1 10 radius +3
magicstone2 created for towers 1 10 rate *2
magicstone3 created for towers 1 10 rate *3
magicstone4 created for towers 1 10 damage +10
magicstone5 created for towers 1 10 damage +30
magicstone6 created for obstacles 1 10 slow *2
program initialized waves/wave1.wav
1 tower0 created | 3 2
i used 2point(s) of magicpower to build tower0
```

```

route8: now tower0 is able to reach me
route9: now tower0 is able to reach me
route3: now tower0 is able to reach me
route10: now tower0 is able to reach me
route4: now tower0 is able to reach me
route2: now tower0 is able to reach me
route11: now tower0 is able to reach me
route5: now tower0 is able to reach me
i used 1point(s) of magicpower to upgrade tower0
1 route6 obstacle built
i used 1point(s) of magicpower to build obstacle on route6
i used 1point(s) of magicpower to upgrade obstacle at6
program saved 5.sav
program quitted
program exited

```

8.2.6 Fejlesztési kísérlet kevés varázserővel

- **Leírás**

Mivel a varázskövek is varázserőbe kerülnek, így előfordulhat, hogy kevés varázserő lévén nem tudjuk fejleszteni a tornyunkat. Ezt az esetet ellenőrzi ez a teszeset. Először toronyépíréssel feléljük a készletünket, majd fejleszteni szeretnénk, azonban not enough mana kimenetet kapunk.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
időfüggés

- **Bemenet**

```

new game
6.makro
exit

```

vagy

```

new game
console
wave1.wav
add tower
3 2
add tower
7 2
add tower
4 3
add tower
6 6
add tower
8 4
add tower
5 3
select tile

```

```
3 2  
buy stone  
0  
quit  
1  
6  
exit
```

- **Elvárt kimenet**

```
program started config.ini  
game started  
route0 created | 0 0 | ----  
route1 created |1 0 | route0  
route0 next route added | route1  
route2 created |2 1 | route1  
route1 next route added | route2  
route3 created |3 1 | route2  
route2 next route added | route3  
route4 created |4 2 | route3  
route3 next route added | route4  
route5 created |5 2 | route4  
route4 next route added | route5  
route6 created |6 3 | route5  
route5 next route added | route6  
route7 created |7 3 | route6  
route6 next route added | route7  
route8 created |2 2 | route2  
route2 next route added | route8  
route9 created |2 3 | route8  
route8 next route added | route9  
route10 created |3 3 | route9  
route9 next route added | route10  
route11 created |4 4 | route10  
route10 next route added | route11  
route12 created |5 4 | route11  
route11 next route added | route12  
route13 created |5 5 | route12  
route12 next route added | route13  
route14 created |6 5 | route12  
route12 next route added | route14  
route15 created |7 5 | route14  
route14 next route added | route15  
route16 created |8 3 | route7  
route7 next route added | route16  
route17 created |9 3 | route16  
route16 next route added | route17  
route18 created |5 6 | route13  
route13 next route added | route18  
magicstone0 created for towers 1 10 radius +1  
magicstone1 created for towers 1 10 radius +3
```

```

magicstone2 created for towers 1 10 rate *2
magicstone3 created for towers 1 10 rate *3
magicstone4 created for towers 1 10 damage +10
magicstone5 created for towers 1 10 damage +30
magicstone6 created fot obstacles 1 10 slow *2
program initialized waves/wave1.wav
1 tower0 created | 3 2
i used 2point(s) of magicpower to build tower0
route8: now tower0 is able to reach me
route9: now tower0 is able to reach me
route3: now tower0 is able to reach me
route10: now tower0 is able to reach me
route4: now tower0 is able to reach me
1 tower1 created | 7 2
i used 2point(s) of magicpower to build tower1
route6: now tower1 is able to reach me
route7: now tower1 is able to reach me
route16: now tower1 is able to reach me
1 tower2 created | 4 3
i used 2point(s) of magicpower to build tower2
route10: now tower2 is able to reach me
route4: now tower2 is able to reach me
route11: now tower2 is able to reach me
route5: now tower2 is able to reach me
1 tower3 created | 6 6
i used 2point(s) of magicpower to build tower3
route13: now tower3 is able to reach me
route18: now tower3 is able to reach me
route14: now tower3 is able to reach me
route15: now tower3 is able to reach me
2 tower4 created | 8 4
i used 2point(s) of magicpower to build tower4
route7: now tower4 is able to reach me
route16: now tower4 is able to reach me
route17: now tower4 is able to reach me
not enough mana
not enough mana
program saved 6.sav
program quitted
program exited

```

8.2.7 Lövés teszt - creep elhagyja a tüzelő torony hatósugarát

- **Leírás**

Leteszünk egy tornyot. Ha egy torony beregisztrál egy creepet célpontként, akkor addig lövi, amíg meg nem hal, vagy pedig, amíg el nem hagyja a torony hatósugarát. Ez a teszteset az utóbbi eseményt teszteli. (az útelágazást determinisztikusra állítjuk)

- **Ellenőrzött funkcionális, várható hibahelyek**
időfüggés

- **Bemenet**

new game

7.makro

exit

vagy

new game

console

wave1.wav

add tower

3 2

debugselectroute

2 1

0

play

15

quit

1

7

exit

- **Elvárt kimenet**

program started config.ini

game started

route0 created | 0 0 | ----

route1 created |1 0 | route0

route0 next route added | route1

route2 created |2 1 | route1

route1 next route added | route2

route3 created |3 1 | route2

route2 next route added | route3

route4 created |4 2 | route3

route3 next route added | route4

route5 created |5 2 | route4

route4 next route added | route5

route6 created |6 3 | route5

route5 next route added | route6

route7 created |7 3 | route6

route6 next route added | route7

route8 created |2 2 | route2

route2 next route added | route8

route9 created |2 3 | route8

route8 next route added | route9

route10 created |3 3 | route9

route9 next route added | route10

route11 created |4 4 | route10

route10 next route added | route11

route12 created |5 4 | route11

route11 next route added | route12

route13 created |5 5 | route12

```
route12 next route added | route13
route14 created | 6 5 | route12
route12 next route added | route14
route15 created | 7 5 | route14
route14 next route added | route15
route16 created | 8 3 | route7
route7 next route added | route16
route17 created | 9 3 | route16
route16 next route added | route17
route18 created | 5 6 | route13
route13 next route added | route18
magicstone0 created for towers 1 10 radius +1
magicstone1 created for towers 1 10 radius +3
magicstone2 created for towers 1 10 rate *2
magicstone3 created for towers 1 10 rate *3
magicstone4 created for towers 1 10 damage +10
magicstone5 created for towers 1 10 damage +30
magicstone6 created for obstacles 1 10 slow *2
game paused
program initialized waves/wave1.wav
game paused
1 tower0 created | 3 2
i used 2point(s) of magicpower to build tower0
route8: now tower0 is able to reach me
route9: now tower0 is able to reach me
route3: now tower0 is able to reach me
route10: now tower0 is able to reach me
route4: now tower0 is able to reach me
game started for 15.0 seconds
0 creep0 created | route0 | 1
10 creep0 moved to | route1
30 creep0 moved to | route2
50 creep0 moved to | route3
50 tower0 new target: creep0
50 projectile0 created | creep0 | 10 | ordinary
50 tower0 projectile created: projectile0
55 projectile1 created | creep0 | 10 | ordinary
55 tower0 projectile created: projectile1
56 creep0 injured | projectile0 | 10
56 projectile0 i've finally arrived
60 projectile2 created | creep0 | 10 | ordinary
60 tower0 projectile created: projectile2
60 creep0 injured | projectile1 | 10
60 projectile1 i've finally arrived
65 creep0 injured | projectile2 | 10
65 projectile2 i've finally arrived
65 projectile3 created | creep0 | 10 | ordinary
65 tower0 projectile created: projectile3
69 creep0 injured | projectile3 | 10
69 projectile3 i've finally arrived
```

```

70 creep0 moved to | route4
70 projectile4 created | creep0 | 10 | ordinary
70 tower0 projectile created: projectile4
75 creep0 injured | projectile4 | 10
75 projectile4 i've finally arrived
75 projectile5 created | creep0 | 10 | ordinary
75 tower0 projectile created: projectile5
80 projectile6 created | creep0 | 10 | ordinary
80 tower0 projectile created: projectile6
80 creep0 injured | projectile5 | 10
80 projectile5 i've finally arrived
85 projectile7 created | creep0 | 10 | ordinary
85 tower0 projectile created: projectile7
86 creep0 injured | projectile6 | 10
86 projectile6 i've finally arrived
90 creep0 moved to | route5
90 tower0 target lost | got away
92 creep0 injured | projectile7 | 10
92 projectile7 i've finally arrived
110 creep0 moved to | route6
130 creep0 moved to | route7
game paused
program saved 7.sav
program quitted
program exited

```

8.2.8 Lövés teszt - creep elpusztulása (+fejlesztett torony nagyobb sebzésének ellenőrzése)

- **Leírás**
Először leteszünk egy tornyot, fejlesztjük,, majd elindítjuk a játékot 10 másodpercre. Itt egyből leteszteljük, hogy a fejlesztett torony meszebbre tud-e lőni. Ugyanis az előző teszesetben ugyanezekkel a paraméterekkel a creep túljutott a tornyon, most azonban meghal. (az útelágazást determinisztikusra állítjuk)
- **Ellenőrzött funkcionalitás, várható hibahelyek**
időfüggés

- **Bemenet**

new game
8.makro
exit

vagy

new game
console
wave1.wav
add tower

```
3 2
debugselectroute
2 1
0
select tile
3 2
buy stone
5
play
10
quit
y
exit
```

- **Elvárt kimenet**

```
program started config.ini
game started
route0 created | 0 0 | ----
route1 created |1 0 | route0
route0 next route added | route1
route2 created |2 1 | route1
route1 next route added | route2
route3 created |3 1 | route2
route2 next route added | route3
route4 created |4 2 | route3
route3 next route added | route4
route5 created |5 2 | route4
route4 next route added | route5
route6 created |6 3 | route5
route5 next route added | route6
route7 created |7 3 | route6
route6 next route added | route7
route8 created |2 2 | route2
route2 next route added | route8
route9 created |2 3 | route8
route8 next route added | route9
route10 created |3 3 | route9
route9 next route added | route10
route11 created |4 4 | route10
route10 next route added | route11
route12 created |5 4 | route11
route11 next route added | route12
route13 created |5 5 | route12
route12 next route added | route13
route14 created |6 5 | route12
route12 next route added | route14
route15 created |7 5 | route14
route14 next route added | route15
route16 created |8 3 | route7
route7 next route added | route16
```

```
route17 created |9 3 | route16
route16 next route added | route17
route18 created |5 6 | route13
route13 next route added | route18
magicstone0 created for towers 1 10 radius +1
magicstone1 created for towers 1 10 radius +3
magicstone2 created for towers 1 10 rate *2
magicstone3 created for towers 1 10 rate *3
magicstone4 created for towers 1 10 damage +10
magicstone5 created for towers 1 10 damage +30
magicstone6 created fot obstacles 1 10 slow *2
game paused
program initialized waves/wave1.wav
game paused
1 tower0 created | 3 2
i used 2point(s) of magicpower to build tower0
route8: now tower0 is able to reach me
route9: now tower0 is able to reach me
route3: now tower0 is able to reach me
route10: now tower0 is able to reach me
route4: now tower0 is able to reach me
i used 1point(s) of magicpower to upgrade tower0
game started for 10.0 seconds
0 creep0 created | route0 | 1
10 creep0 moved to | route1
30 creep0 moved to | route2
50 creep0 moved to | route3
50 tower0 new target: creep0
50 projectile0 created | creep0 | 40 | ordinary
50 tower0 projectile created: projectile0
55 projectile1 created | creep0 | 40 | ordinary
55 tower0 projectile created: projectile1
56 creep0 injured | projectile0 | 40
56 projectile0 i've finally arrived
60 projectile2 created | creep0 | 40 | ordinary
60 tower0 projectile created: projectile2
60 creep0 injured | projectile1 | 40
60 projectile1 i've finally arrived
65 creep0 injured | projectile2 | 40
65 projectile2 i've finally arrived
65 tower0 target lost | died
game paused
program saved 8.sav
program quitted
program exited
```

8.2.9 Fejlesztett torony nagyobb tüzelési gyakoriságának ellenőrzése

- **Leírás**

Fejlesztés után leellenőrizzük, hogy tényleg nagyobb-e a tüzelési gyakorisága a toronynak. Szintén lerakunk a megszokott 3 2 helyre egy tornyot, majd fejlesztjük. A paraméterek ugyanazok, mint a 7-es és 8-as esetben. A 7-es esetben a creep túljut a tornyon, a 8-asban a nagyobb sebzés miatt hal meg, itt pedig a gyakoribb tüzelés következtében.

- **Ellenőrzött funkcionális, várható hibahelyek**
időfüggés

- **Bemenet**

new game
9.makro
exit

vagy

new game
console
wave1.wav
add tower
3 2
debugselectroute
2 1
0
select tile
3 2
buy stone
3
play
10
quit
1
9
exit

- **Elvárt kimenet**

program started config.ini
game started
route0 created | 0 0 | ----
route1 created |1 0 | route0
route0 next route added | route1
route2 created |2 1 | route1
route1 next route added | route2
route3 created |3 1 | route2
route2 next route added | route3
route4 created |4 2 | route3
route3 next route added | route4
route5 created |5 2 | route4
route4 next route added | route5
route6 created |6 3 | route5
route5 next route added | route6

```
route7 created |7 3 | route6
route6 next route added | route7
route8 created |2 2 | route2
route2 next route added | route8
route9 created |2 3 | route8
route8 next route added | route9
route10 created |3 3 | route9
route9 next route added | route10
route11 created |4 4 | route10
route10 next route added | route11
route12 created |5 4 | route11
route11 next route added | route12
route13 created |5 5 | route12
route12 next route added | route13
route14 created |6 5 | route12
route12 next route added | route14
route15 created |7 5 | route14
route14 next route added | route15
route16 created |8 3 | route7
route7 next route added | route16
route17 created |9 3 | route16
route16 next route added | route17
route18 created |5 6 | route13
route13 next route added | route18
magicstone0 created for towers 1 10 radius +1
magicstone1 created for towers 1 10 radius +3
magicstone2 created for towers 1 10 rate *2
magicstone3 created for towers 1 10 rate *3
magicstone4 created for towers 1 10 damage +10
magicstone5 created for towers 1 10 damage +30
magicstone6 created for obstacles 1 10 slow *2
game paused
program initialized waves/wave1.wav
game paused
1 tower0 created | 3 2
i used 2point(s) of magicpower to build tower0
route8: now tower0 is able to reach me
route9: now tower0 is able to reach me
route3: now tower0 is able to reach me
route10: now tower0 is able to reach me
route4: now tower0 is able to reach me
i used 1point(s) of magicpower to upgrade tower0
game started for 10.0 seconds
0 creep0 created | route0 | 1
10 creep0 moved to | route1
30 creep0 moved to | route2
50 creep0 moved to | route3
50 tower0 new target: creep0
50 projectile0 created | creep0 | 10 | ordinary
50 tower0 projectile created: projectile0
```

51 projectile1 created | creep0 | 10 | ordinary
51 tower0 projectile created: projectile1
53 projectile2 created | creep0 | 10 | ordinary
53 tower0 projectile created: projectile2
55 projectile3 created | creep0 | 10 | ordinary
55 tower0 projectile created: projectile3
56 creep0 injured | projectile0 | 10
56 projectile0 i've finally arrived
56 projectile4 created | creep0 | 10 | ordinary
56 tower0 projectile created: projectile4
57 creep0 injured | projectile1 | 10
57 projectile1 i've finally arrived
58 projectile5 created | creep0 | 10 | ordinary
58 tower0 projectile created: projectile5
59 creep0 injured | projectile2 | 10
59 projectile2 i've finally arrived
60 projectile6 created | creep0 | 10 | ordinary
60 tower0 projectile created: projectile6
60 creep0 injured | projectile3 | 10
60 projectile3 i've finally arrived
61 projectile7 created | creep0 | 10 | ordinary
61 tower0 projectile created: projectile7
62 creep0 injured | projectile4 | 10
62 projectile4 i've finally arrived
63 creep0 injured | projectile5 | 10
63 projectile5 i've finally arrived
63 projectile8 created | creep0 | 10 | ordinary
63 tower0 projectile created: projectile8
65 creep0 injured | projectile6 | 10
65 projectile6 i've finally arrived
65 projectile9 created | creep0 | 10 | ordinary
65 tower0 projectile created: projectile9
66 creep0 injured | projectile7 | 10
66 projectile7 i've finally arrived
66 projectile10 created | creep0 | 10 | ordinary
66 tower0 projectile created: projectile10
68 creep0 injured | projectile8 | 10
68 projectile8 i've finally arrived
68 projectile11 created | creep0 | 10 | ordinary
68 tower0 projectile created: projectile11
69 creep0 injured | projectile9 | 10
69 projectile9 i've finally arrived
70 creep0 moved to | route4
70 projectile12 created | creep0 | 10 | ordinary
70 tower0 projectile created: projectile12
71 creep0 injured | projectile10 | 10
71 projectile10 i've finally arrived
71 projectile13 created | creep0 | 10 | ordinary
71 tower0 projectile created: projectile13
73 creep0 injured | projectile11 | 10

```

73 projectile11 i've finally arrived
73 tower0 target lost | died
74 creep0 injured | projectile12 | 10
74 projectile12 i've finally arrived
76 creep0 injured | projectile13 | 10
76 projectile13 i've finally arrived
game paused
program saved 9.sav
program quitted
program exited

```

8.2.10 Fejlesztett akadály lassításának az ellenőrzése

- **Leírás**
Leellenőrzük, hogy a fejlesztett akadály valóban lassít-e. Létrehozunk egy akadályt az 1 0 tile-on, és a log file-ban láthatjuk az időkimenet alapján, hogy a creep a második tile-on sokkal lassabban haladt át, mint az elsőn.
- **Ellenőrzött funkcionális, várható hibahelyek**
időfüggés
- **Bemenet**
new game
10.makro
exit

vagy

```

new game
console
wave1.wav
add obstacle
1 0
select tile
1 0
buy stone
6
play
8
quit
1
10
exit

```

- **Elvárt kimenet**
program started config.ini
game started
route0 created | 0 0 | ----
route1 created |1 0 | route0

```
route0 next route added | route1
route2 created |2 1 | route1
route1 next route added | route2
route3 created |3 1 | route2
route2 next route added | route3
route4 created |4 2 | route3
route3 next route added | route4
route5 created |5 2 | route4
route4 next route added | route5
route6 created |6 3 | route5
route5 next route added | route6
route7 created |7 3 | route6
route6 next route added | route7
route8 created |2 2 | route2
route2 next route added | route8
route9 created |2 3 | route8
route8 next route added | route9
route10 created |3 3 | route9
route9 next route added | route10
route11 created |4 4 | route10
route10 next route added | route11
route12 created |5 4 | route11
route11 next route added | route12
route13 created |5 5 | route12
route12 next route added | route13
route14 created |6 5 | route12
route12 next route added | route14
route15 created |7 5 | route14
route14 next route added | route15
route16 created |8 3 | route7
route7 next route added | route16
route17 created |9 3 | route16
route16 next route added | route17
route18 created |5 6 | route13
route13 next route added | route18
magicstone0 created for towers 1 10 radius +1
magicstone1 created for towers 1 10 radius +3
magicstone2 created for towers 1 10 rate *2
magicstone3 created for towers 1 10 rate *3
magicstone4 created for towers 1 10 damage +10
magicstone5 created for towers 1 10 damage +30
magicstone6 created for obstacles 1 10 slow *2
game paused
program initialized waves/wave1.wav
game paused
1 route1 obstacle built
i used 1point(s) of magicpower to build obstacle on route1
i used 1point(s) of magicpower to upgrade obstacle at1
game started for 8.0 seconds
0 creep0 created | route0 | 1
```

```
10 creep0 moved to | route1
70 creep0 moved to | route2
game paused
program saved 10.sav
program quitted
program exited
```

8.2.11 Speciális lövedékek sebzésének ellenőrzése

- **Leírás**
Ellenőrizzük, hogy a speciális lövedékek valóban kettéhasítják-e a crept.
- **Ellenőrzött funkcionális, várható hibahelyek**
időfüggés

- **Bemenet**

new game
11.makro
exit

vagy

```
new game
console
wave1.wav
add tower
3 2
debugspecialhit
1
debugselectroute
2 1
1
play
7
quit
1
11
exit
```

- **Elvárt kimenet**

program started config.ini
game started
route0 created | 0 0 | ----
route1 created |1 0 | route0
route0 next route added | route1
route2 created |2 1 | route1

```
route1 next route added | route2
route3 created |3 1 | route2
route2 next route added | route3
route4 created |4 2 | route3
route3 next route added | route4
route5 created |5 2 | route4
route4 next route added | route5
route6 created |6 3 | route5
route5 next route added | route6
route7 created |7 3 | route6
route6 next route added | route7
route8 created |2 2 | route2
route2 next route added | route8
route9 created |2 3 | route8
route8 next route added | route9
route10 created |3 3 | route9
route9 next route added | route10
route11 created |4 4 | route10
route10 next route added | route11
route12 created |5 4 | route11
route11 next route added | route12
route13 created |5 5 | route12
route12 next route added | route13
route14 created |6 5 | route12
route12 next route added | route14
route15 created |7 5 | route14
route14 next route added | route15
route16 created |8 3 | route7
route7 next route added | route16
route17 created |9 3 | route16
route16 next route added | route17
route18 created |5 6 | route13
route13 next route added | route18
magicstone0 created for towers 1 10 radius +1
magicstone1 created for towers 1 10 radius +3
magicstone2 created for towers 1 10 rate *2
magicstone3 created for towers 1 10 rate *3
magicstone4 created for towers 1 10 damage +10
magicstone5 created for towers 1 10 damage +30
magicstone6 created fot obstacles 1 10 slow *2
game paused
program initialized waves/wave1.wav
game paused
1 tower0 created | 3 2
i used 2point(s) of magicpower to build tower0
route8: now tower0 is able to reach me
route9: now tower0 is able to reach me
route3: now tower0 is able to reach me
route10: now tower0 is able to reach me
route4: now tower0 is able to reach me
```

```

game started for 7.0 seconds
0 creep0 created | route0 | 1
10 creep0 moved to | route1
30 creep0 moved to | route2
50 creep0 moved to | route8
50 tower0 new target: creep0
50 projectile0 created | creep0 | 10 | special
50 tower0 projectile created: projectile0
55 projectile1 created | creep0 | 10 | special
55 tower0 projectile created: projectile1
56 creep0 injured | projectile0 | 10
56 projectile0 i've finally arrived
56 creep1 created | route8 | 1
60 projectile2 created | creep0 | 10 | special
60 tower0 projectile created: projectile2
60 creep0 injured | projectile1 | 10
60 projectile1 i've finally arrived
60 creep2 created | route8 | 1
64 creep0 injured | projectile2 | 10
64 projectile2 i've finally arrived
65 creep3 created | route8 | 1
65 projectile3 created | creep0 | 10 | special
65 tower0 projectile created: projectile3
69 creep0 injured | projectile3 | 10
69 projectile3 i've finally arrived
69 creep4 created | route8 | 1
game paused
program saved 11.sav
program quitted
program exited

```

8.2.12 Kód teszt

- **Leírás**

Leteszünk egy tornyot a 3 2 helyre. Fejlesztjük egy hatótávolság-növelő kövel, és beállítjuk hogy legyen köd. Ezek után elindítjuk 4 mp-re a játékot. A log file-ban láthatjuk hogy a torny már eléri a route2-t, tehát ha nem lenne köd, amikor a creep eléri a route2-t, a torony lőne. A log kimenet alapján azonban látszik, hogy a torony nem nő.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
időfüggés

- **Bemenet**

new game
12.makro
exit

vagy

```
new game
console
wave1.wav
debugfog
1
add tower
3 2
select tile
3 2
buy stone
0
play
4
quit
1
12
exit
```

- **Elvárt kimenet**

```
program started config.ini
game started
route0 created | 0 0 | ----
route1 created |1 0 | route0
route0 next route added | route1
route2 created |2 1 | route1
route1 next route added | route2
route3 created |3 1 | route2
route2 next route added | route3
route4 created |4 2 | route3
route3 next route added | route4
route5 created |5 2 | route4
route4 next route added | route5
route6 created |6 3 | route5
route5 next route added | route6
route7 created |7 3 | route6
route6 next route added | route7
route8 created |2 2 | route2
route2 next route added | route8
route9 created |2 3 | route8
route8 next route added | route9
route10 created |3 3 | route9
route9 next route added | route10
route11 created |4 4 | route10
route10 next route added | route11
route12 created |5 4 | route11
route11 next route added | route12
route13 created |5 5 | route12
route12 next route added | route13
route14 created |6 5 | route12
route12 next route added | route14
```

```
route15 created |7 5 | route14
route14 next route added | route15
route16 created |8 3 | route7
route7 next route added | route16
route17 created |9 3 | route16
route16 next route added | route17
route18 created |5 6 | route13
route13 next route added | route18
magicstone0 created for towers 1 10 radius +1
magicstone1 created for towers 1 10 radius +3
magicstone2 created for towers 1 10 rate *2
magicstone3 created for towers 1 10 rate *3
magicstone4 created for towers 1 10 damage +10
magicstone5 created for towers 1 10 damage +30
magicstone6 created for obstacles 1 10 slow *2
game paused
program initialized waves/wave1.wav
game paused
1 tower0 created | 3 2
i used 2point(s) of magicpower to build tower0
route8: now tower0 is able to reach me
route9: now tower0 is able to reach me
route3: now tower0 is able to reach me
route10: now tower0 is able to reach me
route4: now tower0 is able to reach me
route2: now tower0 is able to reach me
route11: now tower0 is able to reach me
route5: now tower0 is able to reach me
i used 1point(s) of magicpower to upgrade tower0
game started for 4.0 seconds
0 creep0 created | route0 | 1
10 creep0 moved to | route1
30 creep0 moved to | route2
game paused
program saved 12.sav
program quitted
program exited
```

8.3 A tesztelést támogató programok tervei

A csapat által írt program, a tesztek összehasonlítására a várt értékkel: **noobcomp**

Az előző pontban megadott logkimeneteket előre megadjuk a tesztelő csapatnak a forrásfájlokkal együtt. Miután a tesztelők mauálisan, vagy makrók alapján végigfuttatják a

teszteseteket, a kapott és várt eredményeket ezzel a segédprogrammal hasonlíthatják össze. A **noobcmp** program annyiban tér el a más összehasonlító programoktól, hogy bizonyos esetekben nem veszi figyelembe a kimenetek közöttk különbséget. Ezek az esetek pedig pontosan az időlogok, amelyek nyilvánvalóan minden tesztesetben mások lesznek. (pl: a tower0 created előtt álló időlog a determinisztikus viselkedés szempontjából irreveláns)

Amennyiben eltérést talál a program úgy kiírja az különböző sorok sorszámát.

8.4 Napló

Kezdet	Időtartam	Résznevők	Leírás
2014.04.02 20:45	3 óra	Radnai	prototípus írása, a teszeset elkészítése és ellenőrzése a programmal
2014.04.03. 20:15	1 óra	Molnár	dokumentáció töltése (Creep osztály)
2014.04.03. 16.00	1 óra	Németh	dokumentáció töltése (Tower osztály)
2014.04.04. 10:30	1 óra	Molnár	dokumentáció (MagicStone)
2014.04.04. 11:30	1 óra	Németh	dokumentáció (Route,Coordinate,Tile)
2014.04.05 02:30	4 óra	Srajner	Game osztály véglegesítése, Többi osztály tisztázása
2014.04.05 17:00	7,5 óra	Radnai	prototípus megírása addig, hogy a teszesetek futtathatóak legyenek. minden funkció működőképes, logok és pár appróbb dolog hiányzik csak
2014.04.06. 15:00	5 óra	Iklódi	teszesetek be- és kimeneteinek megírása, dokumentum formázása

10. Prototípus beadása

10.1 Fordítási és futtatási útmutató

10.1.1 Fájllista

Fájl neve	Méret	Keletkezés ideje	Tartalom
config.ini	1 KB	2014.04.21. 10:43	Kezdeti pálya,egység és egyéb indulási feltételek, tulajdonságok megadása, betöltése a játék kezdetéhez.
1.makro	1 KB	2014.04.21. 10:27	8.2.1 teszteset parancsait tartalmazó input fájl
2.makro	1 KB	2014.04.21. 10:27	8.2.2 teszteset parancsait tartalmazó input fájl
3.makro	1 KB	2014.04.21. 10:27	8.2.3 teszteset parancsait tartalmazó input fájl
4.makro	1 KB	2014.04.21. 10:27	8.2.4 teszteset parancsait tartalmazó input fájl
5.makro	1 KB	2014.04.21. 10:27	8.2.5 teszteset parancsait tartalmazó input fájl
6.makro	1 KB	2014.04.21. 10:27	8.2.6 teszteset parancsait tartalmazó input fájl
7.makro	1 KB	2014.04.21. 10:44	8.2.7 teszteset parancsait tartalmazó input fájl
8.makro	1 KB	2014.04.21. 10:47	8.2.8 teszteset parancsait tartalmazó input fájl
9.makro	1 KB	2014.04.21. 10:51	8.2.9 teszteset parancsait tartalmazó input fájl
10.makro	1 KB	2014.04.21. 10:54	8.2.10 teszteset parancsait tartalmazó input fájl
11.makro	1 KB	2014.04.21. 10:57	8.2.11 teszteset parancsait tartalmazó input fájl
12.makro	1 KB	2014.04.21. 11:01	8.2.12 teszteset parancsait tartalmazó input fájl
Coordinate.java	1 KB	2014.04.12.	Coordinate class és függvényei
Creep.java	3 KB	2014.04.12.	Creep class és függvényei
Dwarf.java	1 KB	2014.04.12.	Dwarf class és függvényei
Elf.java	1 KB	2014.04.12.	Elf class és függvényei
Game.java	13 KB	2014.04.12.	Game class és függvényei
GameState.java	1 KB	2014.04.12.	GameState class és függvényei
Halfling.java	1 KB	2014.04.12.	Halflingclass és függvényei
Human.java	1 KB	2014.04.12.	Humanclass és függvényei
IndexCoordinate.java	2 KB	2014.04.12.	IndexCoordinate class és függvényei
MagicStone.java	2 KB	2014.04.12.	MagicStone class és függvényei
Main.java	16 KB	2014.04.12.	Main class és függvényei
Projectile.java	3 KB	2014.04.12.	Projectile class és függvényei
Race.java	1 KB	2014.04.12.	Race class és függvényei
Route.java	4 KB	2014.04.12.	Route class és függvényei

SpecialRoute.java	2 KB	2014.04.12.	Special Route class és függvényei
Tile.java	1 KB	2014.04.12.	Tile class és függvényei
Tower.java	4 KB	2014.04.12.	Tower class és függvényei
2.wav	1 KB	2014.04.06.	Komplettebb wave input
wave1.wav	1 KB	2014.04.02.	Tesztekhez használt alap wave input

10.1.2 Fordítás

A batch file nevű mappában van egy batch file, ami optimális esetben lefordítja a kódot. Azonban a szkeleton bemutatásakor kiderült, hogy az R4J terem gépein nincsen JDK, ezért a következő módon fogjuk a teremben a futtatható kódot előállítani:

Elinítjuk az Eclipse fejlesztőkörnyezetet, és beimportáljuk mind a noobsaruman, mind a noobcmp projekteket. A noobsaruman program maga a játékprogram prototípusa, a noobcmp pedig a kimenetek összehasonlítását segítő segédprogram.

A “Run” gombra kattintva, Run Configurations... menüpontban a megfelelő projektet kiválasztva az Eclipse lefordítja, majd futtatja is a projektet.

10.1.3 Futtatás

A fent leírt módon, az Eclipse “Run” ikonjára kattintva, kiválasztjuk a megfelelő konfigurációt.

A tesztelés menete:

- Először futtassuk a noobsaruman-t. Haladjunk sorban a teszteseteken. Egy-egy teszteset leellenőrzése történhet makro, illetve kézzel beadott parancsfeldolgozás alapján is.
- Ha lefutattunk egy tesztesetet, nyomjunk F5-öt (refresh) a projektnévre kattintva. Ekkor megjelenik a log mappában legalul a legutóbbi teszt kimeneteteket tartalmazó mappa, benne az egyes kimeneti file-okkal.
- Ezeket a file-okat másoljuk át a noobcmp test mappájába. (ha nem üres a mappa, előbb ürítük ki)
- Futtassuk a noobcmp programot. A program megkérdezi, hogy melyik tesztesetet kívánjuk ellenőrizni. (1-12) Adjuk meg a megfelelő számot.
- A cmp program kiírja, hogy “The files are the same”, amennyiben teljes az egyezés. Ha eltérést talál, úgy jelzi a hibás sorokat.
- Alkalmazzuk a fenti lépéseket minden a 12 tesztesetre.

Megjegyzés: Amennyiben az összes kimeneti file-t bemásoljuk a test-nek megfelelő mappába (pl: 2-es test esetén test2-be) akkor a 0-t beadva a noobcmp programnak, egyszerre leellenőrizni az összes tesztesetet.

10.2 Tesztek jegyzőkönyvei

10.2.1 Új játék kezdése, lementése és a lementett verzió elindítása

Teszteleő neve	Radnai
Teszt időpontja	2014.04.21. 14:45

10.2.2 Toronyépítés

Teszteleő neve	Radnai
Teszt időpontja	2014.04.21. 15:07

10.2.3 Akadályépítés

Teszteleő neve	Radnai
Teszt időpontja	2014.04.21. 15:18

10.2.4 Építési kísérlet kevés varázserővel

Teszteleő neve	Radnai
Teszt időpontja	2014.04.21. 15:24

10.2.5 VarázskövéTEL (+fejlesztett torony nagyobb hatósugának ellenőrzése)

Teszteleő neve	Radnai
Teszt időpontja	2014.04.21. 15:27

10.2.6 Fejlesztési kísérlet kevés varázserővel

Teszteleő neve	Radnai
Teszt időpontja	2014.04.21. 15:30

10.2.7 Lövés teszt - creep elhagyja a tüzelő torony hatósugarát

Teszteleő neve	Iklódi
Teszt időpontja	2014.04.21. 15:32

10.2.8 Lövés teszt - creep elpusztulása (+fejlesztett torony nagyobb sebzésének ellenőrzése)

Tesztelő neve	Iklódi
Teszt időpontja	2014.04.21. 15:34

10.2.9 Fejlesztett torony nagyobb tüzelési gyakoriságának ellenőrzése

Tesztelő neve	Iklódi
Teszt időpontja	2014.04.21. 15:36

10.2.10 Fejlesztett akadály lassításának az ellenőrzése

Tesztelő neve	Iklódi
Teszt időpontja	2014.04.21. 15:40

10.2.11 Speciális lövedékek sebzésének ellenőrzése

Tesztelő neve	Iklódi
Teszt időpontja	2014.04.21. 15:42

10.2.12 Kód teszt

Tesztelő neve	Iklódi
Teszt időpontja	2014.04.21. 15:43

10.3 Értékelés

Tag neve	Munka százalékban
Iklódi Eszter	22,37 %
Molnár Bence	17,35 %
Németh Zsolt	16,96 %
Radnai Balázs	22,01 %
Srajner Ferenc	21,31 %

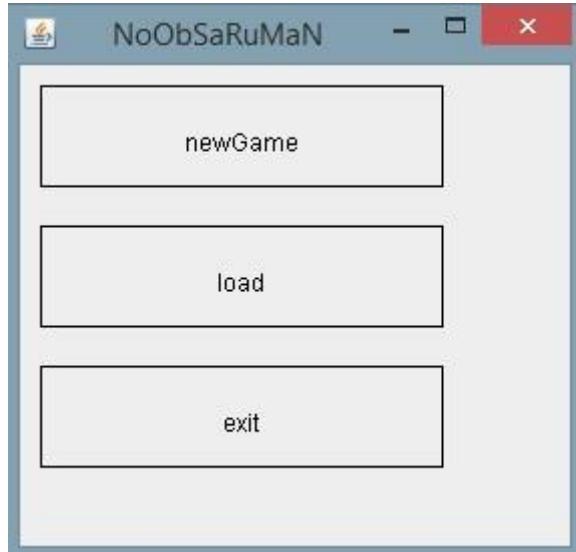
10.4 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2014.04.12.	0,5 óra	Iklódi Radnai Molnár Németh Srajner	program megkezdése
2014.04.19 13:00	4 óra	Radnai	prototípus befejezése, tesztesetek lefuttatása, kijavítása.
2014.04.21. 14:30	1,5 óra	Molnár	Dokumentáció készítése
2014.04.21 14:30	2,5 óra	Iklódi	Dokumentálás, tesztesetek ellenőrzése, véglegesítése
2014.04.21 20:45	2 óra	Németh	Implementáció Eclipse Helios kompatibilissá tétele, kommentezés
2014.04.22 01:00	2,5 óra	Srajner	Compare program implementálása és végeleges ellenőrzés

11. Grafikus felület specifikációja

11.1 A grafikus interfész

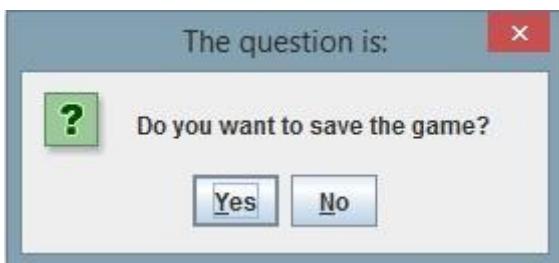
11.1.1 A játék főmenüje



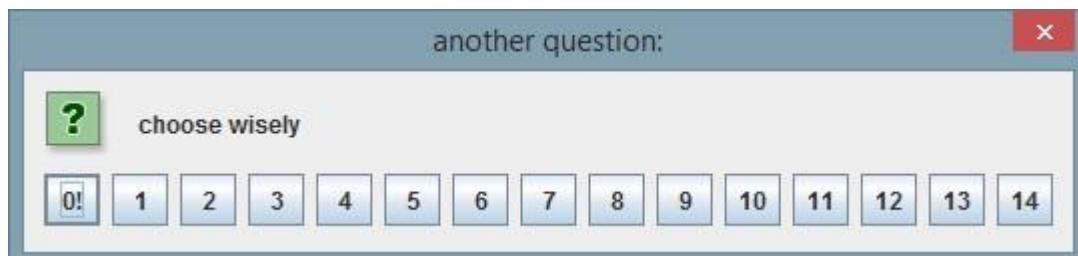
11.1.2 A játék grafikus felülete



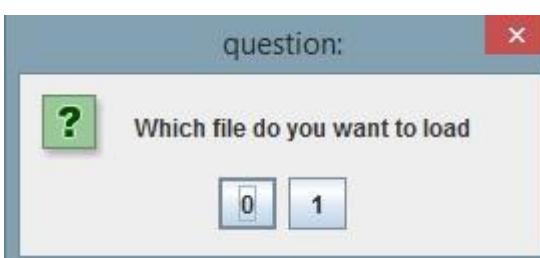
11.1.3 Message box – mentés



11.1.4 Message box – mentés esetén



11.1.5 Message box – load



11.2 A grafikus rendszer architektúrája

11.2.1 A felület működési elve

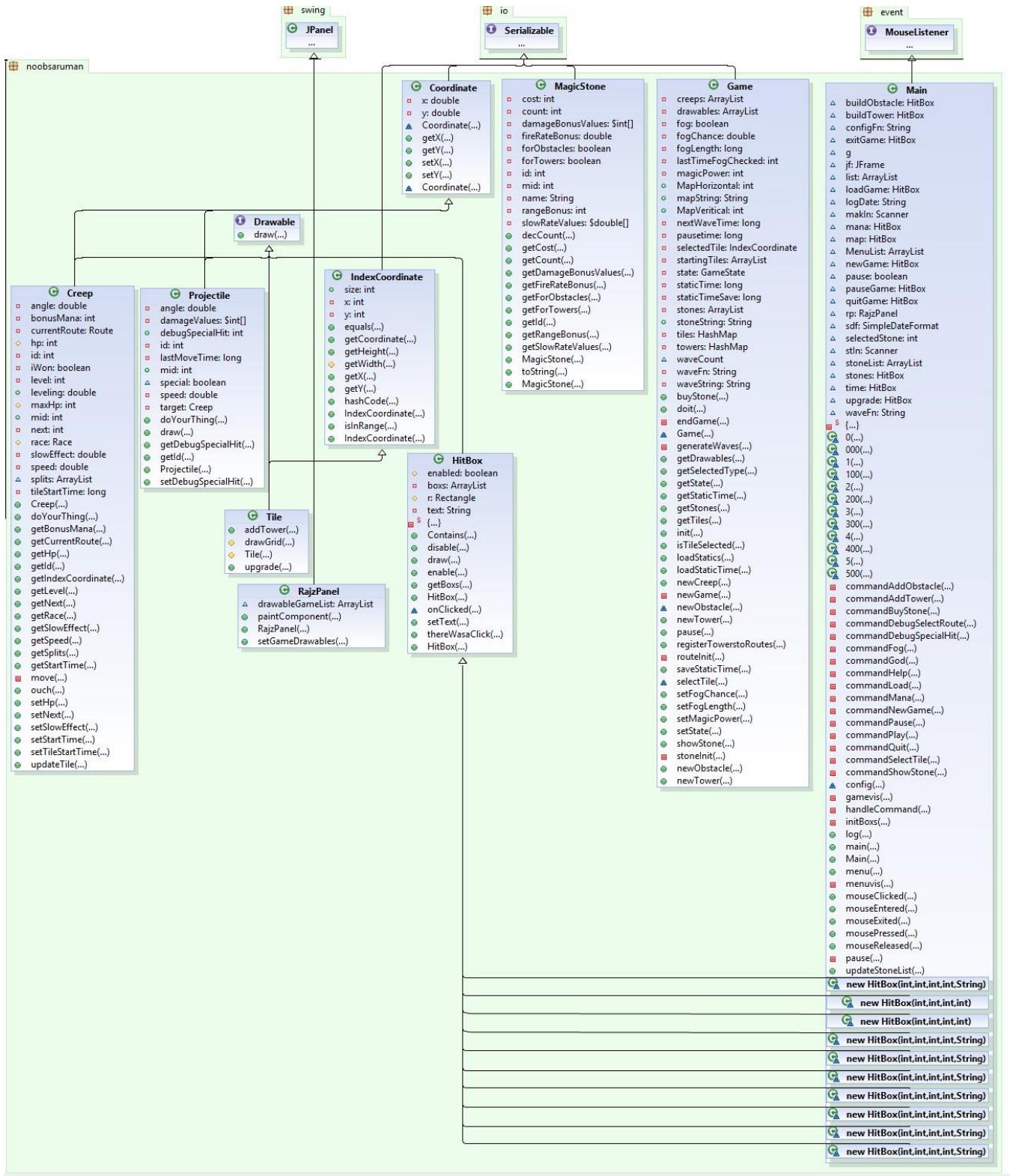
Mivel a modell megjelenítése gyakran, (jó esetben másodpercenként legalább 60-szor történik,) ezért célszerű **pull** alapon kezelní a felületet. Így csökkenthetjük az overhead-et. A grafikus megjelenítés új interface-ekkel, és osztályokkal kerül megvalósításra, ezek megjelenítését a Main osztály kezeli.

Modell: A prototípus során a modell teljes egészében elkészült. A modellt a Game osztály irányítja, és már megfelelően kezelte le a lehetséges eseményeket, és azokat a specifikáció szerint végezte. Ezért a modell csak megjelenítéshez szükséges változókkal bővülhet ki.

View: A grafikus felület fő célja a Viewmodell megjelenítésének megfelelő implementálása. A view egy ablakban lesz a Controller-rel. A modell megjelenítése több új osztály segítségével van megoldva, ezek a későbbi pontokban vannak részletezve.

Controller: Már a korábbi verziók során is része volt, ám a grafikus felülettel célszerű az irányítást is grafikusan megoldani, mivel jóval gyorsabb a korábbi parancssoros megoldásnál. A koordinákat igénylő parancsok először egy helyet kérnek a megjelenített térképről. A parancsok a HitBox osztályból származtatott gombokkal érhetők el.

11.2.2 A felület osztály-struktúrája



11.3 A grafikus objektumok felsorolása

11.3.1 RajzPanel

- **Felelősség**

Ez az osztály felel azért, hogy a benne tárolt elemek megjelenjenek a képernyőn. Ő maga egy megjelenített Swing objektum.

- **Ősosztályok**

JPanel

- **Interfészek**

- **Attribútumok**

- **drawableGameList:ArrayList<Drawable>** ebben a tömbben tároljuk a képernyőn megjelenítendő objektumokat

- **Metódusok**

- **public RajzPanel(ArrayList<Drawable> drawables):** a paraméterül kapott tömbben lévő elemeket megjeleníti.
- **public void setGameDrawables(ArrayList<Drawable> l):** betöltéskor lekérdezi, hogy milyen objektumokat kell majd megjeleníteni
- **public void paintComponent(Graphics g):** végigmegy a tömb elemein és a hitboxokon külön külön majd megjeleníti őket a képernyőn

11.3.2 HitBox

- **Felelősség**

Minden kattintható objektum megjelenéséért ez felelős, például a gombok, a kirajzolt tile-ok.

- **Interfészek**

Drawable

- **Attribútumok**

- **enabled : boolean:** amennyiben az érték igaz, úgy meg van jelenítve az objektum, hamis esetén nincs
- **boxes: ArrayList<HitBox>:** listában tárolja a hitboxokat
- **r: Rectangle:** minden hitbox négyzet alapú, ez a négyzet ez az attribútum
- **text: String:** a hitboxra kiírandó szöveget tárolja

- **Metódusok**

- **public boolean Contains(int x,int y):** igazzal tér vissza, ha a kattintott koordináta benne van a hitboxban
- **public void disable():** az enabled értékét hamisra állítja
- **public void draw(Graphics g):** kirajzolja a hitboxot
- **public void enable():** az enabled értékét igazra állítja
- **public static ArrayList<HitBox> getBoxes():** visszaadja a boxes-ot
- **public HitBox(int x, int y, int width,int height):** a konstruktor, az alapvető attribútumokat adhatjuk itt meg
- **abstract void onClicked(int x, int y):** absztrakt metódus, mások valósítják meg
- **public void setText(String s):** text értékének beállítása
- **public static void thereWasAClick(int x, int y):** vizsgálja, hogy kattintásnak fogható fel vagy sem a hitboxon történt kattintás

11.3.3 Drawable

- **Felelősség**

Interfész biztosít a Game osztály kirajzolásának megvalósításához.

- **Ősosztályok**

• ---

- **Interfészek**

- **Attribútumok**

- **Metódusok**

- **public void draw(Graphics g):** ez a függvény fog majd implementálást követően a kirajzolásért felelni abban az osztályban, ami implementálja az interfész

11.3.4 Creep

- **Felelősség**

Az ellenfelek absztrakt ősosztálya

- **Ősosztályok**

Coordinate

- **Interfészek**

Drawable

- **Attribútumok**

- **Metódusok**

- **public void draw(Graphics g):** biztosítja, hogy példányai elérhetőek a Drawable interface-n keresztül

11.3.5 Tile

- **Felelősség**

A pálya egy mezejét reprezentálja.

- **Ősosztályok**

IndexCoordinate

- **Interfészek**

Drawable

- **Attribútumok**

- **Metódusok**

- **public void draw(Graphics g):** biztosítja, hogy példányai elérhetőek a Drawable interface-n keresztül

11.3.6 Projectile

- **Felelősség**

A lövedéket valósítja meg.

- **Ősosztályok**

Coordinate

- **Interfészek**

Drawable

- **Attribútumok**

- **Metódusok**
- **public void draw(Graphics g):** biztosítja, hogy példányai elérhetőek a Drawable interface-n keresztül

11.3.7 Game

- **Felelősség**

Az MVC architektúrában a modellt jelképezi. Tartalmazza a játék menetéhez tartozó attribútumokat és metódusokat. Ez az osztály felelős az egész játékmenet megfelelő működéséért.

- **Ősosztályok**

- **Interfészek**

Serializable

- **Attribútumok**

- **MapVertical: int :** a pálya vertikális mérete, kattintás esetén az aktuális tile meghatározásában van szerepe
- **MapHorizontal: int :** a pálya horizontális mérete, kattintás esetén az aktuális tile meghatározásában van szerepe
- **drawables: ArrayList<Drawable>:** tárolja azokat az elemeket amiket meg kell majd jelenítenünk a képernyőn

- **Metódusok**

- **Game(ArrayList<Drawable> list):** a paraméterként kapott elemekkel feltölti a drawables tömböt és kezdésnek kikacsolja a ködöt

11.3.8 Main

- **Felelősség**

Kezelőfelületért és annak grafikájáért felelős osztály. Az MVC modellből a Viewt és a Controllert valósítja meg.

- **Ősosztályok**

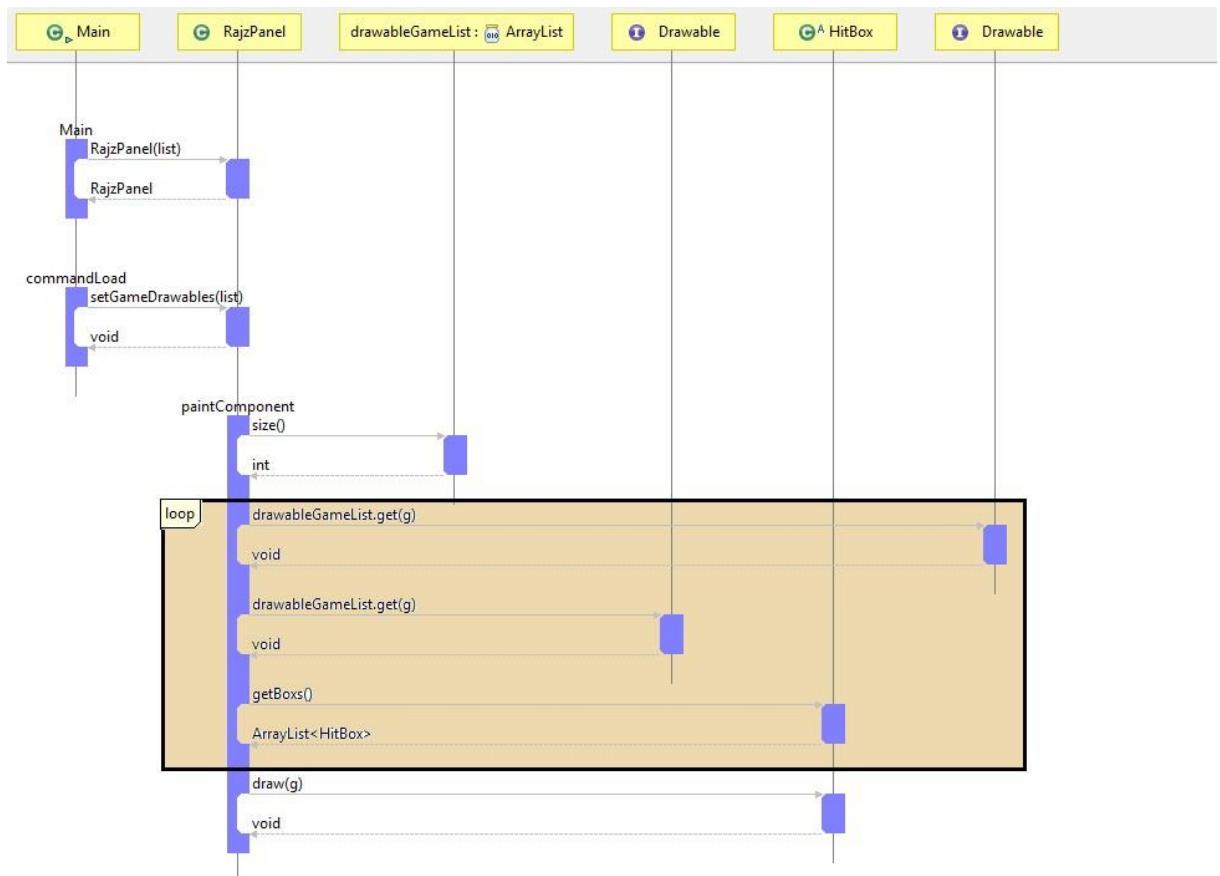
- **Interfészek**

MouseListener

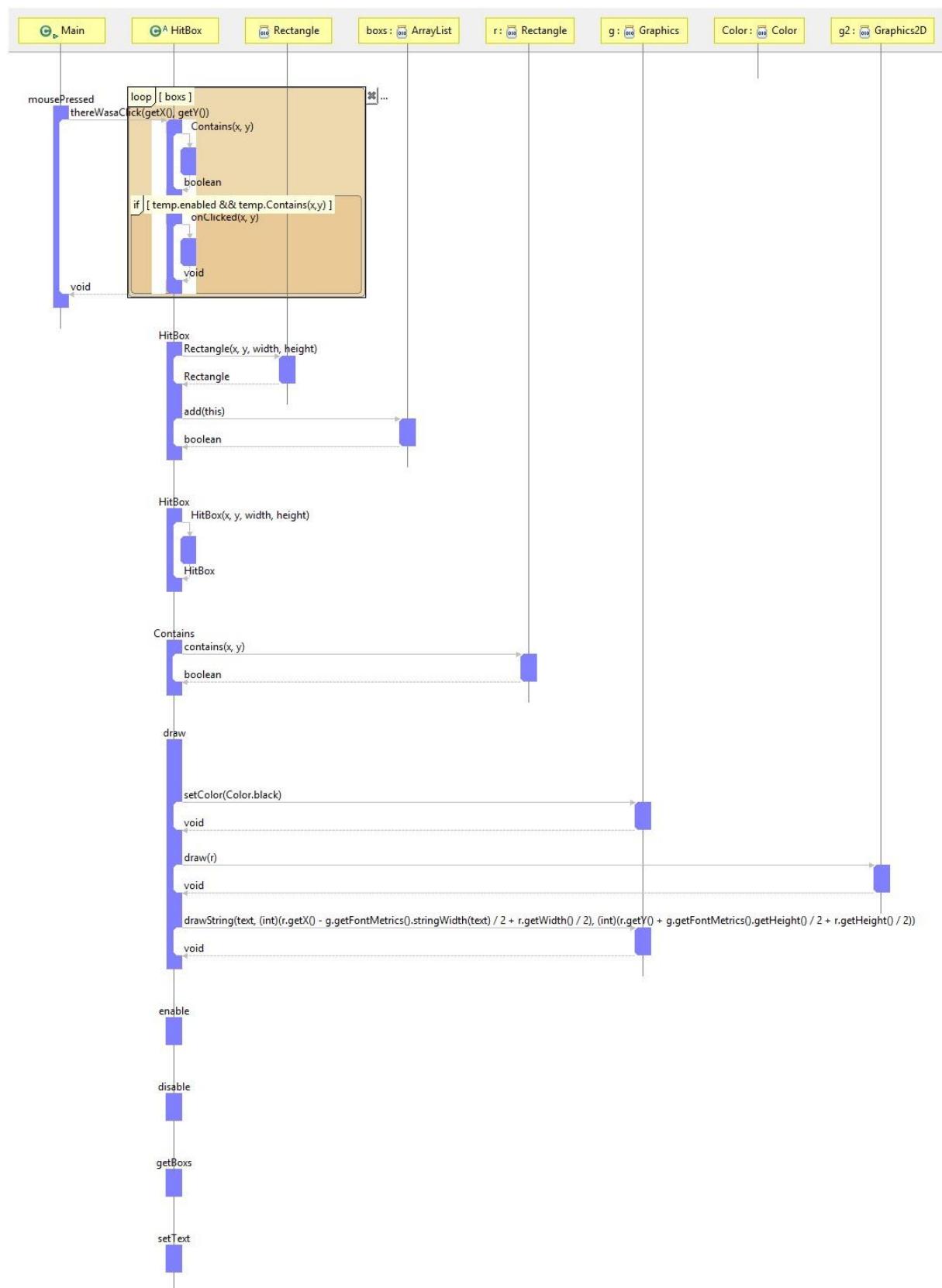
- **Attribútumok**
- **pause: boolean:** amennyiben ez igaz, úgy a játék szünetel
- **jf: JFrame:** az alap Frame
- **list: ArrayList<Drawable>:** kirajzolható elemek listája
- **MenuList: ArrayList<Drawable>:** a menü elemek listája
- **rp: RajzPanel:** a rajzpanel osztályt ezen keresztül érhetjük el
- **exitGame: HitBox:** az exitgame gomb
- **buildTower: HitBox:** a buildtower gomb
- **buildObstacle: HitBox:** a buildobstacle gomb
- **map: HitBox:** A map(pálya) maga. Kezeli a
- **newGame: HitBox:** az új játék gomb
- **mana: HitBox:** a mana megjelenítése
- **time: HitBox:** az idő megjelenítése
- **loadGame: HitBox:** a loadgame gomb
- **quitGame: HitBox:** a quitgame gomb
- **stones: HitBox:** a kövek felsorolásáért felelős hitbox
- **upgrade: HitBox:** az upgrade gomb
- **Metódusok**
- **private void commandLoad():** mentett játék betöltését végzi. Message box-ban kínálja fel a lehetőségeket, amelyek közül közül kiválaszthatjuk a folytatni kívánt játékot.
- **private void commandQuit():** a kilépésért és annak grafikus lebonyolításáért felelős. Kilépés előtt felajánlja a mentés lehetőségét
- **private void gamevis(boolean p):** p.tól függően akívvá teszi a játékon belüli gombokat (pl.: buildTower, buildObstacle...)
- **private void menuvis(boolean p);** p.tól függően akívvá teszi a menü gombjait(newGame, loadGame, exit)
- **private void initBoxes():** inicializálja a kezelő felület gombjait, és a függvényen belül történik az egyes HitBox abstract onClicked() függvényének az implementálása is
- **public Main():** minden egyéb meghívásáért felelős függvény, valamint létrehozza az alapvető grafikai elemeket amiket a későbbiekben felüldefiniálunk. Itt hozzuk létre a log könyvtárakat, illetve indítjuk el a Game doit-szálát.

11.4 Kapcsolat az alkalmazói rendszerrel

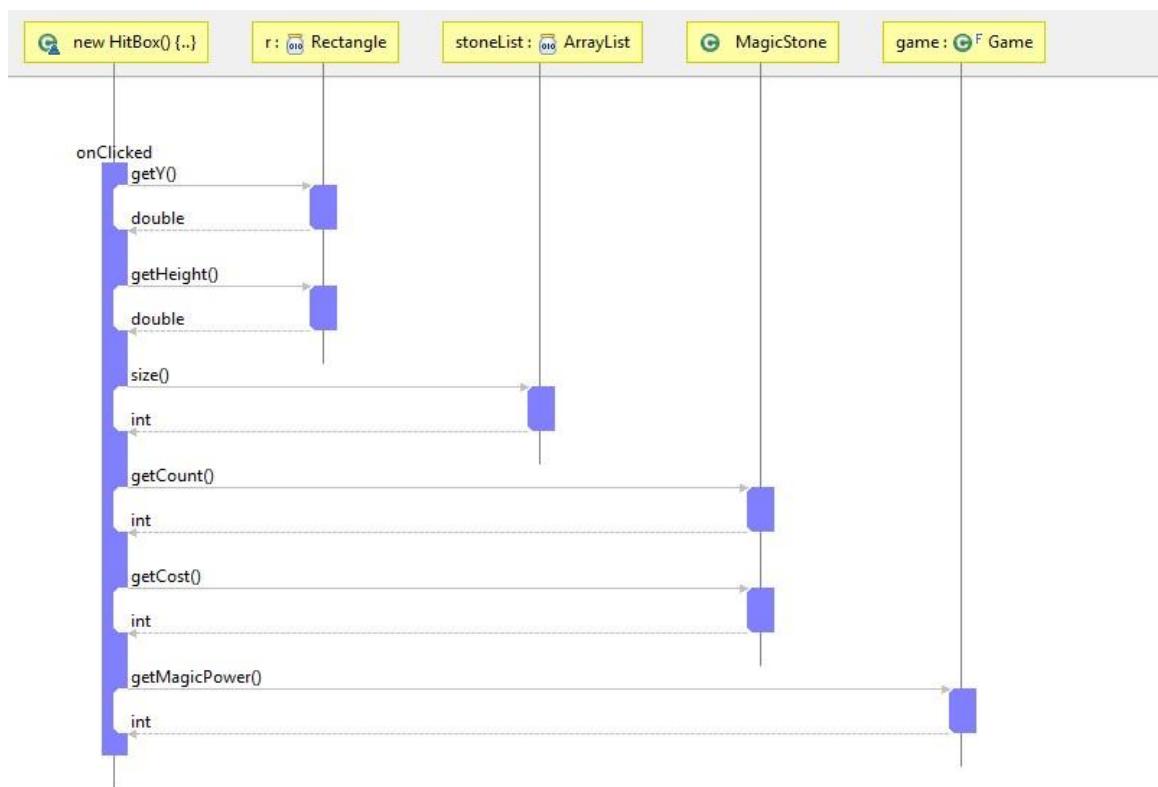
11.4.1 RajzPanel szekvencia



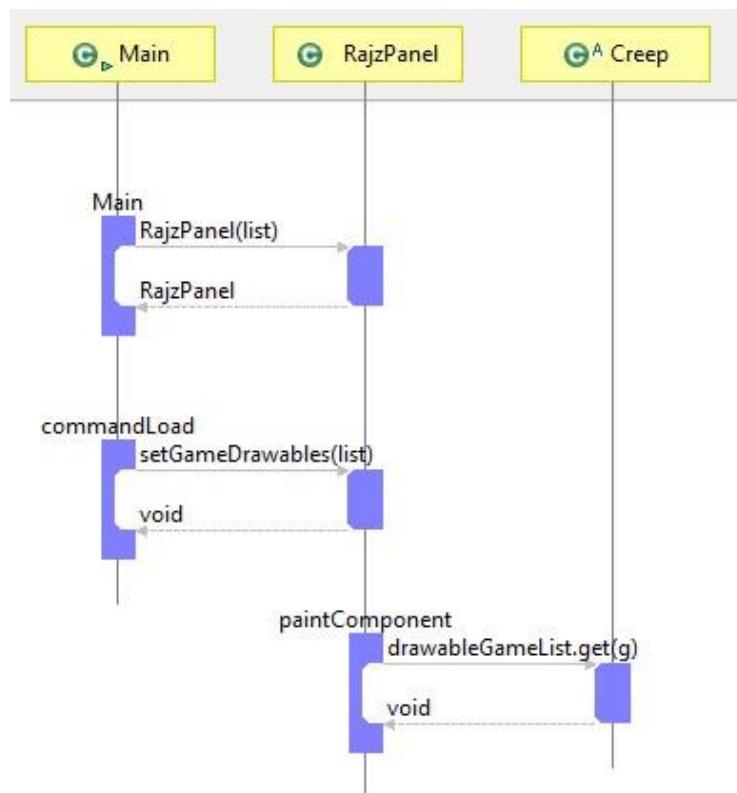
11.4.2 HitBox szekvencia



11.4.3 HitBoxOnClicked szekvencia



11.4.4 CreepDraw szekvencia



Napló

Kezdet	Időtartam	Résztvevők	Leírás
2014.04.26. 14:30	7 óra	Radnai	Grafikus felület implementálása + megbeszélés a többiekkel
2014.04.26. 14:00	2 óra	Molnár Németh Srajner	Grafikus felület megbeszélése
2014.04.27. 11:00	1 óra	Molnár	Grafikus felület tesztelés
2014.04.27. 13:00	3 óra	Radnai	hiányzó elemek implementálása, megmaradt tesztelés során kiderült hibák javítása. (pl köd)

13. Grafikus változat beadása

13.1 Fordítási és futtatási útmutató

13.1.1 Fájllista

config mappa:

Fájl neve	Méret	Keletkezés ideje	Tartalom
config.ini	2 kB	2014.05.10.	konfigurális beállítások
configsafe.ini	2 kB	2014.05.11.	konfigurális beállítások (angol)

images mappa:

Fájl neve	Méret	Keletkezés ideje	Tartalom
3.png	518 byte	2014.05.10.	lövedék képe
4.png	366 byte	2014.05.10.	lövedék képe
bground.jpg	948 kbyte	2014.05.11.	háttér képe
Dwarf11.png	941 byte	2014.05.10.	törp mozdulat 1
Dwarf22.png	922 byte	2014.05.10.	törp mozdulat 2
Dwarf33.png	1120 kbyte	2014.05.10.	törp mozdulat 3
Elf11.png	927 byte	2014.05.10.	elf mozdulat 1
Elf22.png	986 byte	2014.05.10.	elf mozdulat 2
Elf33.png	966 byte	2014.05.10.	elf mozdulat 3
FFMQL_-_Focus_Tower_Sprite_.png	1340 byte	2014.05.10.	Torony képe
Halfling11.png	1050 byte	2014.05.10.	hobbit mozdulat 1
Halfling22.png	957 byte	2014.05.10.	hobbit mozdulat 2
Halfling33.png	978 byte	2014.05.10.	hobbit mozdulat 3
Human11.png	990 byte	2014.05.10.	ember mozdulat 1
Human22.png	987 byte	2014.05.10.	ember mozdulat 2
Human33.png	1022 byte	2014.05.10.	ember mozdulat 3
specprocejtile.png	460 byte	2014.05.10.	speciális lövedék képe

src mappa:

Fájl neve	Méret	Keletkezés ideje	Tartalom
Coordinate.java	1 kB	2014.04.12.	Coordinate osztály forrása
Creep.java	4 kB	2014.04.12.	Creep osztály forrása
Drawable.java	1 kB	2014.04.12.	Drawable osztály forrása
Dwarf.java	2 kB	2014.04.12.	Dwarf osztály forrása
Elf.java	2 kB	2014.04.12.	Elf osztály forrása
Game.java	15 kB	2014.04.12.	Game osztály forrása
GameState.java	1 kB	2014.04.12.	GameState osztály forrása
Halfling.java	2 kB	2014.04.12.	Halfling osztály forrása
HitBox.java	2 kB	2014.04.12.	HitBox osztály forrása
Human.java	2 kB	2014.04.12.	Human osztály forrása
IndexCoordinate.java	2 kB	2014.04.12.	IndexCoordinate osztály forrása
MagicStone.java	2 kB	2014.04.12.	MagicStone osztály forrása
Main.java	25 kB	2014.04.12.	Main osztály forrása

Projectile.java	4 kB	2014.04.12.	Projectile osztály forrása
Race.java	1 kB	2014.04.12.	Race osztály forrása
RajzPanel.java	1 kB	2014.04.12.	RajzPanel osztály forrása
Route.java	5 kB	2014.04.12.	Route osztály forrása
SpecialRoute.java	3 kB	2014.04.12.	SpecialRoute osztály forrása
Tile.java	2 kB	2014.04.12.	Tile osztály forrása
Tower.java	5 kB	2014.04.12.	Tower osztály forrása

waves mappa:

Fájl neve	Méret	Keletkezés ideje	Tartalom
wave0.wav	4 kB	2014.05.11.	elleniséghullámok

13.1.2 Fordítás és telepítés

A fordításhoz, a fordításra használt gépen működnie kell a javac parancsnak a parancssorban. Amennyiben ez nem teljesül, az oracle oldalán lehetséges megoldást lelni.

Ha működik, akkor a parancssoron belül el kell navigálni a kicsomagolt mappa src/noobsaruman könyvtárába, és a következő kódot írni:

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

13.1.3 Futtatás

A futtatáshoz, a futtatásra használt gépen működnie kell a java parancsnak a parancssorban. Amennyiben ez nem teljesül, az oracle oldalán lehetséges megoldást lelni.

Ha működik, akkor a parancssoron belül el kell navigálni a kicsomagolt mappa gyökerébe, és a következő kódot írni:

```
java noobsaruman.Main
```

Ezzel elindul a program.

Megjegyzés: Egyes operációs rendszeri beállítások esetén nem fogadja el a config file-t a program, ekkor a configsafe.ini file-ra kell kicserélni.

13.2 Értékelés

Tag neve	Munka százalékban
Iklódi Eszter	22.37%
Molnár Bence	17.00%
Németh Zsolt	17.00%
Radnai Balázs	22.32%
Srajner Ferenc	21.31%

13.3 **Napló**

Kezdet	Időtartam	Résztvevők	Leírás
2014.05.10	5 óra	Németh Molnár Radnai Srajner	A játék kinézetének, optikai részének véglegesítése
2014.05.11	1 óra	Németh Molnár Radnai Srajner	A dokumentáció és minden más véglegesítése

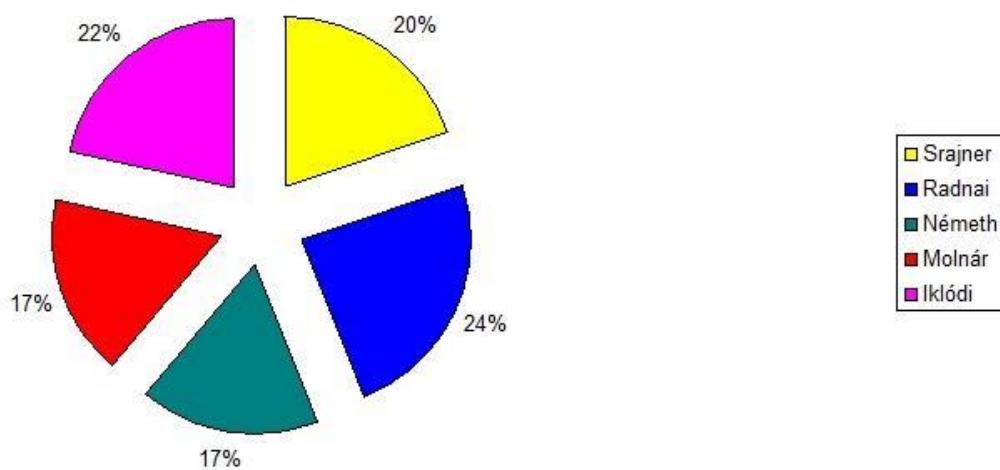
14. Összefoglalás

14.1 Projektre fordított munkaidők

14.1.1. Táblázatosan személyekre lebontva

	Srajner	Radnai	Németh	Molnár	Iklódi	Sum
2. Követelmény, projekt, funkcionálitás	3,5	0,5	3,5	3	5	15,5
3. Analízis modell kidolgozása 1.	16	16	16	18,5	24,5	91
4. Analízis modell kidolgozása 2.	5	1,5	4	4	5	19,5
5. Szkeleton tervezése	3	0	3	1	7	14
6. Szkeleton beadása	3	9	3	3	3	21
7. Prototípus koncepciója	10,5	9,5	5,5	6,5	8,5	40,5
8. Részletes tervezek	4	10,5	2	2	5	23,5
10. Prototípus beadása	3	4,5	2,5	2	3	15
11. Grafikus felület specifikálása	2	10	2	3	0	17
13. Grafikus változat beadása	6	6	6	6	0	24
Sum	56	67,5	47,5	49	61	281
%	19.9	24	16,9	17,4	21,8	

Munkaelosztás



14.1.2. Összes napló

2. hét

Kezdet	Időtartam	Résztvevők	Leírás
2014.02.19. 8:30	0,5 óra	Molnár Németh Srajner Iklódi	Értekezlet. Alapkoncepció és követelmények átbeszélése, ötletelés
2014.02.20.18:00	2 óra	Iklódi	use-case elemek meghatározása, leírása, dokumentum töltése
2014.02.20.18:00	2 óra	Németh	Funkciók (2.2.2)
2014.02.20. 18:15	2 óra	Srajner	dokumentáció bővítése, Projekt terv, Követelmények (>2.3.1)
2014. február 21.	1,5 óra	Molnár	dokumentáció áttekintése, formázása, funkciók (2.2.2) kiegészítése, feladatkörök, 2.1.*
2014. február 21.	0,5 óra	Radnai	áttekintés, appróbb formai módosítások.
2014.02.22. 10.00	1 óra	Németh	Szótár (2.5)
2014.02.22. 10:00	1 óra	Molnár	Szótár (2.5)
2014.02.22. 10:30	1 óra	Srajner	Projekt terv kiegészítés (2.6.3 +) (2.1.3)
2014.02.22. 12:45	2,5 óra	Iklódi	dokumentáció felülvizsgálata

3. hét

Kezdet	Időtartam	Résztvevők	Leírás
2014.02.26. 8:00	2 óra	Iklodi Molnár Németh Radnai Srajner	Konzultáció a konzulenssel, analízis modell fő vázának egyeztetése

2014.02.27. 13:00	2 óra	Iklodi Molnár Németh Radnai Srajner	gyűlés, objektumok összegyűjtése
2014. 02. 27. 15:00	2 óra	Srajner Radnai Iklódi	osztálydiagram első verziójának megrajzolása, és hozzá egyes funkciók megtervezése
2014.03.01. 15:00	5 óra	Iklodi Molnár Németh Radnai	Osztálydiagramm átbeszélése és megalkotása
2014.03.01. 20:00	5 óra	Iklódi Molnár Németh Srajner	Szekvenciadiagrammok elkészítése, osztálydiagramm és szekvencia diagramm összehangolása
2014.03.02 12:00	0,5 óra	Srajner	State Chartok digitális megjelenítése
2014.03.02. 11:00	3 óra	Iklódi	State Chartok megrajzolása, doksi szerkesztése
2014.03.02. 11.00	2 óra	Németh	Szekvenciadiagrammok,state-chartok beszerkesztése, dokumentumok ellenőrzése
2014.03.02. 17:00	5,5 óra	Iklódi	Dokumentum szerkesztése, szekvencia diagramok, osztálydiagram módosítása
2013.03.02. 17:00	5 óra	Radnai	dokumentum átvizsgálása, hiányok pótlása,szekvencia diagramok lefedettségének vizsgálata
2014.03.02. 18:00	4,5 óra	Molnár Srajner	Dokumentum szerkesztése, osztálydiagram és szekvencia diagram készítése

4. het

Kezdet	Időtartam	Résztvevők	Leírás
2014.03.05. 8:00	1,5 óra	Iklodi Molnár Németh Srajner	Konzultáció a konzulenssel, analízis modell hibáinak feljegyzése
2014.03.07. 13:00	2 óra	Iklodi Molnár Németh Srajner	A felmerülő hibák egyeztetése, kijavítása

2014. 03.08. 20:00	1,5 óra	Iklodi Radnai Srajner	Hiányos részek pótlása, szekvencia és osztálydiagramok átnézése
2014. 03.09. 15:00	0,5	Molnár Németh	Dokumentáció formázása, lezárása

5. hét

Kezdet	Időtartam	Résztvevők	Leírás
2014.03.14. 18:30	1 óra	Iklódi	use-case diagramok elkészítése
2014. 03.14. 21:15	1 óra	Molnár	use-case leírások elkészítése
2014. 03.15. 16:00	6 óra	Iklódi	szekvenciadiagramok megtervezése, forgatókönyv, szkeleton dialógus megírása, koordinálás
2014.03.15. 16:00	3 óra	Srajner	szekvenciadiagramok megszerkesztése
2014.03.15 20:00	3 óra	Németh	kommunikációs diagramok megszerkesztése

6. hét

Kezdet	Időtartam	Résztvevők	Leírás
2014.03.19. 08:15	2 óra	Iklódi Srajner Németh Molnár	szkeleton terv átbeszélése
2014.03.19. 10:20	1 óra	Iklódi Srajner Radnai	a szkelton program felületének lekódolása
2014.03.22. 12:00	6 óra	Radnai	a program megírása, tesztelése
2014.03.22. 14:00	1 óra	Molnár	kódolás közben felmerülő kérdések pontosítása
2014.03.23. 10.00	1 óra	Németh	Kommunikációs diagramok újraraírozása
2014.03.23. 10:00	2 óra	Radnai	program véglegesítése, szekvencia diagrammok módosítása

7. hét

Kezdet	Időtartam	Résztvevők	Leírás
2014.03.28. 18:00	2,5 óra	Iklódi Németh Molnár Srajner	kupaktanács
2014.03.29. 3:00	4 óra	Srajner	prototípus dokumentációnak prototípusának elkészítése
2014.03.29. 13:30	4,5 óra	Radnai	a játék program írása, kezdetleges grafikus megjelenítéssel, hogy írás közben könnyen tesztelhető legyen. ebből fog majd készülni a prototípus
2014.03.30. 00:30	1,5 óra	Iklódi	Tesztesetek megírása
2014.03.30. 19:15	3 óra	Radnai	Log eljárások tervezése, dokumentálása
2014.03.30. 21:00	1 óra	Molnár	Kimeneti,bemeneti nyelv szerkesztése
2014.03.30. 23:00	1,5 óra	Iklódi	Dokumentum letisztázása
2014.03.31. 10:00	3 óra	Iklódi Molnár Németh Srajner	Dokumentum hiányosságának pótlása, végső módosítások véglegesítése
2014.04.06. 01:00	1 óra	Srajner	utómódosítások gui szó kiszedése, split módosítása, stb.

8. hét

Kezdet	Időtartam	Résztvevők	Leírás
2014.04.02 20:45	3 óra	Radnai	prototípus írása, a teszteset elkészítése és ellenőrzése a programmal
2014.04.03. 20:15	1 óra	Molnár	dokumentáció töltése (Creep osztály)
2014.04.03. 16.00	1 óra	Németh	dokumentáció töltése (Tower osztály)
2014.04.04. 10:30	1 óra	Molnár	dokumentáció (MagicStone)
2014.04.04. 11:30	1 óra	Németh	dokumentáció (Route,Coordinate,Tile)
2014.04.05 02:30	4 óra	Srajner	Game osztály véglegesítése, Többi osztály tisztázása

2014.04.05 17:00	7,5 óra	Radnai	prototípus megírása addig, hogy a tesztesetek futtathatóak legyenek. minden funkció működőképes, logok és pár appróbb dolog hiányzik csak
2014.04.06. 15:00	5 óra	Iklódi	tesztesetek be- és kimeneteinek megírása, dokumentum formázása

10. hét

Kezdet	Időtartam	Résztvevők	Leírás
2014.04.12.	0,5 óra	Iklódi Radnai Molnár Németh Srajner	program megkezdése
2014.04.19 13:00	4 óra	Radnai	prototípus befejezése, tesztesetek lefuttatása, kijavítása.
2014.04.21. 14:30	1,5 óra	Molnár	Dokumentáció készítése
2014.04.21 14:30	2,5 óra	Iklódi	Dokumentálás, tesztesetek ellenőrzése, véglegesítése
2014.04.21 20:45	2 óra	Németh	Implementáció Eclipse Helios kompatibilissá tétele, kommentezés
2014.04.22 01:00	2,5 óra	Srajner	Compare program implementálása és végleges ellenőrzés

11. hét

Kezdet	Időtartam	Résztvevők	Leírás
2014.04.26. 14:30	7 óra	Radnai	Grafikus felület implementálása + megbeszélés a többiekkel
2014.04.26. 14:00	2 óra	Molnár Németh Srajner	Grafikus felület megbeszélése
2014.04.27. 11:00	1 óra	Molnár	Grafikus felület tesztelés
2014.04.27. 13:00	3 óra	Radnai	hiányzó elemek implementálása, megmaradt tesztelés során kiderült hibák javítása. (pl köd)

13. hét

Kezdet	Időtartam	Résztvevők	Leírás
2014.05.10	5 óra	Németh Molnár Radnai Srajner	A játék kinézetének, optikai részének véglegesítése
2014.05.11	1 óra	Németh Molnár Radnai Srajner	A dokumentáció és minden más véglegesítése

14.2. A feltöltött programok forrássorainak száma

14.2.1. Skeleton

Fájl neve	Méret	Keletkezés ideje	Tartalom	Sorok száma
coordinate.java	1 KB	2014.03.23 10:57	Coordinate class és függvényei	39
creep.java	2 KB	2014.03.23 10:57	Creep class és függvényei	70
dummy.java	12 KB	2014.03.23 10:57	A skeleton kezelő class-ja	350
dwarf.java	1 KB	2014.03.23 10:57	Dwarf class és függvényei	19
elf.java	1 KB	2014.03.23 10:57	Elf class és függvényei	19
halfling.java	1 KB	2014.03.23 10:57	Halfling class és függvényei	19
human.java	1 KB	2014.03.23 10:57	Human class és függvényei	18
indexcoordinat e.java	1 KB	2014.03.23 10:57	IndexCoordinate class és függvényei	29
magicstone.jav a	2 KB	2014.03.23 10:57	MagicStone class és függvényei	52

main.java	1 KB	2014.03.23 10:57	Main class	23
projectile.java	1 KB	2014.03.23 10:57	Projectile class és függvényei	37
route.java	2 KB	2014.03.23 10:57	Route class és függvényei	64
tile.java	1 KB	2014.03.23 10:57	Tile class és függvényei	25
tower.java	2 KB	2014.03.23 10:57	Tower class és függvényei	61

14.2.2. Prototípus

Fájl neve	Méret	Keletkezés ideje	Tartalom	Sorok száma
config.ini	1 KB	2014.04.21. 10:43	Kezdeti pálya,egység és egyéb indulási feltételek, tulajdonságok megadása, betöltése a játék kezdetéhez.	65
1.makro	1 KB	2014.04.21. 10:27	8.2.1 teszeset parancsait tartalmazó input fájl	4
2.makro	1 KB	2014.04.21. 10:27	8.2.2 teszeset parancsait tartalmazó input fájl	10
3.makro	1 KB	2014.04.21. 10:27	8.2.3 teszeset parancsait tartalmazó input fájl	10
4.makro	1 KB	2014.04.21. 10:27	8.2.4 teszeset parancsait tartalmazó input fájl	18
5.makro	1 KB	2014.04.21. 10:27	8.2.5 teszeset parancsait tartalmazó input fájl	20

6.makro	1 KB	2014.04.21. 10:27	8.2.6 teszteset parancsait tartalmazó input fájl	20
7.makro	1 KB	2014.04.21. 10:44	8.2.7 teszteset parancsait tartalmazó input fájl	11
8.makro	1 KB	2014.04.21. 10:47	8.2.8 teszteset parancsait tartalmazó input fájl	15
9.makro	1 KB	2014.04.21. 10:51	8.2.9 teszteset parancsait tartalmazó input fájl	15
10.makro	1 KB	2014.04.21. 10:54	8.2.10 teszteset parancsait tartalmazó input fájl	12
11.makro	1 KB	2014.04.21. 10:57	8.2.11 teszteset parancsait tartalmazó input fájl	14
12.makro	1 KB	2014.04.21. 11:01	8.2.12 teszteset parancsait tartalmazó input fájl	14
Coordinate.java	1 KB	2014.04.12.	Coordinate class és függvényei	34
Creep.java	3 KB	2014.04.12.	Creep class és függvényei	107
Dwarf.java	1 KB	2014.04.12.	Dwarf class és függvényei	25
Elf.java	1 KB	2014.04.12.	Elf class és függvényei	24
Game.java	13 KB	2014.04.12.	Game class és függvényei	360
GameState.java	1 KB	2014.04.12.	GameState class és függvényei	18
Halfling.java	1 KB	2014.04.12.	Halflingclass és függvényei	22
Human.java	1 KB	2014.04.12.	Humanclass és függvényei	22
IndexCoordinate.java	2 KB	2014.04.12.	IndexCoordinate class és függvényei	81
MagicStone.java	2 KB	2014.04.12.	MagicStone class és függvényei	64
Main.java	16 KB	2014.04.12.	Main class és függvényei	375

Projectile.java	3 KB	2014.04.12.	Projectile class és függvényei	88
Race.java	1 KB	2014.04.12.	Race class és függvényei	20
Route.java	4 KB	2014.04.12.	Route class és függvényei	111
SpecialRoute.java	2 KB	2014.04.12.	Special Route class és függvényei	76
Tile.java	1 KB	2014.04.12.	Tile class és függvényei	21
Tower.java	4 KB	2014.04.12.	Tower class és függvényei	100
2.wav	1 KB	2014.04.06.	Komplettebb wave input	15
wave1.wav	1 KB	2014.04.02.	Tesztekhez használt alap wave input	1

14.2.3. Grafikus

Fájl neve	Méret	Keletkezés ideje	Tartalom	Sorok száma
config.ini	2 kB	2014.05.10.	konfigurális beállítások	81
configsafe.ini	2 kB	2014.05.11.	konfigurális beállítások (angol)	81
3.png	518 byte	2014.05.10.	lövedék képe	-
4.png	366 byte	2014.05.10.	lövedék képe	-
bground.jpg	948 kbyte	2014.05.11.	háttér képe	-
Dwarf11.png	941 byte	2014.05.10.	törp mozdulat 1	-
Dwarf22.png	922 byte	2014.05.10.	törp mozdulat 2	-
Dwarf33.png	1120 kbyte	2014.05.10.	törp mozdulat 3	-
Elf11.png	927 byte	2014.05.10.	elf mozdulat 1	-
Elf22.png	986 byte	2014.05.10.	elf mozdulat 2	-
Elf33.png	966 byte	2014.05.10.	elf mozdulat 3	-
FFMQL_-_Focus_Tower_Sprite.png	1340 byte	2014.05.10.	Torony képe	-
Halfling11.png	1050 byte	2014.05.10.	hobbit mozdulat 1	-
Halfling22.png	957 byte	2014.05.10.	hobbit mozdulat 2	-
Halfling33.png	978 byte	2014.05.10.	hobbit mozdulat 3	-
Human11.png	990 byte	2014.05.10.	ember mozdulat 1	-
Human22.png	987 byte	2014.05.10.	ember mozdulat 2	-
Human33.png	1022 byte	2014.05.10.	ember mozdulat 3	-

specproejtile.p ng	460 byte	2014.05.10.	speciális lövedék képe	-
Coordinate.java	1 kB	2014.04.12.	Coordinate osztály forrása	23
Creep.java	4 kB	2014.04.12.	Creep osztály forrása	95
Drawable.java	1 kB	2014.04.12.	Drawable osztály forrása	9
Dwarf.java	2 kB	2014.04.12.	Dwarf osztály forrása	31
Elf.java	2 kB	2014.04.12.	Elf osztály forrása	31
Game.java	15 kB	2014.04.12.	Game osztály forrása	376
GameState.java	1 kB	2014.04.12.	GameState osztály forrása	11
Halfling.java	2 kB	2014.04.12.	Halfling osztály forrása	36
HitBox.java	2 kB	2014.04.12.	HitBox osztály forrása	50
Human.java	2 kB	2014.04.12.	Human osztály forrása	31
IndexCoordinat e.java	2 kB	2014.04.12.	IndexCoordinat e osztály forrása	65
MagicStone.jav a	2 kB	2014.04.12.	MagicStone osztály forrása	54
Main.java	25 kB	2014.04.12.	Main osztály forrása	536
Projectile.java	4 kB	2014.04.12.	Projectile osztály forrása	86
Race.java	1 kB	2014.04.12.	Race osztály forrása	14
RajzPanel.java	1 kB	2014.04.12.	RajzPanel osztály forrása	28
Route.java	5 kB	2014.04.12.	Route osztály forrása	111
SpecialRoute.ja va	3 kB	2014.04.12.	SpecialRoute osztály forrása	67
Tile.java	2 kB	2014.04.12.	Tile osztály forrása	26
Tower.java	5 kB	2014.04.12.	Tower osztály forrása	97
wave0.wav	4 kB	2014.05.11.	ellenséghullám ok	45

14.3. Projekt összegzés:

Mit tanultak a projektből konkrétan és általában?

Konkrétan:

Iklódi Eszter

Végre egy világosabb képet kaptam a szoftverfejlesztés mechanizmusáról. Nagyon jó kis tanuló projekt volt. Szoftvertechnológiát hallgatva még meglehetően ködös volt egy analízis modell fogalma, azonban ezt a félévet végigcsinálva már összeállt a fejemben egy átfogó kép a RUP-ról. Meglepő volt, hogy mennyi munka volt az elején (főleg az analízis modellel), és hogy milyen nehéz is, amikor az embernek papíron kell kitalálnia a szekvenciákat. És akármennyire is próbáltunk precízek lenni, a kódolás során így is derültek ki hiányosságok. Bár én személyesen a munka második felében kevésbé vettem ki a részem, a dokumentációk szerkesztésénél, és a kommunikációnál is látszott, hogy a már túl vagyunk a munka nehezén.

Srajner Ferenc

- A megrendelővel való kommunikáció nagyon fontos (jelen esetben a konzulens). A heti egy órát kevésnek éreztem. Valamint a specifikáció több pontja is pontatlan volt. Mi ezt úgy próbáltuk sokáig korrigálni, hogy nem kötöttünk le dolgokat az elején, ám amikor eljutottunk addig, hogy specifikálnunk kellett, egyet nem értés volt a megbízóval.

- Már egy elenyésző méretű program is vehemens mennyiséggű dokumentációval rendelkezik. A jövőben, ha valóban használható dokumentációt kéne készítenem, mindenképp keresnék erre alkalmas programot, a sablonok véleményem szerint nem tették sokkal átláthatóbbá a helyzetet.

Molnár Bence

Mennyire fontos tud lenni az előre történő pontos specifikáció és a csapattal ill. konzulenssel való hosszas egyeztetés egyes komolyabb kérdés megvitatása esetén. Újabb és újabb problémák a legpontosabb megbeszélések és elgondolások után is képesek újra és újra felmerülni.

Németh Zsolt

Megtanultam hogy mik a prioritási sorrendek, hogyan kell egy projektben csoportban dolgozni, közösen írni a dokumentációt valamint szerkeszteni felsőbb utasításra a diagrammokat. Az esetleges változásokat a projektben megtanultuk jól kezelni, visszamenni egészen a kezdetekig és újragondolni akár a projekt alapelveit is.

Általában:

Iklódi Eszter

Csapatban dolgozni nehéz. Főként a kommunikáció miatt. Még akkor is nehéz, ha mindenki lelkes, és szinte kéri a teendőt. Nem egyszerű a feladatok kiosztása sem, megtalálni, hogy kinek milyen munkát lehet úgy odaadni, hogy azt gond nélkül, legjobb tudása szerint meg tudja csinálni. Számomra világossá vált a menedzser, mint személy létfogosultsága.

Sajnos, amit nem tanultam meg, az az, hogy hogyan lehet a kódon együtt dolgozni. Ugyanis nálunk a képességek, és a motivációk miatt úgy alakult a helyzet, hogy a kódolást lényegében egy ember

végezte. Alapvetően nekem nem tetszett ez a felosztás, de hatékonysági szempontból, és az egyéb elfoglaltságok miatt végül is ez maradt.

Srajner Ferenc

Miközben végigléptünk a RUP pontjain rengeteg kérdés merült fel bennem, és teljesen átlátható képet nem tudtam alkotni a szoftverfejlesztésről.

Nem tudtuk pontosan a szoftvertechnológián tanult módon véghez vinni a feladatot, mivel minden újdonságnak számított. Ennek ellenére úgy vélem, hogy az általunk kért feladatokat az elvártaknak megfelelően teljesítettük, és első próbálkozásnak nagyon hasznos volt. Úgy érzem több szoftverlabor 4-szerű tárgyat kéne végigcsinálnom, hogy igazán átláthassam a feladatokat, és mind a dokumentáció, mind a kód harmóniában legyen.

Molnár Bence

Az alapvető csapatunka és közös megvalósítás alapjainak megismerése. Egy "nagy" projekt elkészítéséről alkothattam némi vázlagos képet, amelyet a későbbiek során remélhetőleg fel tudok majd használni a munkám során is.

Németh Zsolt

Általában megtanultam csoportban dolgozni, a csoport vezető fontosságát, a vitás kérdések megoldásának menetét. Remélhetőleg nagyobb projektekben fel tudom majd használni az itt megszerzett tapasztalatokat.

Mi volt a legnehezebb és a legkönnyebb?

nehéz:

A legnehezebb egyértelműen az egyeztetés volt. Mivel mindenki legalább egy másik ember konfirmálását igényelte minden elvégzett munka után, így meglehetősen megnövekedett a kommunikációs overhead. A Google-doksi közös szerkesztése jó dolog, azonban egy-egy hosszabb fejezetnél, már igen áttekinthetetlen és lassan fut a gépen. Nehéz volt még a konfigurációs menedzsment, hogy olyan batch file-t tudunk írni, vagy olyan egyértelmű utasítássorozatot tudunk megadni, aminek segítségével a tesztelő csapat gond nélkül futtatni tudja a programunkat. A tesztelésekben szinte minden csoportnak gondjai voltak ezen a ponton a mi laborunkban.

könnyű:

Jó volt, hogy senkit nem kellett noszogatni, és hogy mindenki lelkesen állt neki a munkának. Mivel a csapat nagy része egy szobában lakik, így könnyű volt megbeszéléseket tartani. A kódolás részével sem akadtak gondok, mivel a fent említett indokok miatt sajnos egy emberre hárult a munka.

Összhangban állt-e az idő és a pontszám az elvégzendő feladatokkal?

A félév elején határozottan több teendő akadt egy hétre, de ez egyáltalán nem volt probléma, mivel a zárthelyik és egyéb házi feladatok pedig pont azokra a hetekre estek, ahol ebből a tárgyból volt egy kisebbfajta lazulás. Így szerintem a feladatok időbeosztása teljesen rendjén van.

Én a pontozásban sem találtam kivétnivalót, talán csak annyit, hogy a mi csapatunk az analízis modell 1-en már nagyon sokat dolgozott, ellenben a 2-n jóval kevesebbet, és ugye a 1. ért 20 pontot, a 2. pedig 30-at. Ezzel azonban azért nincs gond, mert lényegében itt 50 pont jár összesen egy két

részletben elvégezhető feladatra. Ami nem mindig volt tiszta, az hogy pontosan miért is annyi pontot kapunk amennyit.

A krediteket nézve meglepő tapasztalat, hogy egy emberre tényleg kb 60 munkaóra jutott. Ez ugye pontosan a két kredit "definíciója", mi mégis soknak éreztük a munkát. Talán azért van ez így, mert a 60 óra az iskolai kereteken kívül történt (konzultációkat csak akkor naplóztuk, ha utána tényleges megbeszélés volt), és mert hétről hétre folyamatosan kellett dolgozni. Továbbá valószínű, hogy a több kredites tárgyakra nem igaz az n*30 óra tanulás.

Ha nem, akkor hol okozott ez nehézséget?

Furcsa tapasztalatot nyújtottak a kapott pontszámok. Volt, hogy az egész hétvégét végigdolgoztuk (analízis modell 1.), és a pontszám nem tükrözte a befektetett energiát. Máskor szinte lezsernek mondható munkára maximális pontot kaptunk. Pozitívum azonban, hogy a kérdéses részknél a konzulenssel történő egyeztetés után a pontszámok módosításra kerültek. Egy apró megjegyzésként még hozzáttenném, hogy sokszor piros javításokkal csak egy-egy alrésznek a x.x.1. fejezetében találkoztunk. Továbbá többször úgy éreztem, hogy csak a dokumentum formaiságai, amik számítottak, és az, hogy a modell logikailag mennyire helytálló, háttérbe szorult.

Milyen változtatási javaslatuk van?

A bemutatásnál a pontszámot az alapján kapjuk, amit a tesztelő csapat írt a munkánkról. Világos a koncepció, azonban szerintem nem elvetendő, hogy a konzulens ne pusztán a teszt jegyzőkönyvekre alapozva adjon pontszámot, hanem önmaga is vessen egy-két pillantást a kódra, vagy akár futtassa is. Sajnos nem minden csapat egyformán bírál. És ez nem csak a tagok jófejsége miatt van így, hanem mert egyszerűen sokunknak új még ez a terület, és nem is tudjuk pontosan, hogy milyen is mondjuk egy prototípusnak egy jó logolása.

Milyen feladatot ajánlanának a projektre?

Szerepjáték, amiben egy egyetemista életét lehet szimulálni:

Lehetőség van dönten a tanulás, közélet és pihenés (valamint esetleg egyéb tevékenységek) között, és ennek függvényében kapják a diplomát a játékosok. Azonban, hogy ne csak a tanulást érje meg kihasználni, egyetem után munkát csak közélet során szerzett ismeretséggel lehet találni, valamint ha a kipihentség túl alacsony, akkor se a tanulás, se a közéleti tényezők nem effektívek. Ezen kívül a szabadidőben további rész területeit lehet tanulni a szakmának, ami szintén segíthet egy jó munka találásában. A játék elején meg lehet határozni speciális tulajdonságokat, amik segíthetnek egyes tevékenységek effektívségén.