7. Prototípus koncepció

40 – *[scrum_that]*

Konzulens:
Szabó Ádám Imre

Csapattagok:

Kovács Levente Ákos	CM6UKU	vazul250@gmail.com
Lovász Attila Bence	INCMI7	attonet2@gmail.com
Graics Vince	HY9XQ6	wince17@gmail.com
Magyar Milán Bertalan	MCDNQL	milangfx@gmail.com
Tóth Krisztián Dávid	J38GIK	tht.krisztian@gmail.com

2015. március 29.

Tartalomjegyzék

0. Módosítások

0.1. Változások a specifikációban

0.1.1. Ragacs eltűnése

A ragacs eltűnik a pályáról, miután négy robot ráugrott (elkopik)

Az Obstacle ősosztályban bevezettünk egy int típusú lifetime változót, amit a ragacs konstruktorában 4-re inicializálunk. Ha egy robot belelép a ragacsba, akkor ezt a változót eggyel csökkentjük, illetve ha elérte a 0-t, akkor értelemszerűen már nincs hatással a robotokra.

0.1.2. Olaj felszáradása

Egy meghatározott idő letelte után az olajfolt eltűnik a pályáról (felszárad)

Az említett lifetime változót az olaj konstruktorában 15-re inicializáljuk és mindig eggyel csökkentjük, amikor a robotok ugranak. Így 15 ugrás után már nem lesz hatással a robotokra, eltűnik.

0.1.3. Tisztító kisrobotok

A pályára időnként keményen dolgozó kisrobotok jutnak be, akik szépen sorban feltakarítják a foltokat. A kisrobot egységnyi sebességgel halad, adott ideig (pl. két kör) takarítja a foltot. A folt feltakarítása után a legközelebbi folthoz indul. Ha egy robot ráugrik, akkor a kisrobot megsemmisül és olajfolt kerül a helyére; a ráugró robot nem szenved sérülést. Ha a kisrobot másik kisrobotnak vagy robotnak ütközik, irányt vált.

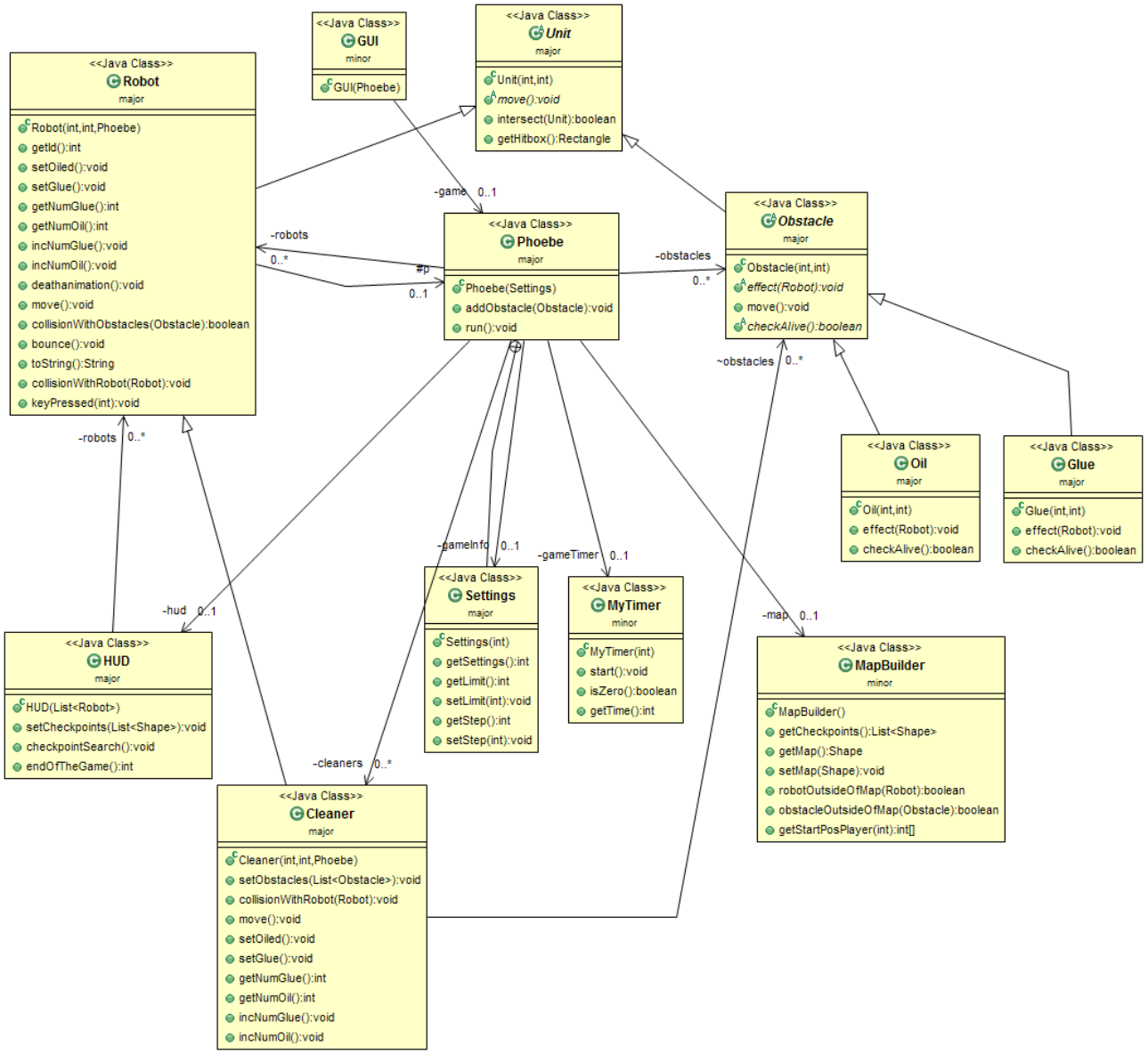
Létrehoztunk egy új Cleaner osztályt, ami a Robotból származik le. Letiltottuk azokat az örökölt metódusokat, amik a tisztító robotra nem érvényesek. Ilyen például a setOiled() vagy setGlue(), hiszen a tisztító robotra ezek az akadályok nincsenek hatással. Mivel a tisztító robotnak tudnia kell, hogy melyik akadály van hozzá legközelebb, ezért egy tagváltozóként megkapja a pályán levő összes akadály listáját. Ezen a listán végigiterálva számolja ki az Obstacle visszatérésű nextObstacle() metódussal, hogy melyik folthoz induljon. Felül kellett definiálni a move() metódust, hiszen a kisrobot állapotától (mozog vagy takarít) függ, hogy hogyan viselkedik. Ha a kisrobot MOVING állapotban van, akkor folyamatosan kiszámolja, hogy melyik folt van hozzá a legközelebb és addig megy, amíg azt el nem éri. Ha elért egy folthoz, akkor WORKING állapotba kerül. WORKING állapotban 3 lépésig a folton dolgozik, majd ha letelt a 3 lépés, akkor feltakarítja a foltot és újra MOVING állapotba kerül.

0.1.4. Robotok ütközése

A robotok képesek ütközni, ha azonos helyre érkeznek ugrásuk végén. Ilyenkor a gyorsabb robot összetöri (megsemmisíti) a lassabbat, és kettejük átlagsebességével halad tovább (vektorátlag!).

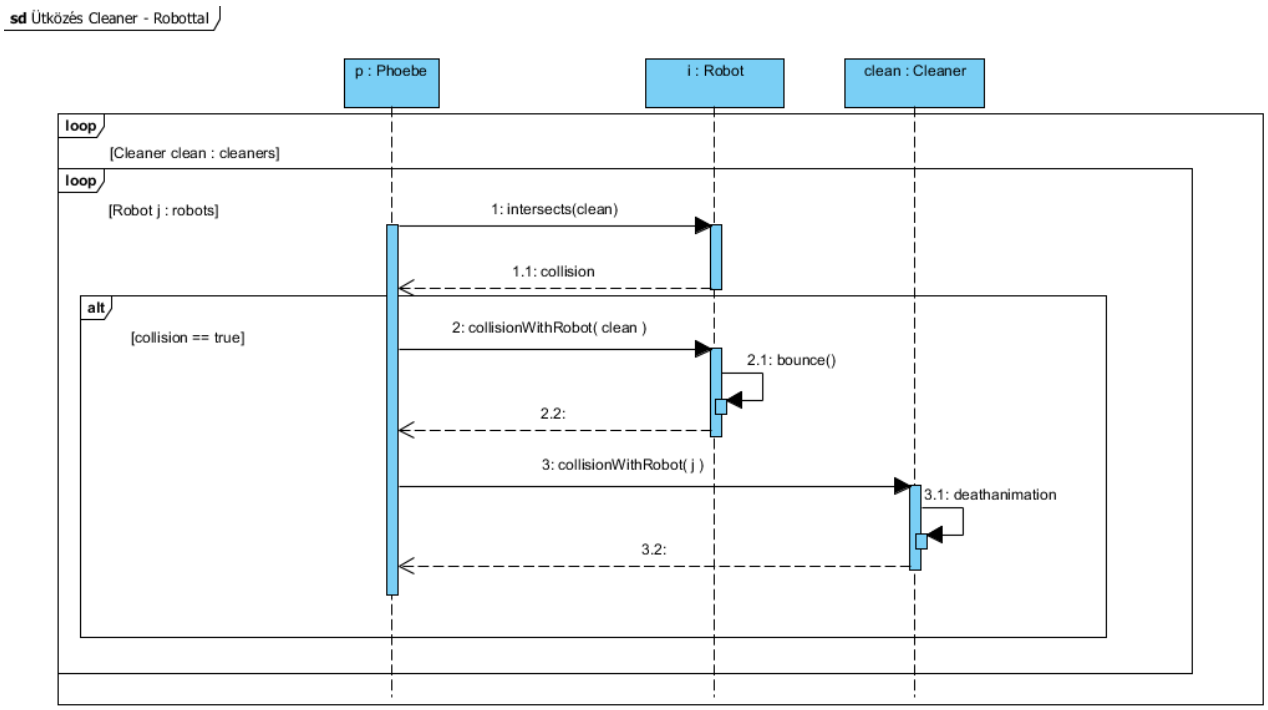
A feladatot eredetileg is úgy specifikáltuk, hogy a robotok ütközhetnek egymással, viszont az ütközések kimenetele mindig az volt, hogy a robotok lepattannak egymásról és másik irányban haladnak tovább. Mivel az implementációnkban két robot sebessége csak akkor különbözhet, ha az egyik ragacsba lépett, ezért megnézzük, hogy a robotok slowed attribútuma egyenlő-e. Ha igen, akkor az eredetileg is specifikált bounce() metódus hívódik meg, tehát a robotok lepattannak egymásról. Ha az egyik robot nagyobb slowed értékkel rendelkezik, mint a másik, akkor megöli az ellenfelét.

0.2. Osztálydiagram

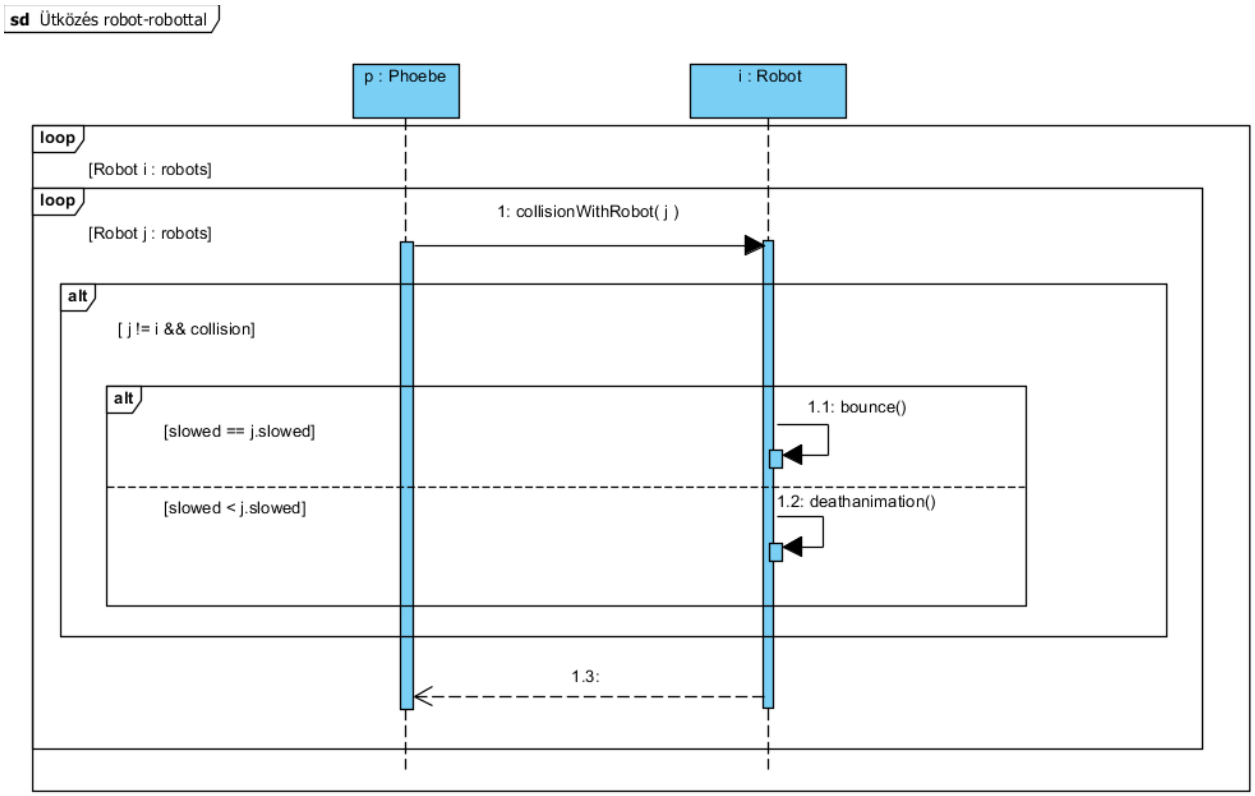


0.1. ábra. Osztálydiagram

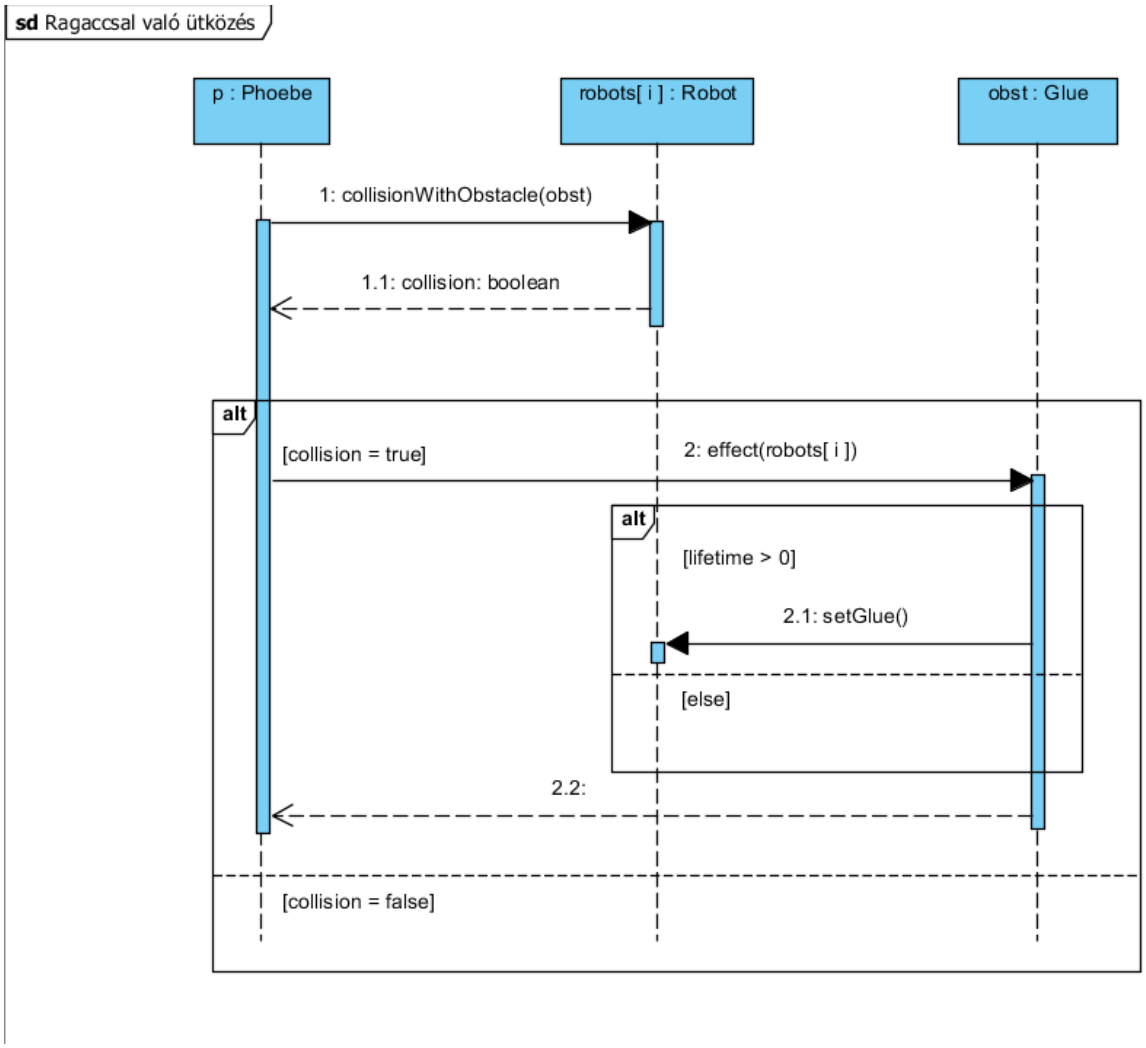
0.3. Szekvencia diagramok



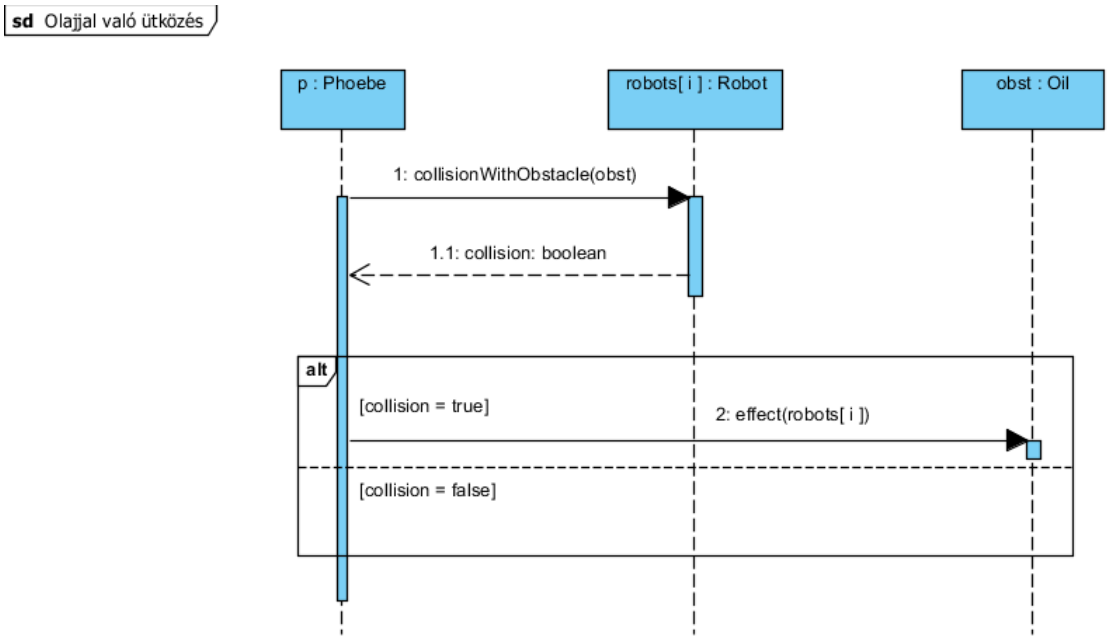
0.2. ábra. Robot ütközése takarító robottal



0.3. ábra. Robot ütközése másik robottal



0.4. ábra. Robot ütközése ragaccsal



0.5. ábra. Robot ütközése olajjal

7. Prototípus koncepciója

7.1. Prototípus interface-definíciója

7.1.1. Az interfész általános leírása

A prototípus csak a szabványos ki- és bemeneteket használja kommunikációra. Így az input fájlból is beolvasható, illetve az output fájlba is menthető. Ezáltal előre elkészített teszteseteket használva automatikusan is lehet tesztelni. Egy teszteset a prototípusnak adott parancsok sorozatából és az arra kapott kimenetből áll. Egy teszt sikeres, ha a kapott- és az elvárt kimenet megegyezik.

7.1.2. Bemeneti nyelv

- chechpointSearch
  - Leírás: Checkpoint keresése a robot helyzetén.
  - Opciók: -
- collisionWithCleaner
  - Leírás: Robot ütközése a takarító kisrobottal.
  - Opciók: -
- collisionWithObstacles
  - Leírás: Robot ütközése akadályokkal.
  - Opciók: -
- collisionWithRobot
  - Leírás: Robot ütközése a másik robottal.
  - Opciók: -
- effect
  - Leírás: Az akadály hatása a robotra.
  - Opciók: -
- Glue
  - Leírás: Ragacs létrehozása a pályán.
  - Opciók: A ragacs koordinátái.
- keyPressed
  - Leírás: Akadály lerakása a robot koordinátáira.
  - Opciók: Adott gomb megnyomása.
- listCleaners
  - Leírás: A palyán lévő takarító kisrobotok kilistázása.
  - Opciók: -
- listObstacles
  - Leírás: Kilistázza a pályán lévő akadályokat.

  - Opciók: -

- listRobots
  - Leírás: Kilistázza a robotokat.
  - Opciók: -
- MapBuilder
  - Leírás: Pálya betöltése a játékba.
  - Opciók: Pálya kiválasztása.
- move
  - Leírás: Robot mozgása.
  - Opciók: Adott gombok lenyomása az irányváltoztatáshoz.
- Oil
  - Leírás: Olajfolt létrehozása a pályán.
  - Opciók: Az olajfolt koordinátái.
- robotOutsideOfMap
  - Leírás: A robot koordinátái a pályán kívülre esnek.
  - Opciók: -
- setCheckpoints
  - Leírás: A pálya checkpointjainak a beállítása.
  - Opciók: -

7.1.3. Kimeneti nyelv

jelölés:<osztály tagváltozója>

- Robot:
  - ToString(): (nem írja ki automatikusan a szöveget csak visszaad egy stringet, amit a Phoebe run metódusában kell kiíratni)
    - "Robot [id=<id>, slowed=<slowed>,oiled=<oiled>, x=<x>,y=<y>, numGlue=<numGlue>,numOil=<numOil>,nextx=<arrowendx>, nexty=<arrowendy>,alpha=<alpha>,width=<WIDTH>,height=<HEIGHT>]"
  - Keypressed(int k):
    - "nextx ,nexty modified to:<arrowendx>,<arrowendy>"
    - "new oil created at: <x>,<y>" (ha volt olajunk és k == VK_DOWN)
    - "not enough oil"(ha nincs olajunk és k==VK_DOWN)
    - "new glue created at:<x>,<y>"(ha volt ragacsunk és k== VK_UP)
    - "not enough glue"(ha nincs ragacsunk és k==VK_UP)
  - CollisionWithRobot(Robot robot2):
    - "there was a collision between this: <this.toString()> and this: <robot2.toString()>"
  - deathAnimation():

        * "This:<robot.toString()> died"
    - CollisionWithObstacle(Obstacle o):
        * "there was a collision between this: <this.toString()> and this: <o.toString()>"

- Oil:
    - effect(robot r):
        * "you jumped into oil"
    - toString():
        * "Oil[x=<x>, y=<y>, Width=<WIDTH>, Height=<HEIGHT>, remainingturns=<Turnsremaining>]"

- Glue:
    - effect(robot r):
        * "you jumped into glue"
    - toString():
        * "Glue [x=<x>, y=<y>, Width=<WIDTH>, Height=<HEIGHT>,remaininglife=<Liferemaining>]";

- Cleaner:
    - toString():
        * "Cleaner[x=<x>, y=<y>, Width=<WIDTH>, Height=<HEIGHT>, State=<state>]"
    - CollisionWithRobot(Robot robot2):
        * "there was a collision between this: <this.toString()> and this: <robot2.toString()>"
    - cleanUp(Obstacle o):
        * " this: <o.toString()> was cleaned up"
    - CollisionWithObstacle(Obstacle o):
        * "there was a collision between this: <this.toString()> and this: <o.toString()>"

7.2. Összes részletes use-case

Use-case neve	**chechpointSearch**
Rövid leírás	A robot egy checkpointba ugrott.
Aktorok	Tesztelő
Forgatókönyv	A program megnézi, hogy a robot olyan területre ugrott, ahol checkpoint található. Ha beleugrott, megnöveli a robot „checkpoint elérésének” számát.

Use-case neve	**collisionWithCleaner**
Rövid leírás	Robot és kisrobot területe metszik egymást.
Aktorok	Tesztelő
Forgatókönyv	Ha a metszet nem üres, akkor a kisrobot meghal.

Use-case neve	**collisionWithObstacles**
Rövid leírás	Robot és akadály területe metszik egymást.
Aktorok	Tesztelő
Forgatókönyv	Ha a metszet nem üres, akkor beállítja a robotot a kívánt hatásra.

Use-case neve	**collisionWithRobot**
Rövid leírás	Robot és robot területe metszik egymást.
Aktorok	Tesztelő
Forgatókönyv	Ha a metszet nem üres és ugyanakkora a sebességük, akkor lepattannak egymásról. Ha különböző a sebességük, akkor a lassabb meghal.

Use-case neve	**effect**
Rövid leírás	Akadály hatása a robotra.
Aktorok	Tesztelő
Forgatókönyv	Ha robot akadályba lép, akkor az adott akadálynak megfelelő változtatás hajtódik végre a roboton.

Use-case neve	**Glue**
Rövid leírás	Ragacs létrehozása
Aktorok	Tesztelő
Forgatókönyv	A tesztelő a pálya megadott koordinátáira ragacsot helyez el.

Use-case neve	**keyPressed**
Rövid leírás	A robot koordinátáira akadályt tesz le.
Aktorok	Tesztelő
Forgatókönyv	Robot ugrása előtt a tesztelő az adott gombbal akadályt rak le a robot helyére.

Use-case neve	**listCleaners**
Rövid leírás	Kilistázza a kisrobotokat.
Aktorok	Tesztelő
Forgatókönyv	Kiírja a kimenetre a játékban levő kisrobotokat.

Use-case neve	**listObstacles**
Rövid leírás	Kilistázza az akadályokat.
Aktorok	Tesztelő
Forgatókönyv	Kiírja a kimenetre a játékban levő akadályokat (ragacsokat és olajfoltokat).

Use-case neve	**listRobots**
Rövid leírás	Kilistázza a robotokat
Aktorok	Tesztelő
Forgatókönyv	Kiírja a kimenetre a játékban levő robotokat.

Use-case neve	**MapBuilder**
Rövid leírás	Pálya betöltése
Aktorok	Tesztelő
Forgatókönyv	A tesztelő által kiválasztott pálya betöltése.

Use-case neve	**Move**
Rövid leírás	Robot irányának beállítása.
Aktorok	Tesztelő
Forgatókönyv	Robot ugrása előtt a tesztelő beállítja a robot irányát az irányváltoztató gombokkal (0-180).

Use-case neve	**Oil**
Rövid leírás	Olajfolt létrehozása
Aktorok	Tesztelő
Forgatókönyv	A tesztelő a pálya megadott koordinátáira olajfoltot helyez el.

Use-case neve	**robotOutsideOfMap**
Rövid leírás	Robot kiugrik a pályáról.
Aktorok	Tesztelő
Forgatókönyv	Ha a robot kiugrik a pályáról, akkor meghal.

Use-case neve	**setCheckpoints**
Rövid leírás	Checkpointok beállítása
Aktorok	Tesztelő
Forgatókönyv	Beállítja a pályára a checkpointokat.

7.3. Tesztelési terv

Teszt-eset neve	**Mozgás**
Rövid leírás	A robot mozgásának tesztelése.
Teszt célja	Teszteli, hogy a robot mozgása után az előre megadott értékeknek megfelelően történik a mozgás (a mozgás után a megfelelő helyen van).

Teszt-eset neve	**Irányváltoztatás**
Rövid leírás	A robot irányváltoztatásának tesztelése.
Teszt célja	Teszteli, hogy a robot különböző irányok beállítása után is a megfelelő helyre ugrik.

Teszt-eset neve	**Ütközés**
Rövid leírás	A robotok ütközésének tesztelése.
Teszt célja	Teszteli, hogy a két robot ütközése után a megfelelő helyre pattannak-e, illetve hogy az objektumok határai helyesen működnek.

Teszt-eset neve	**Leesés**
Rövid leírás	A robot leesésének tesztelése.
Teszt célja	Teszteli, hogy a pálya szélén a robot megszűnik-e létezni.

Teszt-eset neve	**Checkpoint**
Rövid leírás	A robot checkpoint-ba érésének tesztelése.
Teszt célja	Teszteli a checkpoint-ba érést, illetve hogy a robotnak megfelelően változnak-e az értékei ilyen esetben.

Teszt-eset neve	**Olajba ugrás**
Rövid leírás	A robot olajba ugrását teszteli.
Teszt célja	Teszteli, hogy egy olajba ugrás alkalmával változnak-e a robot értékei, vizsgálja az ütközést.

Teszt-eset neve	**Ragacsba ugrás**
Rövid leírás	A robot ragacsba ugrását teszteli.
Teszt célja	Teszteli, hogy egy ragacsba ugrás alkalmával változnak-e a robot értékei, vizsgálja az ütközést.

Teszt-eset neve	**Olaj lerakás**
Rövid leírás	Olaj lerakása a pályán.
Teszt célja	Teszteli, hogy a lerakás után létrejött-e az olaj a pályán.

Teszt-eset neve	**Ragacs lerakás**
Rövid leírás	Ragacs lerakása a pályán.
Teszt célja	Teszteli, hogy a lerakás után létrejött-e a ragacs a pályán.

Teszt-eset neve	**Olaj hatás**
Rövid leírás	Az olaj hatását teszteli.
Teszt célja	Teszteli, hogy olajos állapotban a robot képes-e irányt változtatni.

Teszt-eset neve	**Ragacs hatás**
Rövid leírás	A ragacs hatását teszteli.
Teszt célja	Teszteli, hogy ragacsos állapotban a robot milyen távolságra képes ugrani.

7.4. Tesztelést támogató segéd- és fordítóprogramok specifikálása

Az egyes tesztesetek futtatásához a JUnit tesztelő keretrendszert használjuk. A segédprogram egy-egy teszthez tárolja a bemeneteket illetve az adott bemenethez várt kimenetet. A segédprogram lefuttatja a megadott teszteket, és összeveti a kimenetüket az előre definiált elvárt kimenettel. Ha a teszt kimenete nem egyezik meg az elvárt kimenettel, akkor a teszt nem sikerült, ellenkező esetben a teszt sikeres. Hiba esetén a segédprogram megjeleníti, hogy hol talált eltérést.

7.5. Napló

Kezdet	Időtartam	Résztvevők	Leírás
2015.03.25. 10:00	2,5 óra	**Kovács** **Lovász** **Tóth** **Graics** **Magyar**	Konzultáció, részfeladatok kiosztása
2015.03.25. 16:00	3 óra	**Tóth**	Dokumentáció frissítése
2015.03.25. 16:00	3 óra	**Kovács**	Kimeneti nyelv
2015.03.28. 10:00	2,5 óra	**Lovász**	Bemeneti use-casek dokumentálása
2015.03.28. 13:00	2,5 óra	**Graics**	Tesztelési terv dokumentálása
2015.03.29. 12:00	3 óra	**Magyar**	Dokumentáció szerkesztése
2015.03.29. 18:00	1 óra	**Tóth**	Osztálydiagram, szekvencia diagramok frissítése
2015.03.29. 20:00	1 óra	**Graics**	Bemeneti nyelv dokumentálása