2. Követelmény, projekt, funkcionalitás

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. február 22.

Tartalomjegyzék

Ábrák jegyzéke

2. Követelmény, projekt, funkcionalitás

2.1. Bevezetés

2.1.1. Cél

A projekt követelményeinek, alapvető felépítésének és funkcionalitásának ismertetése, ezek segítségével a fejlesztési folyamatok majd a végleges program felépítésének megtervezése. Fejlesztés közben ezeket figyelembe kell majd venni és tőlük eltérni nem szabad.

2.1.2. Szakterület

A projekt a Szoftver laboratórium 4. tantárgy feladatának megoldására született. Az elkészítendő szoftver egy számítógépes játék, nem kimondottan egy szakterület részére készül, inkább szórakoztatás céljából.

2.1.3. Definíciók, rövidítések

- **BME** Budapesti Műszaki Egyetem rövidítése.
- **Doxygen** Forráskódból és kommentekből dokumentációt generáló program.
- **Eclipse** Fejlesztőkörnyezet Java programozási nyelvhez.
- **Git** Verziókezelői rendszer.
- **Google Calendar** Google egy szolgáltatása, melyben eseményeket lehet beállítani naptárszerűen. Ezek megoszthatók más felhasználókkal is.
- **Google Mail** Google egy szolgáltatása, mely az internetes levelezést biztosítja a felhasználóinak.
- **HSZK** Hallgatói Számítógép Központ rövidítése.
- **HUD** Head-Up Display rövidítése. Egy szem elé vetített kijelző, melyen a fontosabb, illetve alapvető adatokat jelenítik meg.
- **IDE** Integrated Development Environment (Integrált Fejlesztői Környezet) rövidítése. A számítógép programozást megkönnyítő programok ezek.
- **IntelliJ** Java IDE, megkönnyíti a fejlesztést.
- **JDK** Java Development Kit (Java Fejlesztői Készlet) rövidítése. A Java nyelv fejlesztőeszköze.
- **JRE** Java Runtime Environment rövidítése. Program, ami szükséges a Javában megírt fájlok elkészítéséhez, futtatásához.
- **LaTex** Dokumentum formázó rendszer, mellyel magas színvonalú dokumentációk készíthetők.
- **Netbeans** Fejlesztőkörnyezet, ami Java nyelven alapul. A programban egyszerűen, gyorsan lehet grafikus felületet létrehozni, például menüket.
- **PC** Personal Computer rövidítése. Jelentése: személyi számítógép, manapság a legtöbb számítógép ilyen.
- **Prototípus** A program azon állapota, melyben meg van valósítva minden funkció működőképesen, de a grafikus felület még nincs beleépítve.

- **ShareLaTeX** Internetes LaTeX szerkesztő, mely lehetővé teszi dokumentumok szerkesztését, előkészítését valós idejű online környezetben.
- **Sympa** Egy szoftver, mely lehetővé teszi a levelező listák használatát, menedzselését.
- **Szkeleton** A program azon állapota, melyben a belső felépítés készen van (a függvények osztályok), de még nem csinál semmit.
- **Szoftver** A számítógépen elhelyezhető, futtatható program.
- **Use-case** Lehetséges interakciók leírása a rendszer és az aktor (felhasználó) között.
- **Verziókezelés** Olyan tevékenység, amellyel eltárolhatóak a fájlok régebbi verzió, ezáltal egy-egy módosítás után vissza lehet térni egy adott verzióhoz és nyomon lehet követni a fejlesztést.

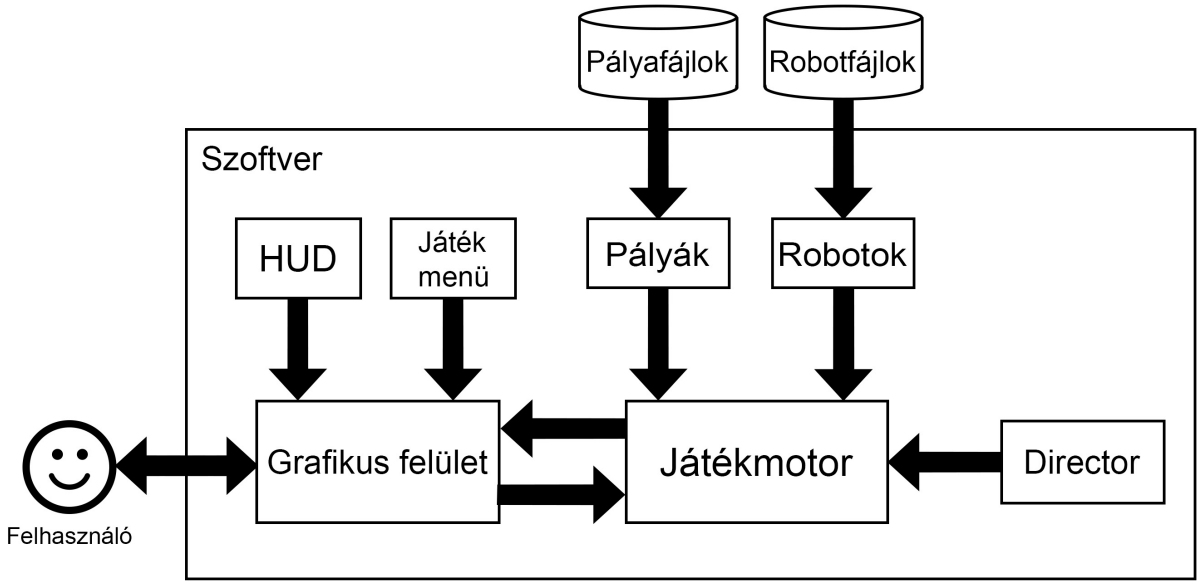
2.1.4. Hivatkozások

Szoftver Labor 4 - `https://www.iit.bme.hu/~szoftlab4/`
GitHub - `https://github.com/tht-krisztian/Szoftlab4.git`

2.1.5. Összefoglalás

A dokumentum további részeiben található:

- 2.2 Áttekintés: A projekt terveinek, funkcióinak áttekintése, a felhasználók lehetőségeinek áttekintése
- 2.3 Követelmények: A projekt során elvárt követelmények kidolgozása, külön kitérve funkcionális, erőforrás, átadással kapcsolatos és egyéb nem funkcionális követelményekre.
- 2.4 Lényeges use-case-ek felsorolása, use-case diagram
- 2.5 Szótár: A projekt során bevezetett fogalmak körülírása.
- 2.6 Projekt terv: A projekt résztvevőinek feladatkörei és határidők részletes kifejtése
- 2.7 Napló: Az elvégzett feladatok és ráfordított idő felsorolása



2.1. ábra. Architekturális kép

2.2. Áttekintés

2.2.1. Általános áttekintés

A szoftver fontos eleme a Játékmotor. Ez teremti meg a kezdeti feltételeket és irányítja a játék folyását. Ez fogja ellenőrizni, hogy a robotok leesnek-e a pályáról, ha olajfoltba lépnek letiltja az irányítást a játékos oldalon, illetve ha ragacsba lép egy játékos, akkor lelassuljon. A Játékmotor indulása előtt lekéri a Pályák alrendszertől a kiválasztott pályát.

A director alrendszer feladata az idő léptetése, a robotok másodpercenkénti mozgását ütemezi.

A HUD alrendszer hivatott jelezni, hogy a játékos számára elérhető-e az olaj vagy ragacskészlet, továbbá a megtett körök számát.

A játék menü alrendszer feladata a játék során a megfelelő parancsra megállítani a játékot és a beállítások (pl.: grafikai, hang) állíthatósága.

A grafikus felület feladata, hogy kapcsolatot létesítsen a játékossal vizuális információkkal és a játékos ezen keresztül irányíthatja a játékmotort. Továbbá hozzáfér a játék menühöz és a HUD-hoz, hiszen ki kell jeleznie azokat.

A Pályák alrendszer betölti a Pályafájlokból a pályákat, tárolja, majd azokat a Játékmotor rendelkezésére bocsájtja.

A Robotok alrendszer betölti a robotokat a Robotfájlokból, tárolja azokat, majd a Játékmotor rendelkezésére bocsájtja.

Hálózatot a szoftver nem igényel. Háttértáron pedig a program .jar archívumainak, a kirajzoláshoz szükséges pálya és robot képeknek, a pályákat tartalmazó fájloknak szükséges helyet biztosítani. Ezeken kívül nincs szükség futás idejű tárhelyigényre.

2.2.2. Funkciók

Mivel unalmas a mérnökök munkája a Marson (nem adnak nekik elég érdekes feladatot, mint például a napelem-építés, vagy az ásványi anyag gyűjtése), ezért robotokat versenyeztetnek egymás ellen; ezt valósítja meg az

alábbi játékprogram.

Egy versenyző játéknak a lényege, hogy a résztvevők egymás ellen próbálnak minél tovább eljutni a pályán, illetve minél jobb időt elérni. Ez itt sincs másként, a verseny célja, hogy a megadott idő alatt több kört tegyünk meg robotunkkal mint az ellenfél. Ebben segítségünkre lehet a pálya kialakítása, és a különböző felszerelések, melyek megnehezíthetik a verseny lefutását. Fontos, hogy a robotok a kijelölt pályán maradjanak, mert csak ezen a területen biztosított a kommunikáció a vezérlőközponttal. Ha egy robot elhagyja a pályát diszkvalifikálják a futamból.

Az idő lejártával (esetleg más játékmódoknál egy adott körszám megtétele után) a játék véget ér. Ebben az esetben az a játékos robotja nyer, amelyik több kört tett meg (vagy elsőként ért be a célba adott kör után), kivéve ha idő előtt letér a pályáról, ezáltal nem vesz részt az eredményhirdetésen a célvonalnál.

A játékteret a csodálatos Mars környezete adja, ami nagyon veszélyes terep. Tartani kell a különféle sugárzásoktól, nehogy kárt tegyenek a mérnökökben és a nézőkben, ezért ők egy védett szobából szemlélik és irányítják az eseményeket. A pályán különféle nehezítések is helyet kapnak, hogy ne legyen a robotoknak (és mérnökeiknek) könnyű az élete. Olyan akadályokkal kell megbirkózniuk a robotoknak, mint például a lyukak elkerülése, melyek a felépítésüket teszik tönkre, ezáltal nem tudnak tovább versenyezni, vagy az olajfoltok időben való észrevétele, ami meggátolja, hogy a mérnökök irányítsák őket, végül de nem utolsó sorban a ragacsokat is ki kell kerülniük, mert azok a sebességüket csökkentik, ezáltal nagy előnyt adva az ellenfél robotjának. A többi robothoz sem éri meg közel tartózkodni, hiszen mint minden verseny alatt, az ütközésnek itt is ára van, ez most az irányítás bezavarodásában merül ki. Végül a legnagyobb akadály maga a pálya. A versenyeket egy lapos fennsíkon szokták tartani a Marson, jól elszeparálva a marslakóktól, hogy azok ne tudjanak belekontárkodni a futamba, de elakadhatnak vagy leeshetnek a szélén, a mérnököknek erre is nagyon kell figyelniük az irányítás alatt.

A robotok a startvonalról indulnak úgy, hogy figyelembe véve a pálya felépítését, igazságos legyen minden résztvevőnek (a mérnökök a pályán kívülről irányítják őket). A „Start”-ot egy visszaszámolás indítja, melynek következtében a robotok elindulnak. A pályán állomások találhatók, melyek minden robot áthaladásánál ellenőrzik, hogy nem történt-e csalás. Minden körben a pályán adott állomásokon kell áthaladni adott sorrendben. A kör csak akkor érvényes, ha a robot ezeknek a megszorításoknak eleget tud tenni. Továbbá a távolság mérése (ezáltal győzelmi feltételek) is ez alapján történik, szóval egyik mérnöknek sem éri meg ide-oda navigálni a robotját a pontok között, a legnagyobb megtett út reményében. A robotok sebessége állandó, a mérnökök csak az irányt tudják befolyásolni a játék lezajlása alatt, ezért nagyon kell figyelni, hogy a kanyarokat jó ívben vegyék be, az akadályokat pedig megfelelő szakértelemmel kerüljék ki. Továbbá az indulás után máris lehetőséget nyújt a játék, hogy a mérnökök használják a robotokba beépített felszereléseket. Ezek nem mások mint a már feljebb leírt ragacsok és olajfoltok. Ezeket maguk mögött képesek hagyni, és a játék végéig megmaradnak a pálya felületén (illetve amíg valaki bele nem megy, akár a mérnök saját robotja), fejfájást okozva ezzel a versenyzőknek. A felszerelések körönként (fentebb említett „Checkpoint-rendszer” szerint) újratöltődnek, ezért a mérnököknek nem kell taktikázniuk, hogy mikor törjenek borsot az ellenfelük orra alá; körönként megtehetik azt.

A verseny ezen lehetőségek megvalósítása által igen érdekes, veszélyes és néha akár frusztráló is, gondoljunk csak a robotok felszereléseire. Egyszóval ideális időtöltés a Marson unatkozó mérnököknek.

2.2.3. Felhasználók

A szoftver kétjátékos módban indíthat. A játékosnak semmiféle előképzettségre nincs szüksége, a játék menete gyorsan elsajátítható. A játékban nem lesz pályaszerkesztő. A játékot nem lehet elmenteni, mindig újból kell kezdeni.

2.2.4. Korlátozások

A BME IIT Szoftver Laboratórium tantárgy oktatói (a megrendelők) által kiírt specifikáció megköveteli, hogy a program a HSZK gépein a beadások alkalmával futtathatók legyenek. Ezekre a gépekre már előre feltelepített JRE 1.6-es környezet elérhető, ezért csak a JDK 1.6-es verziójaban már létező függvények, osztályok

használhatóak a kód megírása során.

2.2.5. Feltételezések, kapcsolatok

Szoftver Labor 4

- feladat határidői `https://www.iit.bme.hu/~szoftlab4/`
- feladatkiírás `https://www.iit.bme.hu/~szoftlab4/feladat.shtml`
- dokumentáció sablonok `https://www.iit.bme.hu/~szoftlab4/templ/`

2.3. Követelmények

2.3.1. Funkcionális követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Use-case	Komment
1.01	Játék menü	bemutatás	alapvető	csapat		
1.02	Beállítások kezelése	bemutatás	opcionális	csapat		
1.03	Játék elindítása	bemutatás	alapvető	megrednelő	Új Játék indítás	
1.04	Pályáról leesők kiesnek	bemutatás	alapvető	megrendelő		
1.05	Előre elkészített versenypálya	bemutatás	alapvető	megrendelő		
1.06	Robotok a kezdőpozíciójukból indulnak	bemutatás	alapvető	megrendelő		
1.07	Pályán vannak olajfoltok és ragacsfoltok	bemutatás	alapvető	megrendelő		
1.08	Robotok fel vannak szerelve olaj és ragacskészlettel	bemutatás	alapvető	megrendelő		
1.09	2 személy tudjon játszani egyszerre	bemutatás	alapvető	csapat		
1.10	Körlimites játékmód	bemutatás	opcionális	csapat		
1.11	Időlimites játékmód	bemutatás	alapvető	megrendelő		
1.12	A robotok tudjanak ugrani	bemutatás	alapvető	megrendelő		
1.13	A ragacs és olaj a pályáról idő után eltűnik	bemutatás	opcionális	csapat		
1.14	A ragacs és olaj a pályáról, ha belelépnek eltűnik	bemutatás	opcionális	csapat		
1.15	A robotok tudnak egymással ütközni	bemutatás	opcionális	csapat		
1.16	Az indulás előtt visszaszámlálás indul	bemutatás	opcionális	csapat		
1.17	A robotok sebessége egységnyi méretű tetszőleges irányú vektorral módosítható	bemutatás	alapvető	megrendelő		
1.18	Egy ugrással a sebességgel egyenesen arányos távolságra tudnak eljutni	bemutatás	alapvető	megrendelő		
1.19	A robot ragacsra érkezve sebessége a felére csökken	bemutatás	alapvető	megrendelő		

1.20	A robot olajfoltra érkezve sebességének módosítása nem lehetséges	bemutatás	alapvető	megrendelő		
1.21	A robot olajat vagy ragacsot tudnak lerakni	bemutatás	alapvető	megrendelő	Akadá-lyozás	
1.21	A játékból ki lehet lépni	bemutatás	alapvető	megrendelő	Kilépés	

2.3.2. Erőforrásokkal kapcsolatos követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
2.01	JRE	bemutatás	alapvető	megrendelő	Java IDE
2.02	Eclipse	nincs	opcionális	csapat	Java IDE
2.03	NetBeans IDE	nincs	opcionális	csapat	Java IDE
2.04	IntelliJ IDEA	nincs	opcionális	csapat	Java IDE
2.05	ShareLatex	nincs	opcionális	csapat	online LaTeX editor
2.06	ShareLatex	nincs	opcionális	csapat	online LaTeX editor
2.06	Git	nincs	alapvető	csapat	Elosztott verziókezelő
2.07	GitHub account	nincs	alapvető	csapat	Git tárhely
2.08	PC	bemutatás	alapvető	megrendelő	
2.09	Monitor	nincs	alapvető	csapat	
2.10	Billentyűzet	nincs	alapvető	csapat	
2.11	Egér	nincs	alapvető	csapat	
2.12	Google Calendar	nincs	opcionális	csapat	Naptár alkalmazás
2.13	Google Mail	nincs	opcionális	csapat	Levelező rendszer
2.14	Sympa levelezőlista	nincs	opcionális	csapat	`lists.sch.bme.hu`
2.15	Doxygen	nincs	opcionális	csapat	Programozói dokumentáció
2.16	ObjectAid	nincs	opcionális	csapat	UML diagrammok
2.17	Sparx Enterprise Architect	nincs	opcionális	csapat	UML diagrammok

2.3.3. Átadással kapcsolatos követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
3.01	Szkeleton átadás	bemutatás	alapvető	megrendelő	márc. 23
3.02	Prototípus átadás	bemutatás	alapvető	megrendelő	ápr. 20
3.03	Teljes program átadása	bemutatás	alapvető	megrendelő	máj. 15
3.04	Útmutató alapján telepíthető, indítható	bemutatás	fontos	megrendelő	
3.06	A programnak működnie kell a BME HSZK számítógépein	bemutatás	alapvető	megrendelő	

2.3.4. Egyéb nem funkcionális követelmények

Nincs egyéb nem funkcionális követelmény.

2.4. Lényeges use-case-ek

2.4.1. Use-case leírások

Use-case neve	Új Játék indítás

Rövid leírás	Új Játék indul
Aktorok	Játékos
Forgatókönyv	A Játékos kiválasztja a „New Game”-t. Ezután két módból választhat: • Kör: adott kört kell megtenni • Idő: adott idő alatt kell elérni minél nagyobb távolságot

Use-case neve	**Mozgás**
Rövid leírás	Robot mozgása
Aktorok	Játékos
Forgatókönyv	A játékos képes megváltoztatni a robot mozgásának irányát a jobbra és balra gombokkal.

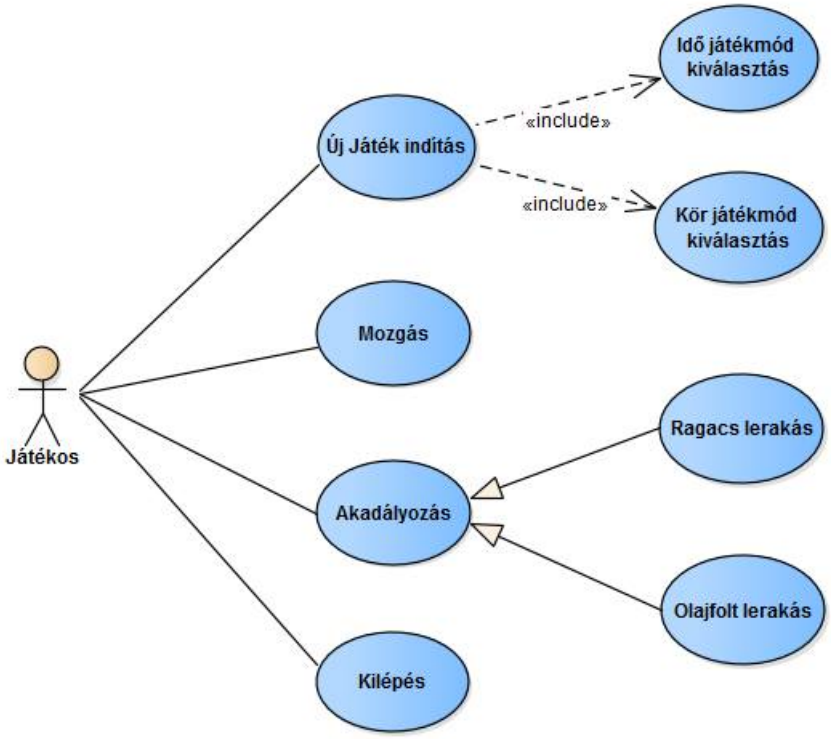
Use-case neve	**Akadályozás**
Rövid leírás	Ellenfél lassítása, iránymentesítése
Aktorok	Játékos
Forgatókönyv	A Játékos képes az ellenfél robotját akadályozni. Ezt kétféleképpen tudja: - ragacsot tesz le: az ellenfél lassabbá válik - olajfoltot tesz le: az ellenfél nem tudja megváltoztatni egy ideig a robotja irányát

Use-case neve	**Kilépés**
Rövid leírás	Az alkalmazás bezárása
Aktorok	Játékos
Forgatókönyv	A Játékos kilép a programból, ha a jobb felső sarokban lévő bezárásra kattint.

2.4.2. Use-case diagram

2.5. Szótár

- **Játékos** - A program felhasználója, a robot irányítója, aki játszik.
- **Robot** - A játékos által irányított virtuális gépezet. Képes ugrani, ragacsot és olajat helyezni a pályára, illetve le tud esni a pályáról.
- **Verseny** – A játék folyamata. Célja minél több kör megtétele adott idő alatt vagy adott körszám megtétele minél rövidebb idő alatt.
- **Pálya** - A verseny helyszíne: az a sík, amin a játék zajlik. A játék közben nem változik, a robotok ide helyezhetik az olajfoltokat és a ragacsokat.
- **Start** – Az az időpillanat, amikor a játék elindul és a játékosok megkezdik a versenyzést. Ezt egy visszaszámlálás előzi meg.
- **Checkpoint** – Olyan vonal a pályán, ami ellenőrzi, hogy a játékos megfelelő irányban halad és nem próbál szabálytalanságot elkövetni.



2.2. ábra. Use-Case diagram

- **Célvonal** – Ha játékosok adott körszám megtételéig versenyeznek, akkor az a Checkpoint, amit bizonyos körszám után elérve a játéknak vége van.
- **Startvonal** – Az a Checkpoint, amiről a robotok elindulnak.
- **Idő lejárta** – Az a pillanat, amikor a játékosok rendelkezésére álló idő elfogy, eléri a nullát. Ilyenkor az adott játéknak vége van.
- **Kör** - Egy vonal a pályán, ami minden Checkpointot pontosan egyszer érint és a kezdő- és végpontja a startvonal.
- **Távolság** – Az a mértékegység, ami megmondja, hogy egy robot milyen hosszú utat tett meg a pályán a verseny ideje alatt.
- **Zászlók** – Az az indikátor, ami jelzi, hogy a játékos éppen hány Checkpointon jutott át az adott körben. Minden kör végén törlődik, de a megtett körök száma megnő eggyel.
- **Akadály** - Olyan objektum a pályán, ami megnehezíti a játékosok célba jutását. Lehet állandó, ami megmarad a játék egész ideje alatt, illetve ideiglenes, ami eltűnik, amint egy robot rálép.
- **Lyuk** – Állandó akadály, amit a pálya előre tartalmaz, tehát nem a játékosok helyezik el. Ha egy robot rálép, akkor tönkre teszi annak felépítését, ezáltal megakadályozza a verseny folytatásában.
- **Felszerelés** – Ragacs vagy olaj, amit a robot magánál hord és bármelyik lépésnél a pályára helyezhet. Lehelyezése után ideiglenes akadályként funkcionál.
- **Ragacs** – Olyan ideiglenes akadály, ami felezi a rálépő robot sebességének a nagyságát.

- **Olaj** – Olyan ideiglenes akadály, ami megakadályozza a rálépő robot irányának megváltoztatását.
- **Irányítás** – Az a folyamat, amikor a játékos a hozzá tartozó robot irányát megváltoztatja.
- **Irány** – A robot sebességének az a komponense, ami meghatározza, hogy a robot merre halad.
- **Sebesség** – Egy kétdimenziós vektor a pálya síkjában, ami eredendően egységnyi méretű minden robotra. Méretét csak ragacsok tudják változtatni. Irányát a játékosok irányítása, illetve az ütközések módosítják.
- **Ugrás** – A robotok képesek átugrani egy akadályt. Az ugrás nagyságát a robothoz tartozó sebességvektor nagysága határozza meg.
- **Leesés** – Az az esemény, amikor egy robot a pálya felületén kívülre kerül, ezáltal elhagyva a pályát.
- **Ütközés** – Az az esemény, amikor a két robot találkozik egymással, azaz minimum érintik egymást. A robot irányát ilyenkor a játékos nem változtathatja, hanem az ütközésnek megfelelően módosul.
- **Diszkvalifikáció** - A pálya elhagyásának, a pályáról való leesésnek a következménye. Ilyenkor a pályát elhagyó robotot irányító játékos veszít.
- **Szabálytalanság** – Az az eset, amikor a játékos nem az előírt sorrendben próbál végighaladni a Checkpointokon.

2.6. Projekt terv

2.6.1. Csapat

A csapat 5 főből áll. A következő táblázat a személyes preferenciákat tartalmazza, a feladatokat úgy osztjuk ki, hogy közel azonos nehézségűek legyenek, miközben ezeket a preferenciákat szem előtt tartjuk.

Név	**Felelősségek**
Kovács Levente Ákos	UML, kód
Lovász Attila Bence	UML, kód
Graics Vince	Dokumentáció, kód
Magyar Milán Bertalan	Grafika, Dokumentáció, kód
Tóth Krisztián Dávid (csapatvezető)	Menedzsment, Dokumentáció, kód

2.6.2. Kommunikáció

- **Levelezőlista** A csapat üzemeltet egy levelezőlistát a Sympa rendszeren belül, ez az egyik formája a kommunikációnknak.
- **Megbeszélések** Heti két-három alkalommal megbeszéléseket tartunk. Illetve csoporttársak révén szinte minden óránk egy időben van, tehát folyamatos kapcsolatban állunk egymással.

2.6.3. Használt programok

- **Verziókezelés**

  A Github nevű szolgáltatást használjuk, mely alkalmas a több személyes szoftver fejlesztésre.

- **Fejlesztőkörnyezet**

  A kódot Eclipse környezetben írjuk, Java nyelven. Emelett a Sparx Enterprise Architect nevű alkalmazást használjuk az UML diagrammok készítésére.

2.6.4. Mérföldkövek, határidők

Dátum	Leírás	Ellenőrzés
2015.02.23	Követelmény, projekt, funkcionalitás	beadás
2015.03.02	Analízis modell kidolgozása 1.	beadás
2015.03.09	Analízis modell kidolgozása 2.	beadás
2015.03.16	Szkeleton tervezése	beadás
2015.03.23	Szkeleton	beadás
2015.03.25	**Szkeleton**	bemutatás
2015.03.30	Prototípus koncepciója	beadás
2015.04.07	Részletes tervek	beadás
2015.04.20	Prototípus	beadás
2015.04.22	**Prototípus**	bemutatás
2015.04.27	Grafikus felület specifikációja	beadás
2015.05.11	Grafikus változat	beadás
2015.05.13	**Grafikus változat**	bemutatás
2015.05.15	Összefoglalás	beadás

A **szkeleton** változat célja annak bizonyítása, hogy az objektum és dinamikus modellek a definiált feladat egy modelljét alkotják. A szkeleton egy program, amelyben már valamennyi, a végső rendszerben is szereplő business objektum szerepel. Az objektumoknak csak az interfésze definiált. Valamennyi metódus az indulás pillanatában az ernyőre szöveges változatban kiírja a saját nevét, majd meghívja azon metódusokat, amelyeket a szolgáltatás végrehajtása érdekében meg kell hívnia. Amennyiben a metódusból valamely feltétel fennállása esetén hívunk meg más metódusokat, akkor a feltételre vonatkozó kérdést interaktívan az ernyon fel kell tenni és a kapott válasz alapján kell a továbbiakban eljárni. A szkeletonnak alkalmasnak kell lenni arra, hogy a különböző forgatókönyvek és szekvencia diagramok ellen őrizhetők legyenek. Csak karakteres ernyőkezelés fogadható el, mert ez biztosítja a rendszer egyszerűségét.

A **prototípus** program célja annak demonstrálása, hogy a program elkészült, helyesen működik, valamennyi feladatát teljesíti. A prototípus változat egy elkészült program kivéve a kifejlett grafikus interfészt. A változat tervezési szempontból elkészült, az ütemezés, az aktív objektumok kezelése megoldott. A business objektumok - a megjelenítésre vonatkozó részeket kivéve - valamennyi metódusa a végleges algoritmusokat tartalmazza. A megjelenítés és működtetés egy alfanumerikus ernyőn követhető, ugyanakkor a megjelenítés fájlban is logolható, ezzel megteremtve a rendszer tesztelésének lehetőségét. Különös figyelmet kell fordítani az interfész logikájára, felépítésére, valamint arra, hogy az mennyiben tükrözi és teszi láthatóvá a program működését, a beavatkozások hatásait.

A **grafikus** (teljes) változat a prototípustól elvileg csak a kezelői felület minőségében különbözhet. Ennek változatnak az értékelésekor a hangsúlyt sokkal inkább a megvalósítás belső szerkezetére, semmint a külalakra kell helyezni.

2.7. Napló

Kezdet	Időtartam	Résztvevők	Leírás
2015.02.16. 16:30	1 óra	**Kovács** **Lovász** **Tóth** **Graics** **Magyar**	Kezdeti megbeszélés. Specifikáció körvonalazása. Lehetséges use-case-ek átbeszélése. Feladatok kiosztása
2015.02.16. 20:15	1,5 óra	**Tóth**	Tevékenység: Specifikáció készítés

Kezdet	**Időtartam**	**Résztvevők**	**Leírás**
2015.02.17. 17:00	1 óra	**Lovász** **Kovács**	Tevékenység: Lehetséges Use-Case-ek megalkotása, Use-Case Diagramm elkészítése
2015.02.17. 20:30	1,5 óra	**Graics**	Tevékenység: Funkcionalitás megírása
2015.02.18. 8:15	1 óra	**Kovács** **Lovász** **Tóth** **Graics** **Magyar**	Funkcionalitások átbeszélése, mérlegelés. Feladatok kiosztása
2015.02.19 18:00	2 óra	**Magyar**	Tevékenység: Szótár megalkotása
2015.02.19 21:00	2 óra	**Graics**	Tevékenység: Definíciók és rövidítések megalkotása
2015.02.22. 14:00	0,5 óra	**Tóth**	Tevékenység: Dokumentáció végső simításai